AD.H

Arenadata Hyperwave

Arenadata Hyperwave (ADH) — это универсальная гибридная платформа, основанная на компонентах с открытым исходным кодом и проприетарной разработке, предназначенная для хранения, обработки и анализа данных любой структуры и объема.

Версия CURRENT Язык: RU

Содержание

1. Установка
  - Online-установка
    - Установка ADCM
    - Подготовка хостов
    - Установка кластера ADH
      - Создание кластера
      - Добавление сервисов
      - Добавление хостов в кластер
      - Добавление компонентов
      - Настройка сервисов
      - Настройка кластера
      - Установка кластера
    - Установка мониторинга
      - Способ 1. Сервис мониторинга
      - Способ 2. Кластер мониторинга
        - Создание кластера
        - Добавление сервисов
        - Добавление хостов в кластер
        - Добавление компонентов
        - Настройка сервисов
        - Установка кластера
        - Интеграция с кластером ADH
  - Offline-установка
    - Установка ADCM
    - Подготовка хостов
    - Установка кластера Enterprise Tools
      - Создание кластера
      - Добавление сервисов
      - Добавление хостов в кластер
      - Добавление компонентов
      - Установка кластера
    - Установка кластера ADH
      - Создание кластера
      - Добавление сервисов
      - Добавление хостов в кластер
      - Добавление компонентов
      - Настройка сервисов
      - Настройка кластера
      - Импорт настроек ET
      - Установка кластера
    - Установка мониторинга
      - Способ 1. Сервис мониторинга
      - Способ 2. Кластер мониторинга
        - Настройка кластера Enterprise Tools
        - Интеграция с кластером ADH

2. Базовые операции
  - Airflow
    - Подключение к Airflow
      - CLI
      - REST API
    - Обзор UI
    - Работа с DAG
      - Создание простого DAG
      - Работа с TaskFlow
      - Использование сенсоров
      - Кастомизация расписания DAG
      - Добавление кастомных операторов и хуков
      - Динамическая генерация DAG
    - Логирование
    - Управление сервисом через ADCM
  - Core configuration
    - Получение клиентских конфигураций
    - Управление сервисом через ADCM
  - Flink
    - Обзор UI
    - Подключение к Flink
      - CLI
      - PyFlink
    - Flink SQL Gateway
    - Запуск Flink в YARN
    - Логирование
    - Оптимизация производительности Flink
    - Создание streaming ETL с помощью Flink
    - Управление сервисом через ADCM
      - Flink
      - Flink2
  - HBase
    - Подключение к HBase
      - Использование нативного Java API
      - Использование внешнего API
    - Обзор UI
    - Выборка данных
      - Начало работы
      - Использование фильтров
      - Использование сопроцессоров
      - Сканирование снепшотов
      - Использование объектов MOB
    - Массовая загрузка
      - Встроенные задачи MapReduce
    - Использование снепшотов
    - Репликация
    - Резервное копирование и восстановление данных
    - Оптимизация производительности
    - Логирование
    - Управление сервисом через ADCM
  - HDFS
    - Подключение к HDFS
    - Обзор веб-интерфейсов
    - Базовые операции с файлами
      - Защита файлов
      - Копирование данных в HDFS
      - Добавление каталогов данных HDFS
      - Замена диска без остановки DataNode
      - Удаление файлов и директорий
      - Настройка WebHDFS
      - Проверка файлов через fsck
      - Локальное чтение данных
      - Использование distcp
      - Использование HttpFS
    - Управление хранением данных
      - Политики хранения
      - Квоты
      - Управление кешем
    - Репликация
      - Фактор репликации
      - Rack awareness
      - Erasure coding
      - Безопасный режим NameNode
    - Резервное копирование и восстановление
      - Снепшоты
      - Восстановление NameNode
    - Настройка
    - Checkpointing
    - Интеграция с S3
    - Логирование
    - Использование Observer NameNode
    - Управление сервисом через ADCM
  - Hive
    - Подключение к Hive
      - Beeline shell
      - JDBC
    - Обзор UI
    - Запросы к БД
      - Работа с таблицами
      - Пример вычислений Hive
      - Многотабличная вставка данных
      - Анализ запросов
      - Типы запросов
        - Подзапросы
        - Агрегатные функции
        - Оконные и аналитические функции
        - Материализованные представления
    - Интеграция
      - Hive on Spark
      - Hive и HBase
      - TEZ
        - Просмотр задач Hive
    - Оптимизация производительности
    - Резервное копирование и восстановление
    - Режим высокой доступности
    - Логирование
    - ACID-транзакции
    - Таблицы Iceberg
    - Управление сервисом через ADCM
  - HUE
    - Обзор UI
    - Добавление и использование интерпретатора
    - Таблицы Iceberg
    - Аутентификация SPNEGO
    - Управление сервисом через ADCM
  - Impala
    - Подключение к Impala
      - impala-shell
      - JDBC
    - Обзор веб-интерфейсов
    - Настройки производительности
    - Интеграция
      - Impala и Hive
      - Impala и HBase
    - Аутентификация LDAP
    - Таблицы Iceberg
    - Управление сервисом через ADCM
  - Kyuubi
    - Обзор UI
    - Аутентификация LDAP
    - Управление сервисом через ADCM
  - MapReduce
    - Начало работы
    - Подключение к MapReduce
      - CLI
      - REST API
    - Логирование
    - Настройка shuffle и sort
    - Encrypted shuffle
  - Monitoring
    - Управление сервисом через ADCM
  - Ozone
    - Подключение к Ozone
      - CLI
    - Обзор UI
    - Интеграция Ozone с сервисами кластера
    - Rack awareness
    - Управление сервисом через ADCM
  - Phoenix
    - Подключение к Phoenix
    - Выборка данных
      - Использование SQLLine
      - JOIN
      - Подзапросы
      - Последовательности
      - Использование salted-таблиц
    - Использование таблиц HBase
    - Массовая загрузка
    - Оптимизация производительности
    - Логирование
  - Solr
    - Подключение к Solr
      - CLI
    - Индексирование
      - Обзор
      - Индексирование вложенных документов
      - Частичное обновление документов
      - Переиндексирование
      - Дедупликация
    - Обзор UI
    - Коллекции Solr
    - Логирование
    - Оптимизация производительности
    - Управление сервисом через ADCM
  - Spark
    - Подключение к Spark
      - spark3-submit и spark3-shell
    - Примеры задач
      - Пакетная обработка в памяти
        - Dataset
        - RDD
        - DataFrame
    - Интеграция
      - Spark и Hive
      - ADB Spark 3 Connector
        - Обзор
        - Требования к установке
        - Примеры использования
        - Опции
      - ADQM Spark 3 Connector
        - Обзор
        - Требования к установке
        - Пример использования
        - Опции
      - PySpark
      - Spark Connect
      - Работа с Livy
    - Планирование задач Spark
    - Логирование
    - Оптимизация производительности
    - Таблицы Iceberg
    - Перезапись непартиционированных таблиц Hive
    - Управление сервисом через ADCM
  - SSM
    - Обзор UI
    - Определение правил
    - Определение действий
    - Примеры использования правил
    - Управление сервисом через ADCM
  - Trino
    - Подключение к Trino
      - CLI
      - JDBC
    - Обзор UI
    - Управление каталогами
      - Использование DBCatalogManager
    - Manage fault-tolerant execution
    - Передача учетных данных
    - Высокая доступность Trino Coordinator
    - Интеграции
      - Обзор коннекторов в Trino
      - Каталог Iceberg
      - Каталог Hive
      - Trino ADB connector
        - Обзор
        - Требования
        - Пример использования
        - Настройки
      - Trino OpenAPI connector
        - Обзор
        - Пример использования
        - Настройки
    - Безопасность
      - Аутентификация LDAP
      - Аутентификация Kerberos
      - Авторизация в Ranger с политиками Hive
    - Управление сервисом через ADCM
  - YARN
    - Обзор UI
    - Подключение к YARN
      - CLI
      - REST API
    - Планировщики
      - FairScheduler
      - CapacityScheduler
    - Режим высокой доступности
      - Обзор
      - Примеры
    - Логирование
    - Использование GPU
    - Управление сервисом через ADCM
  - Zeppelin
    - Обзор UI
    - Работа с интерпретаторами
      - Добавление интерпретатора в группу
      - Установка стороннего интерпретатора
      - User-managed интерпретаторы
      - Создание нового интерпретатора
    - Примеры заметок
      - Phoenix
      - Solr
      - Spark
      - Python
    - Логирование
    - Оптимизация производительности Zeppelin
    - Управление сервисом через ADCM
  - ZooKeeper
    - Подключение к ZooKeeper
      - CLI
      - API
    - Работа с znodes
    - Логирование
    - Настройка
    - Управление сервисом через ADCM

3. Руководства
  - Управление кластером через ADCM
    - Действия с кластером
      - Check
      - Manage SSL
      - Manage Credential Encryption
      - Manage Kerberos
      - Prepare hosts
      - Reinstall status-checker
      - Start
      - Stop
      - Обновление
    - Действия с хостами
      - Добавление хоста в кластер
      - Проверка состояния хоста
      - Удаление хоста из кластера
    - Действия с сервисами
  - Безопасность
    - ADPS
      - Обзор
      - Плагины Ranger
        - Плагин HBase
        - Плагин HDFS
        - Плагин Hive
        - Плагин Impala
        - Плагин Ozone
        - Плагин Solr
        - Плагин Spark3
        - Плагин Trino
        - Плагин YARN
    - Kerberos
      - Обзор
      - MIT Kerberos
        - Настройки сервера Kerberos
        - Настройка с помощью ADCM
      - MS Active Directory
        - Установка службы сертификации AD
        - Создание шаблона сертификата
        - Шаблон сертификата о выдаче
        - Запрос нового сертификата
        - Экспорт сертификата ЦС
        - Настройка с помощью ADCM
        - Керберизация с предустановленной Active Directory при отсутствии прав администратора
        - Частые проблемы AD Kerberos
      - FreeIPA
        - Обзор
        - Настройка с помощью ADCM
        - Керберизация с предустановленным FreeIPA при отсутствии прав администратора
      - Samba
        - Настройка с помощью ADCM
      - Подключение к Hive из DBeaver с помощью Kerberos
      - SPNEGO
    - SSL
      - Требования к SSL-сертификатам
      - Установка версии TLS
    - Настройка имперсонации в ADH

4. Справочные материалы
  - Командная строка Hadoop
    - Hadoop CLI
      - Пользовательские команды
        - archive
        - checknative
        - CLASSNAME
        - classpath
        - conftest
        - credential
        - distch
        - distcp
        - dtutil
        - envvars
        - fs
        - gridmix
        - jar
        - jnipath
        - kerbname
        - kdiag
        - key
        - kms
        - trace
        - version
      - Административные команды
        - daemonlog
    - HDFS CLI
      - Пользовательские команды
        - classpath
        - dfs
        - envvars
        - fetchdt
        - fsck
        - getconf
        - groups
        - httpfs
        - lsSnapshottableDir
        - jmxget
        - oev
        - oiv
        - oiv_legacy
        - snapshotDiff
        - version
      - Административные команды
        - balancer
        - cacheadmin
        - crypto
        - datanode
        - dfsadmin
        - dfsrouter
        - dfsrouteradmin
        - diskbalancer
        - ec
        - haadmin
        - journalnode
        - mover
        - namenode
        - nfs3
        - portmap
        - secondarynamenode
        - storagepolicies
        - zkfc
      - Команды отладки
        - computeMeta
        - recoverLease
        - verifyMeta
    - FileSystem shell
      - appendToFile
      - cat
      - checksum
      - chgrp
      - chmod
      - chown
      - copyFromLocal
      - copyToLocal
      - count
      - cp
      - createSnapshot
      - deleteSnapshot
      - df
      - du
      - dus
      - expunge
      - find
      - get
      - getfacl
      - getfattr
      - getmerge
      - head
      - help
      - ls
      - lsr
      - mkdir
      - moveFromLocal
      - moveToLocal
      - mv
      - put
      - renameSnapshot
      - rm
      - rmdir
      - rmr
      - setfacl
      - setfattr
      - setrep
      - stat
      - tail
      - test
      - text
      - touch
      - touchz
      - truncate
      - usage
    - HDFS cheatsheet
  - HBase
    - Команды HBase shell
      - General
        - processlist
        - status
        - table_help
        - version
        - whoami
      - DDL
        - alter
        - alter_async
        - alter_status
        - create
        - describe
        - disable
        - disable_all
        - drop
        - drop_all
        - enable
        - enable_all
        - exists
        - get_table
        - is_enabled
        - is_disabled
        - list
        - list_regions
        - locate_region
        - show_filters
      - Namespace
        - alter_namespace
        - create_namespace
        - describe_namespace
        - drop_namespace
        - list_namespace
        - list_namespace_tables
      - DML
        - append
        - count
        - delete
        - deleteall
        - get
        - get_counter
        - get_splits
        - incr
        - put
        - scan
        - truncate
        - truncate_preserve
      - Tools
        - assign
        - balancer
        - balancer_enabled
        - balance_switch
        - catalogjanitor_enabled
        - catalogjanitor_run
        - catalogjanitor_switch
        - cleaner_chore_enabled
        - cleaner_chore_run
        - cleaner_chore_switch
        - clear_block_cache
        - clear_compaction_queues
        - clear_deadservers
        - close_region
        - compact
        - compaction_state
        - compact_rs
        - flush
        - is_in_maintenance_mode
        - list_deadservers
        - major_compact
        - merge_region
        - move
        - normalize
        - normalizer_enabled
        - normalizer_switch
        - split
        - splitormerge_enabled
        - splitormerge_switch
        - trace
        - unassign
        - wal_roll
        - zk_dump
      - Replication
        - add_peer
        - append_peer_namespaces
        - append_peer_tableCFs
        - disable_peer
        - disable_table_replication
        - enable_peer
        - enable_table_replication
        - get_peer_config
        - list_peers
        - list_peer_configs
        - list_replicated_tables
        - remove_peer
        - remove_peer_namespaces
        - remove_peer_tableCFs
        - set_peer_bandwidth
        - set_peer_exclude_namespaces
        - set_peer_exclude_tableCFs
        - set_peer_namespaces
        - set_peer_replicate_all
        - set_peer_tableCFs
        - show_peer_tableCFs
        - update_peer_config
      - Snapshots
        - clone_snapshot
        - delete_all_snapshot
        - delete_snapshot
        - delete_table_snapshots
        - list_snapshots
        - list_table_snapshots
        - restore_snapshot
        - snapshot
      - Configuration
        - update_all_config
        - update_config
      - Quotas
        - list_quotas
        - list_snapshot_sizes
        - list_quota_snapshots
        - list_quota_table_sizes
        - set_quota
      - Security
        - grant
        - list_security_capabilities
        - revoke
        - user_permission
      - Procedures
        - abort_procedure
        - list_locks
        - list_procedures
      - Visibility labels
        - add_labels
        - clear_auths
        - get_auths
        - list_labels
        - set_auths
        - set_visibility
      - Rsgroup
        - add_rsgroup
        - balance_rsgroup
        - get_rsgroup
        - get_server_rsgroup
        - get_table_rsgroup
        - list_rsgroups
        - move_namespaces_rsgroup
        - move_servers_rsgroup
        - move_servers_namespaces_rsgroup
        - move_servers_tables_rsgroup
        - move_tables_rsgroup
        - remove_rsgroup
        - remove_servers_rsgroup
  - Hive
    - Параметры Hive on Spark
    - Командная строка Beeline
  - Impala
    - Команды impala-shell
  - MapReduce
    - MapReduce CLI
      - Пользовательские команды
        - archive
        - archive-logs
        - classpath
        - distcp
        - envvars
        - job
        - pipes
        - queue
        - version
      - Административные команды
        - frameworkuploader
        - historyserver
        - hsadmin
  - Ozone
    - Ozone CLI
      - bucket
        - addacl
        - clrquota
        - create
        - delete
        - getacl
        - info
        - link
        - list
        - removeacl
        - set-replication-config
        - setacl
        - setquota
        - update
      - key
        - addacl
        - cat
        - checksum
        - cp
        - delete
        - get
        - getacl
        - info
        - list
        - put
        - removeacl
        - rename
        - setacl
      - prefix
        - addacl
        - getacl
        - removeacl
        - setacl
      - snapshot
        - create
        - delete
        - diff
        - info
        - list
        - listDiff
      - token
        - cancel
        - get
        - print
        - renew
      - user
        - assign-admin
        - assign
        - get-secret
        - info
        - list
        - revoke-admin
        - revoke
        - set-secret
      - volume
        - addacl
        - clrquota
        - create
        - delete
        - getacl
        - info
        - list
        - removeacl
        - setacl
        - setquota
        - update
  - Spark
    - spark3-shell
    - spark3-submit
  - SSM
    - Действия
    - Правила
  - YARN
    - YARN CLI
      - Пользовательские команды
        - application
        - applicationattempt
        - classpath
        - container
        - envvars
        - jar
        - logs
        - node
        - queue
        - top
        - version
      - Административные команды
        - daemonlog
        - nodemanager
        - proxyserver
        - registrydns
        - resourcemanager
        - rmadmin
        - schedulerconf
        - scmadmin
        - sharedcachemanager
        - timelineserver
  - Конфигурационные параметры
  - Мониторинг
    - Метрики мониторинга ADH

5. Подготовка окружения
  - Требования к ПО
  - Требования к оборудованию
    - В зависимости от типа ноды
      - NameNode
      - DataNode
      - YARN
      - Ноды PostgreSQL, используемые с Hive Metastore
    - В зависимости от типа нагрузки
  - Требования к файловой системе
  - Требования к сети
    - Сетевые порты ADH
  - Типовые конфигурации
    - Пилотный кластер
  - Настройка пользовательской Java

6. Концепции
  - Архитектура и модель данных
    - Airflow
    - Flink
      - Обзор работы
    - HBase
      - Архитектура
      - Модель данных
    - HDFS
      - Архитектура
        - Масштабируемость
    - Hive
      - TEZ
    - Iceberg
      - Архитектура
    - Impala
      - Архитектура
    - Kyuubi
      - Архитектура
      - Сравнение Kyuubi, HiveServer, Spark Thrift Server
      - Share levels
    - MapReduce
      - Архитектура
      - Обзор работы
    - Monitoring
    - Ozone
      - Архитектура
      - Сравнение HDFS и Ozone
    - Phoenix
      - Обзор
    - Solr
      - Обзор работы
    - Spark
      - Spark Connect
    - SSM
      - Архитектура
    - Trino
      - Архитектура
    - YARN
      - Архитектура
    - Zeppelin
    - ZooKeeper
  - Безопасность
    - Kerberos

7. Релизы
  - Релизы ADH
  - Поддерживаемые сервисы
  - Матрица совместимости версий
  - Известные проблемы

Установка Arenadata Hyperwave

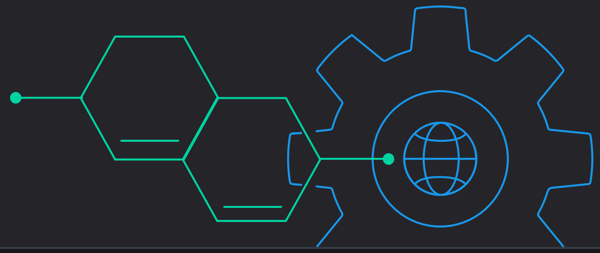
Назад к содержанию

Этот раздел содержит описание установки кластера Arenadata Hyperwave (ADH). Для установки рекомендуется использовать Arenadata Cluster Manager (ADCM) — отдельный продукт Arenadata, предназначенный для простого, удобного и быстрого развертывания и дальнейшей эксплуатации программного обеспечения.

Возможны два способа установки ADH через ADCM.

Online-установка

на хостах с доступом к Интернету



Offline-установка

на хостах без доступа к Интернету

Online-установка

Назад к содержанию

Этот раздел посвящен **online**-установке ADH. Данный способ установки подходит для хостов с доступом к Интернету.

Процесс online-установки через ADCM включает следующие шаги.

Шаг 1.

Установка ADCM

Шаг 2.

Подготовка хостов

ARENADATA
HADOOP

Шаг 3.

Установка кластера ADH

Шаг 4.

Установка мониторинга

Установка ADCM

Дарья Барышева

Содержание

Назад к содержанию

- Шаг 1. Предварительные требования
  - Требования к software и hardware
  - Установка Docker
  - Отключение SELinux
  - Остановка firewalld
  - Настройка доступа к Docker CLI без прав root
- Шаг 2. Установка ADCM
- Шаг 3. Запуск ADCM
- Шаг 4. Проверка установки
  - Проверка статуса Docker-контейнера
  - Проверка доступности порта 8000
  - Проверка web-интерфейса ADCM
  - Установка URL ADCM
- Шаг 5. Обновление ADCM

Для развертывания ADCM требуется отдельный хост. Перед началом установки рекомендуем ознакомиться с документацией ADCM.

Этапы установки описаны ниже.

Шаг 1. Предварительные требования

Требования к software и hardware

ADCM поставляется в виде Docker-образа. Для его установки необходимы:

- Одна из приведенных ниже операционных систем:
  - Ubuntu 22.04
  - CentOS 7
  - RHEL 7
  - Alt Linux SP 8
  - Alt Linux SP 10.2
  - Astra Linux SE 1.7 Орел
  - Astra Linux 1.7.6 Воронеж
  - РЕД ОС 7.3 Сертифицированная редакция
- Пользователь с правами `sudo`.
- Доступ к официальным репозиториям CentOS Extras и CentOS Base. По умолчанию, эти репозитории доступны в операционной системе. Однако, если они были отключены, потребуется их предварительное включение (например, путем редактирования конфигурационного файла в директории */etc/yum.repos.d/*).
- Менеджер программных пакетов YUM/APT.
- Хост, отвечающий следующим требованиям:

ВАЖНО

Системные требования, приведенные ниже, являются **минимальными**. Целевой сайзинг необходимо рассчитывать исходя из требований вашей организации.

- **CPU**: 4 ядра CPU;
- **RAM**: 16 ГБ;
- **HDD**: 50 ГБ.

ВАЖНО

Начиная с версии ADCM 2.7.0 для установки ADPG без подключения к Интернету и использования его в качестве базы данных ADCM требуется любой Container Registry Client с загруженными образами ADPG, ADCM и ADCM Installer. Установка выполняется с помощью ADCM Installer с указанием URL локального хранилища образов ADCM и ADPG.

Чтобы пользователь обладал необходимыми правами для установки ADCM, выполните следующее:

1. Предоставьте пользователю привилегии `root`:

CentOS 7/RHEL 7/РЕД ОС 7.3/Alt Linux SP 8

```
$ sudo usermod -aG wheel <user>
```

где `<user>` — имя пользователя.

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
$ sudo usermod -aG sudo <user>
```

где `<user>` — имя пользователя.

2. Убедитесь, что в файле */etc/sudoers* есть запись, указанная ниже в зависимости от используемой операционной системы, с помощью следующей команды:

```
$ sudo vi /etc/sudoers
```

Чтобы вернуться в терминал, нажмите `Esc`, введите `:q!` и нажмите `Enter`.

CentOS 7/RHEL 7/РЕД ОС 7.3

```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
```

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Alt Linux SP 8

```
## Uncomment to allow members of group wheel to execute any command
WHEEL_USERS ALL=(ALL:ALL) ALL
```

Установка Docker

ВНИМАНИЕ

Установка Docker с официального сайта не рекомендуется. Вместо этого используйте официальные репозитории дистрибутива Linux. Репозитории, предоставляемые разработчиками операционных систем, как правило, более стабильны и безопасны.

Ниже приведены шаги, необходимые для установки Docker, на примере нескольких поддерживаемых в ADCM операционных систем. Для других версий операционных систем, отличных от указанных во вкладках, шаги могут отличаться.

CentOS 7/RHEL 7

1. Обновите все программные пакеты, установленные в системе:

```
$ sudo yum update -y
```

2. Установите пакеты, необходимые для Docker:

```
$ sudo yum install -y yum-utils docker device-mapper-persistent-data lvm2
```

где:

- `yum-utils` — пакет, который предоставляет утилиту `yum-config-manager`;
- `device-mapper-persistent-data` и `lvm2` — пакеты, необходимые драйверу хранилища devicemapper.

3. Запустите Docker:

```
$ sudo systemctl start docker
```

4. Включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable docker
```

Ubuntu 22.04

1. Обновите все программные пакеты, установленные в системе:

```
$ sudo apt-get update
```

2. Установите пакеты, необходимые для Docker:

```
$ sudo apt-get install ca-certificates curl
```

где:

- `ca-certificates` — пакет корневых сертификатов;
- `curl` — пакет для работы с веб-ресурсами.

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

3. Добавьте GPG-ключ для Docker-репозитория:

```
$ sudo install -m 0755 -d /etc/apt/keyrings
$ sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
$ sudo chmod a+r /etc/apt/keyrings/docker.asc
```

4. Добавьте Docker-репозиторий:

```
$ echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \
$(. /etc/os-release && echo "jammy") stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

5. Обновите все программные пакеты, в том числе пакеты для добавленного Docker-репозитория:

```
$ sudo apt-get update
```

6. Установите Docker Engine:

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

7. Запустите Docker:

```
$ sudo systemctl start docker
```

8. Включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable docker
```

Alt Linux SP 8

1. Обновите все программные пакеты, установленные в системе:

```
$ sudo apt-get update
```

2. Установите Docker Engine:

```
$ sudo apt-get install docker-engine
```

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

3. Запустите и включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable --now docker
```

Astra Linux SE 1.7 Орел

1. Обновите все программные пакеты, установленные в системе:

```
$ sudo apt-get update
```

2. Установите пакеты, необходимые для Docker:

```
$ sudo apt-get install ca-certificates curl net-tools
```

где:

- `ca-certificates` — пакет корневых сертификатов;
- `curl` — пакет для работы с веб-ресурсами;
- `net-tools` — пакет для управления сетевой подсистемой ядра Linux, в частности для запуска команды `netstat`.

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

3. Добавьте GPG-ключ для Docker-репозитория:

```
$ curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
```

4. Добавьте Docker-репозиторий:

```
$ echo "deb [arch=amd64] https://download.docker.com/linux/debian stretch stable" | sudo tee -a /etc/apt/sources.list
```

5. Обновите все программные пакеты, в том числе пакеты для добавленного Docker-репозитория:

```
$ sudo apt-get update
```

6. Установите Docker Engine:

```
$ sudo apt install docker-ce docker-ce-cli containerd.io
```

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

7. Запустите Docker:

```
$ sudo systemctl start docker
```

8. Включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable docker
```

РЕД ОС 7.3

1. Установите Docker Engine:

```
$ sudo dnf install docker-ce docker-ce-cli
```

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

2. Запустите и включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable docker --now
```

Отключение SELinux

ПРИМЕЧАНИЕ

Отключение системы безопасности SELinux необходимо только в случае использования операционной системы CentOS 7 или RHEL 7.

Для долговременного отключения SELinux следует прописать `SELINUX=disabled` в конфигурационном файле */etc/selinux/config* и перезагрузить систему.

Файл может быть изменен с помощью редактора `vi` (или `vim`):

```
$ sudo vi /etc/selinux/config
```

Содержимое обновленного файла:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - SELinux is fully disabled.
SELINUX=disabled
# SELINUXTYPE= type of policy in use. Possible values are:
#       targeted - Only targeted network daemons are protected.
#       strict - Full SELinux protection.
SELINUXTYPE=targeted
```

Кроме этого, доступно временное отключение SELinux с помощью следующей команды:

```
$ sudo setenforce 0
```

Тем не менее рекомендуется отключить SELinux на постоянной основе, чтобы предотвратить ее перезапуск при каждой перезагрузке системы.

ВНИМАНИЕ

Не забудьте перезагрузить хост после отключения SELinux.

Остановка firewalld

Если на хосте используется сервис firewalld, требуется его остановка.

Перед остановкой firewalld убедитесь, что он установлен, запущен и включен, выполнив команду:

```
$ sudo systemctl status firewalld
```

Пример результата выполнения команды:

```
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2024-12-02 10:02:57 UTC; 3min 25s ago
     Docs: man:firewalld(1)
 Main PID: 3023 (firewalld)
    Tasks: 2
   Memory: 22.8M
   CGroup: /system.slice/firewalld.service
           └─3023 /usr/bin/python2 -Es /usr/sbin/firewalld --nofork --nopid
```

Остановите firewalld с помощью команды:

```
$ sudo systemctl stop firewalld
```

Альтернативный вариант — отключить применение правил к сетевым пакетам при помощи команды `disable`:

```
$ sudo systemctl disable firewalld
```

Настройка доступа к Docker CLI без прав root

Для настройки доступа к Docker CLI без использования прав `root` необходимо создать группу пользователей `docker` (если она еще не существует) и добавить текущего пользователя в эту группу с определенными правами:

```
$ sudo groupadd docker
$ sudo usermod -a -G docker $USER
```

После запуска команд требуется перезайти в систему под текущим пользователем.

Шаг 2. Установка ADCM

ПРИМЕЧАНИЕ

- Рекомендуется самостоятельно настроить PostgreSQL в качестве внешней базы данных или развернуть встроенную базу данных с помощью ADCM Installer. Дополнительную информацию о настройке PostgreSQL можно получить в статье Работа с внешней базой данных.
- Начиная с версии ADCM 2.5.0 минимальная требуемая версия PostgreSQL — 14.
- Для просмотра полного списка доступных образов можно обратиться к репозиторию ADCM на Docker Hub.
- Все данные ADCM сохраняются в папке */adcm/data* внутри Docker-контейнера. Во избежание потерь данных рекомендуется примонтировать к контейнеру том (Docker Volume) с последующей настройкой его бэкапов.

Для установки ADCM выполните следующие действия:

1. Скачайте Docker-образ из Arenadata Docker Registry:

```
$ sudo docker pull hub.arenadata.io/adcm/adcm:<version>
```

где `<version>` — это версия Docker-образа в одном из следующих форматов:

- `<major>.<minor>.<patch>` — если необходим конкретный патч ADCM. Пример: `2.0.0`.
- `<major>.<minor>` — если необходим последний патч в рамках выбранной версии ADCM. Пример: `2.0`.

ВАЖНО

- `latest` использовался для ранних версий ADCM и начиная с версии `2.0.0` больше не поддерживается.
- До выпуска ADCM 2.0.0 для версионирования использовался следующий формат: `YYYY.MM.DD.HH`.

2. Создайте Docker-контейнер на основе загруженного образа:

ПРИМЕЧАНИЕ

Приведенная ниже команда создает контейнер с ADCM на порте `8000` и устанавливает подключение к внешней базе данных. Все данные контейнера будут храниться в директории */opt/adcm/*. Описание переменных окружения приведено в таблице ниже.

```
$ sudo docker create --name adcm -p 8000:8000 -v /opt/adcm:/adcm/data -e DB_HOST="<DATABASE_HOSTNAME_OR_IP_ADDRESS>" -e DB_PORT="
<DATABASE_TCP_PORT>" -e DB_USER="<DATABASE_USERNAME>" -e DB_NAME="<DATABASE_NAME>" -e DB_PASS="<DATABASE_USER_PASSWORD>" -e
DB_OPTIONS="<DATABASE_OPTIONS>" hub.arenadata.io/adcm/adcm:<version>
```

Если требуется запустить ADCM **с включенным SELinux**, при создании Docker-контейнера необходимо указать опцию `:Z` в параметре, содержащем имя каталога данных:

```
$ sudo docker create --name adcm -p 8000:8000 -v /opt/adcm:/adcm/data:Z hub.arenadata.io/adcm/adcm:<version>
```

Опционально при создании контейнера можно указать переменные окружения, перечисленные ниже. Каждая переменная указывается в формате `-e <name>="<value>"`. Пример:

```
$ sudo docker create --name adcm -p 8000:8000 -v /opt/adcm:/adcm/data -e LOG_LEVEL="INFO" hub.arenadata.io/adcm/adcm:<version>
```

ПРИМЕЧАНИЕ

Специальные переменные окружения, такие как `STATUS_LOG_LEVEL`, `ADCM_LOG_LEVEL`, `AUDIT_LOG_LEVEL`, `LDAP_LOG_LEVEL`, `BACKGROUND_TASKS_LOG_LEVEL` и `TASK_RUNNER_LOG_LEVEL`, являются более приоритетными, чем переменная окружения `LOG_LEVEL`.

Переменные окружения

Имя	Описание
DB_HOST	Имя или IP-адрес хоста, на котором развернута база данных
DB_PORT	Номер TCP-порта для подключения к базе данных
DB_USER	Имя пользователя для подключения к базе данных
DB_NAME	Имя базы данных
DB_PASS	Пароль пользователя для подключения к базе данных
DB_OPTIONS	Параметры для подключения к базе данных: `sslmode` — флаг, используемый для установления SSL-соединения. Возможные значения: `disable` — установить только соединение без использования SSL. `allow` — попытаться установить соединение без использования SSL. Если не удастся — попытаться установить SSL-соединение. `prefer` (по умолчанию) — попытаться установить SSL-соединение. Если не удастся — попытаться установить соединение без использования SSL. `require` — попытаться установить только SSL-соединение. Если присутствует файл корневого сертификата удостоверяющего центра (Certificate Authority, CA) — проверить сертификат так же, как если бы было указано `verify-ca`. `verify-ca` — попытаться установить только SSL-соединение и проверить, что сертификат сервера выдан доверенным удостоверяющим центром. `verify-full` — попытаться установить только SSL-соединение и проверить, что сертификат сервера выдан доверенным удостоверяющим центром и запрашиваемое имя хоста сервера совпадает с указанным в сертификате. `sslcert` — расположение файла SSL-сертификата клиента. Этот параметр игнорируется, если соединение устанавливается без SSL. `sslkey` — расположение секретного ключа, используемого для клиентского сертификата. Этот параметр игнорируется, если соединение устанавливается без SSL. `sslrootcert` — расположение файла, содержащего сертификат(ы) центра сертификации SSL. Если файл существует, сертификат сервера будет проверен одним из этих центров сертификации. Вы также можете указать специальное значение системы — в этом случае будут загружены доверенные корневые сертификаты системы. Точные местоположения этих корневых сертификатов зависят от реализации SSL и платформы.
DEFAULT_ADCM_URL	URL хоста, на котором разворачивается контейнер ADCM. Указывается в формате: `http://<IP>:8000/`. Если переменная не указана при создании контейнера, URL необходимо будет заполнить вручную в поле ***ADCM's URL*** на странице ***Settings***

LOG_LEVEL	Уровень логирования для всех типов логов. Возможные значения:
	`DEBUG`
	`INFO`
	`WARNING`
	`ERROR` (значение по умолчанию)
	`CRITICAL`

STATUS_LOG_LEVEL	Уровень логирования для *status.log*. Возможные значения:
	`DEBUG`
	`INFO`
	`WARNING`
	`ERROR`
	`CRITICAL`

ADCM_LOG_LEVEL	Уровень логирования для *adcm.log* и *adcm_debug.log*. Возможные значения:
	`DEBUG`
	`INFO`
	`WARNING`
	`ERROR`
	`CRITICAL`

AUDIT_LOG_LEVEL	Уровень логирования для *audit.log*. Возможные значения:
	`DEBUG`
	`INFO` (значение по умолчанию)
	`WARNING`
	`ERROR`
	`CRITICAL`

LDAP_LOG_LEVEL	Уровень логирования для *ldap.log*. Возможные значения:
	`DEBUG`
	`INFO`
	`WARNING`
	`ERROR`
	`CRITICAL`

BACKGROUND_TASKS_LOG_LEVEL	Уровень логирования для *cron_task.log*. Возможные значения:
	`DEBUG`
	`INFO`
	`WARNING`
	`ERROR`
	`CRITICAL`

TASK_RUNNER_LOG_LEVEL	Уровень логирования для *task_runner.err*. Возможные значения:
	`DEBUG`
	`INFO`
	`WARNING`
	`ERROR`
	`CRITICAL`

MIGRATION_MODE	Флаг, указывающий на использование режима миграции — переноса базы данных ADCM с SQLite на PostgreSQL. Дополнительную информацию можно получить в статье Миграция. Возможные значения: `0` — режим миграции выключен; `1` — режим миграции включен.
CONSUL_URL	URL сервера HashiCorp Consul. Указывается в формате: `http://<IP>:8500/`
CONSUL_DATACENTER	Datacenter HashiCorp Consul. Если значение переменной не указано, то будет использовано значение, указанное в HashiCorp Consul по умолчанию
CONSUL_CACERT_FILE	Путь до файла сертификата удостоверяющего центра (Certificate Authority, CA) в случае включенного шифрования в HashiCorp Consul
CONSUL_CLIENT_CERT_FILE	Путь до файла сертификата клиента в случае включенного шифрования в HashiCorp Consul
CONSUL_CLIENT_KEY_FILE	Путь до файла с секретным ключом клиента в случае включенного шифрования в HashiCorp Consul

Шаг 3. Запуск ADCM

Поскольку ADCM представляет собой контейнер под управлением Docker, за его запуск и установку отвечают стандартные Docker-команды:

- **Запуск ADCM**.

```
$ sudo docker start adcm
```

- **Остановка ADCM**.

```
$ sudo docker stop adcm
```

- **Автозапуск ADCM**. Чтобы настроить автоматический запуск Docker-контейнера в случае непредвиденных ошибок, используйте следующую команду.

```
$ sudo docker update --restart=on-failure adcm
```

ПРИМЕЧАНИЕ

- Дополнительная информация по политикам перезапуска Docker-контейнеров (restart policies) может быть получена из документации Docker.
- Если для текущего пользователя настроен доступ к Docker CLI без использования прав `root`, слово `sudo` в предыдущих командах необязательно.

Шаг 4. Проверка установки

Проверка статуса Docker-контейнера

Чтобы убедиться в успешной установке Docker-контейнера `adcm`, выполните следующую команду:

```
$ sudo docker container ls
```

Пример результата выполнения команды приведен ниже. Убедитесь, что в столбце `STATUS` выводится значение `Up`:

```
CONTAINER ID        IMAGE                             COMMAND             CREATED             STATUS              PORTS
NAMES
74b5628146e5        hub.arenadata.io/adcm/adcm:2.0    "/etc/startup.sh"   4 days ago          Up 21 minutes       0.0.0.0:8000-
>8000/tcp   adcm
```

Проверка доступности порта 8000

После установки и запуска ADCM его web-интерфейс должен стать доступным на `8000` порте развернутого контейнера.

Для операционной системы Ubuntu 22.04 предварительно установите пакет для запуска команды `netstat`:

```
$ sudo apt-get install net-tools
```

Для проверки доступности порта выполните команды:

- `netstat` — для вывода информации о состоянии порта.

```
$ sudo netstat -ntpl | grep 8000
```

Пример результата выполнения команды:

```
tcp6       0      0 :::8000                 :::*                    LISTEN
1514/docker-proxy-c
```

- `curl` — для проверки URL-соединения.

```
$ curl http://localhost:8000
```

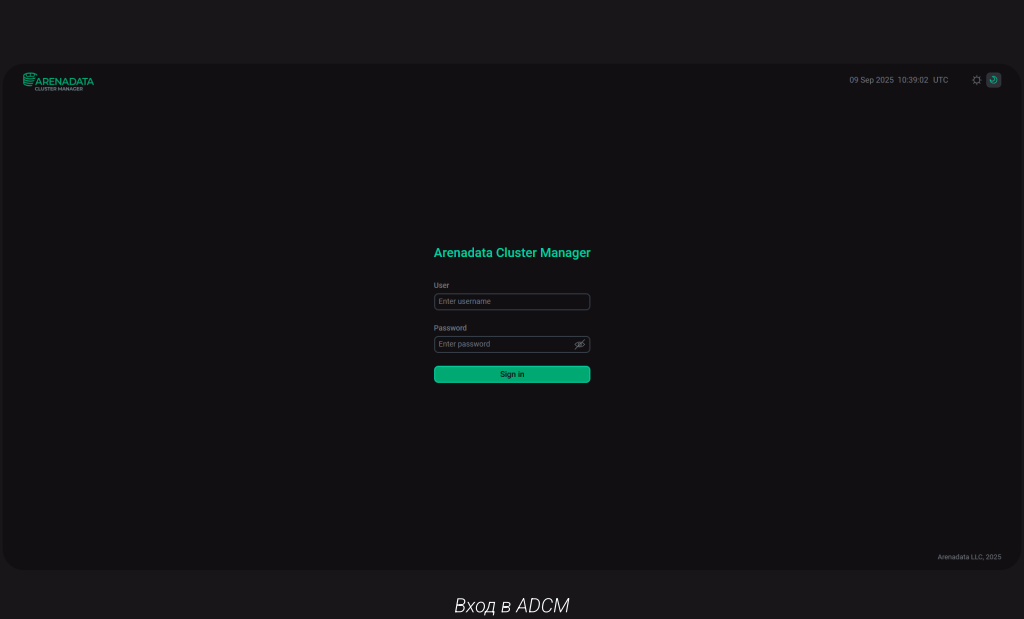
Пример результата выполнения команды:

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Arenadata Cluster Manager</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="assets/favicon.ico">
<link rel="stylesheet" href="styles.10db6328264e0907c52f.css"></head>
<body>
  <app-root></app-root>
<script src="runtime-es2015.7eb406ed18bf0258cd35.js" type="module"></script><script src="runtime-es5.7eb406ed18bf0258cd35.js"
nomodule defer></script><script src="polyfills-es5.2e224d70daec4412d3c2.js" nomodule defer></script><script src="polyfills-es2015.
37e0553ac06970d6a5b5.js" type="module"></script><script src="main-es2015.39851da0ebf9ed6fec45.js" type="module"></script><script
src="main-es5.39851da0ebf9ed6fec45.js" nomodule defer></script></body>
</html>
```

Проверка web-интерфейса ADCM

При установке ADCM автоматически создается пользователь с административными правами. Чтобы зайти в ADCM, используйте следующие учетные данные:

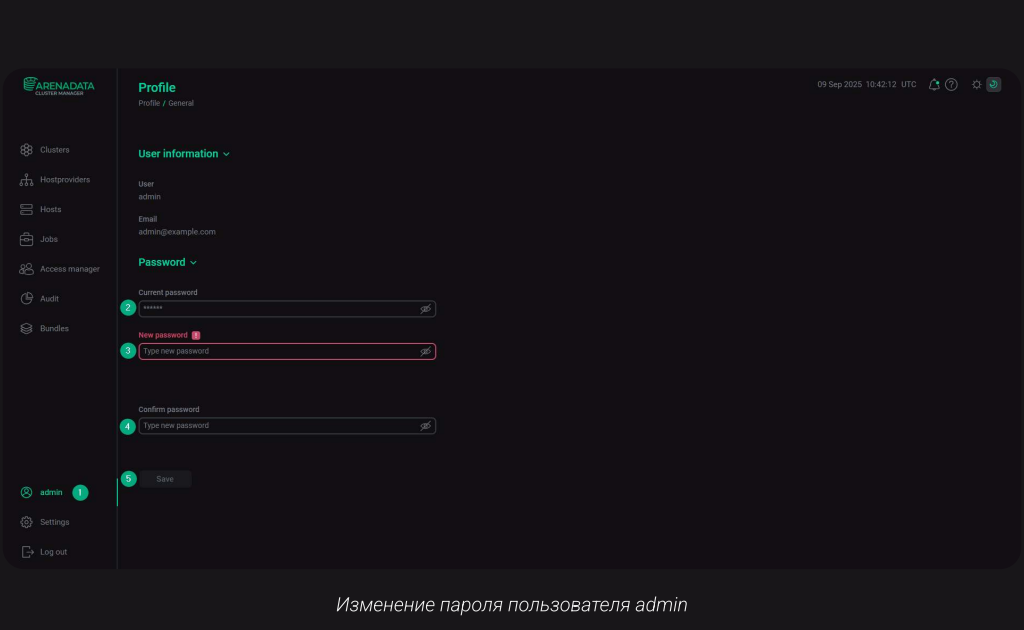
- ***User***: admin
- ***Password***: admin



Вход в ADCM

Для изменения пароля требуется выполнить следующие действия:

1. Откройте профиль текущего пользователя в левом навигационном меню.
2. Введите текущий пароль в поле ***Current password***.
3. Укажите новый пароль в поле ***New password***.
4. Подтвердите новый пароль в поле ***Confirm password***.
5. Нажмите ***Save***.

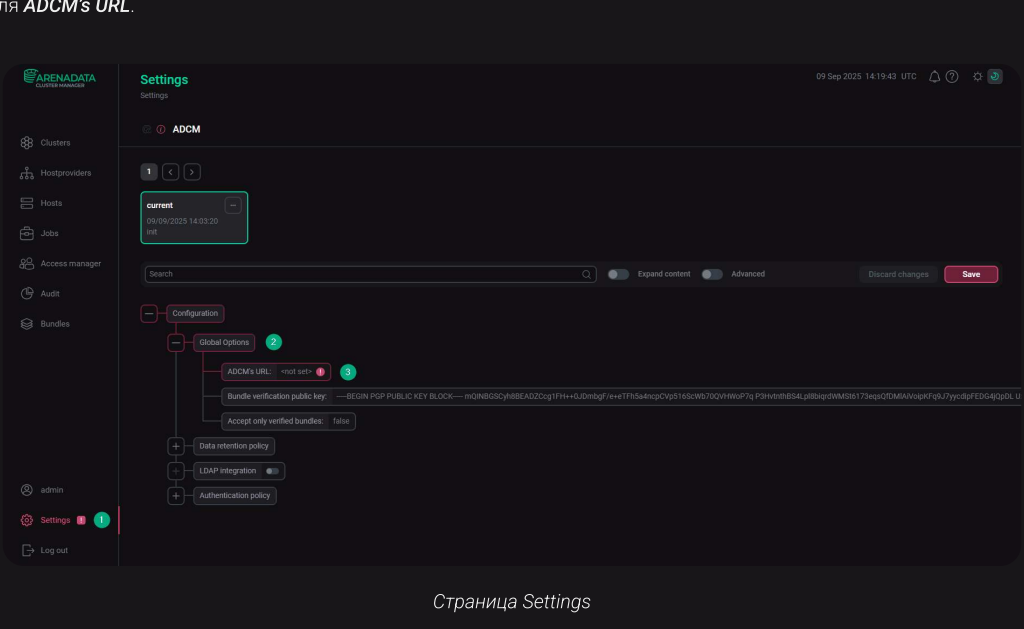


Изменение пароля пользователя admin

Установка URL ADCM

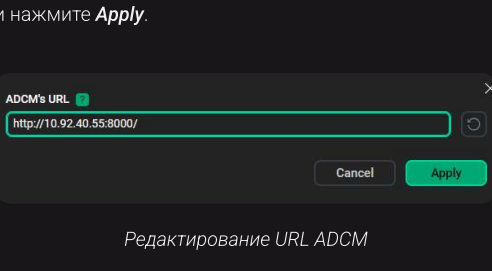
Для успешного обмена данными между ADCM и развернутыми с его помощью кластерами важно корректное определение внешнего адреса ADCM. В дальнейшем этот адрес будет использоваться для отправки в ADCM статусов кластерных компонентов. Для указания URL ADCM используется переменная окружения `DEFAULT_ADCM_URL` при создании контейнера ADCM. Если переменная не была указана, необходимо установить URL вручную в web-интерфейсе ADCM. Для этого выполните шаги:

1. Выберите пункт ***Settings*** в левом навигационном меню.
2. Раскройте узел ***Global Options*** в дереве конфигурационных настроек.
3. Нажмите на значение поля ***ADCM's URL***.



Страница Settings

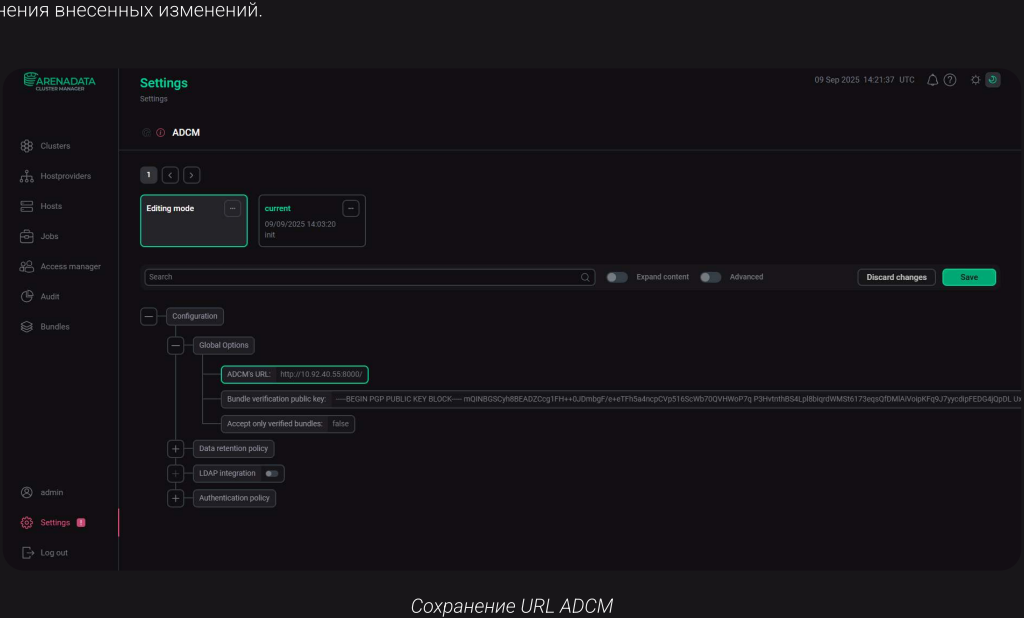
4. В открывшемся окне измените значение ***ADCM's URL*** и нажмите ***Apply***.



http://10.92.40.55:8000/

Редактирование URL ADCM

5. Нажмите ***Save*** для сохранения внесенных изменений.



Сохранение URL ADCM

Шаг 5. Обновление ADCM

ВНИМАНИЕ

- Перед выполнением обновления рекомендуется сделать резервную копию данных ADCM. Дополнительную информацию можно получить в статье Резервное копирование и восстановление.
- Понижение версии ADCM не поддерживается.
- Не рекомендуется обновлять ADCM более, чем на 5 минорных версий (см. `<minor>` в описании формата версионирования ниже).
- Перед обновлением ADCM до версии 2.7.0 выполните миграцию данных из SQLite в PostgreSQL. Миграцию можно осуществить в ADCM любой версии (до 2.6.0 включительно).

Для обновления ADCM до нужной версии выполните следующие шаги:

1. Остановите Docker-контейнер `adcm`:

```
$ sudo docker stop adcm
```

2. Удалите Docker-контейнер `adcm`:

```
$ sudo docker container rm adcm
```

3. Загрузите из Arenadata Docker Registry необходимый Docker-образ:

```
$ sudo docker pull hub.arenadata.io/adcm/adcm:<version>
```

где `<version>` — это версия Docker-образа в одном из следующих форматов:

- `<major>.<minor>.<patch>` — если необходим конкретный патч ADCM. Пример: `2.0.0`.
- `<major>.<minor>` — если необходим последний патч в рамках выбранной версии ADCM. Пример: `2.0`.

ВАЖНО

- `latest` использовался для ранних версий ADCM и начиная с версии `2.0.0` больше не поддерживается.
- До выпуска ADCM 2.0.0 для версионирования использовался следующий формат: `YYYY.MM.DD.HH`.

4. Создайте новый Docker-контейнер на базе загруженного образа:

```
$ sudo docker create --name adcm -p 8000:8000 -v /opt/adcm:/adcm/data -e DB_HOST="<DATABASE_HOSTNAME_OR_IP_ADDRESS>" -e DB_PORT="
<DATABASE_TCP_PORT>" -e DB_USER="<DATABASE_USERNAME>" -e DB_NAME="<DATABASE_NAME>" -e DB_PASS="<DATABASE_USER_PASSWORD>" -e
DB_OPTIONS="<DATABASE_OPTIONS>" hub.arenadata.io/adcm/adcm:<version>
```

ПРИМЕЧАНИЕ

Подробная информация о переменных окружения `DB_HOST`, `DB_PORT`, `DB_USER`, `DB_NAME`, `DB_PASS` и `DB_OPTIONS` приведена в статье Установка.

5. Запустите новый Docker-контейнер `adcm`:

```
$ sudo docker start adcm
```

Подготовка хостов

Содержание

Назад к содержанию

Права пользователей для установки ADH

Перед установкой нового кластера необходимо добавить информацию о его будущих хостах в ADCM. Это задача **хостпровайдеров** — специальных плагинов, отвечающих за взаимодействие между хостами и ADCM.

ADCM поддерживает несколько типов хостпровайдеров. Для получения подробной информации о том, как их настроить и использовать для создания хостов, обратитесь к документации:

- SSH
- Yandex Cloud
- VCloud (VMware)
- Cloud.ru Advanced
- CROC Cloud

ВНИМАНИЕ

При создании хостов ADH в ADCM требуется заполнять их полные доменные имена (FQDN). Максимальная длина FQDN — 38 символов.

Права пользователей для установки ADH

Если в настройках хостпровайдера указан кастомный UNIX-пользователь, от имени которого будут выполняться команды установки ADH, убедитесь, что данный пользователь может выполнять команды с правами суперпользователя.

Чтобы пользователь обладал необходимыми правами для установки ADH, выполните следующее:

1. Предоставьте пользователю привилегии `root`:

CentOS 7/RHEL 7/РЕД ОС 7.3/Alt Linux SP 8

```
$ sudo usermod -aG wheel <user>
```

где `<user>` — имя пользователя.

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
$ sudo usermod -aG sudo <user>
```

где `<user>` — имя пользователя.

2. Убедитесь, что в файле */etc/sudoers* есть запись, указанная ниже в зависимости от используемой операционной системы, с помощью следующей команды:

```
$ sudo vi /etc/sudoers
```

Чтобы вернуться в терминал, нажмите `Esc`, введите `:q!` и нажмите `Enter`.

CentOS 7/RHEL 7/РЕД ОС 7.3

```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
```

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Alt Linux SP 8

```
## Uncomment to allow members of group wheel to execute any command
WHEEL_USERS ALL=(ALL:ALL) ALL
```

Создание кластера

Евгения Кузина, Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Загрузка бандла

Шаг 2. Загрузка бандла в ADCM

Шаг 3. Создание кластера на базе загруженного бандла

В ADCM **кластер** означает набор сервисов, запущенных на группе хостов.

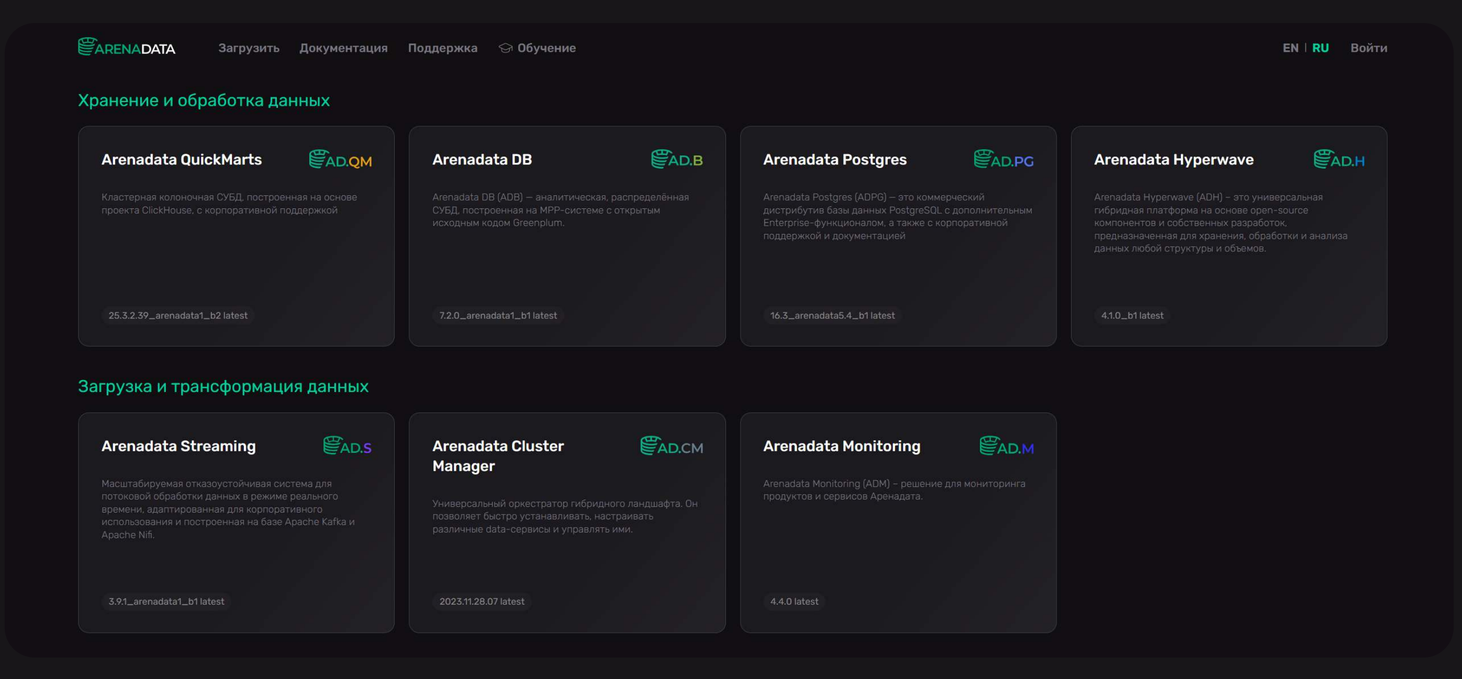
Дистрибутивы кластеров для ADCM поставляются в виде бандлов. В общем случае бандл представляет обычный архив, описывающий логику развертывания кластера.

Шаги для добавления нового кластера приведены ниже.

> ПРИМЕЧАНИЕ
>
> При использовании Enterprise-версии продукта следует пропустить шаг 1, получить бандл у команды поддержки Arenadata и перейти к шагу 2.

Шаг 1. Загрузка бандла

1. Перейдите на сайт https://network.arenadata.io/ и выберите продукт ***Arenadata Hyperwave***.



ARENADATA
Загрузить
Документация
Поддержка
Обучение
EN | RU
Войти
Хранение и обработка данных
Arenadata QuickMarts
Arenadata DB
Arenadata Postgres
Arenadata Hyperwave
Загрузка и трансформация данных
Arenadata Streaming
Arenadata Cluster Manager
Arenadata Monitoring

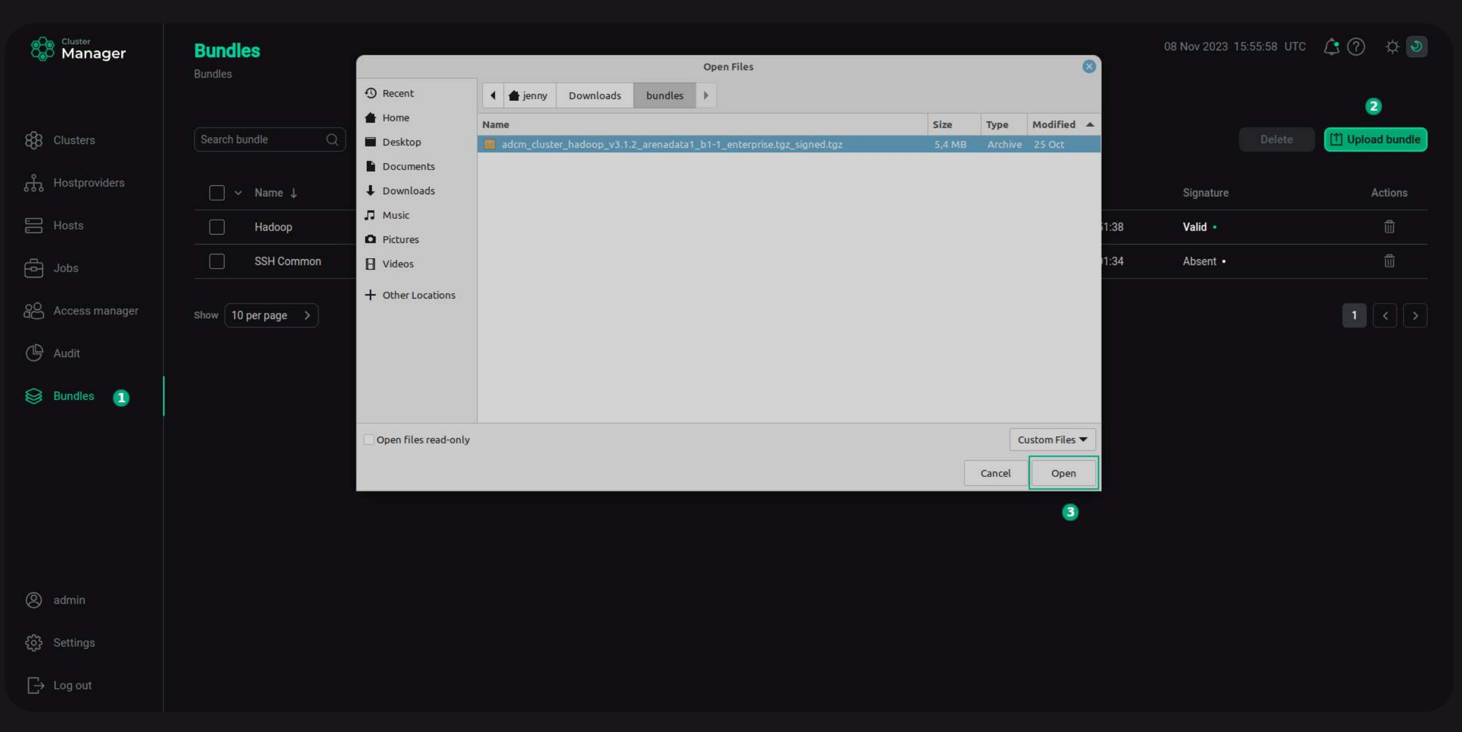
Переход к загрузке дистрибутивов

2. На следующей странице загрузите бандл для нужной версии ADH.

Шаг 2. Загрузка бандла в ADCM

Чтобы загрузить бандл в ADCM:

1. Выберите пункт ***Bundles*** в левом навигационном меню и нажмите ***Upload bundle***.
2. Выберите бандл в открывшемся диалоговом окне.

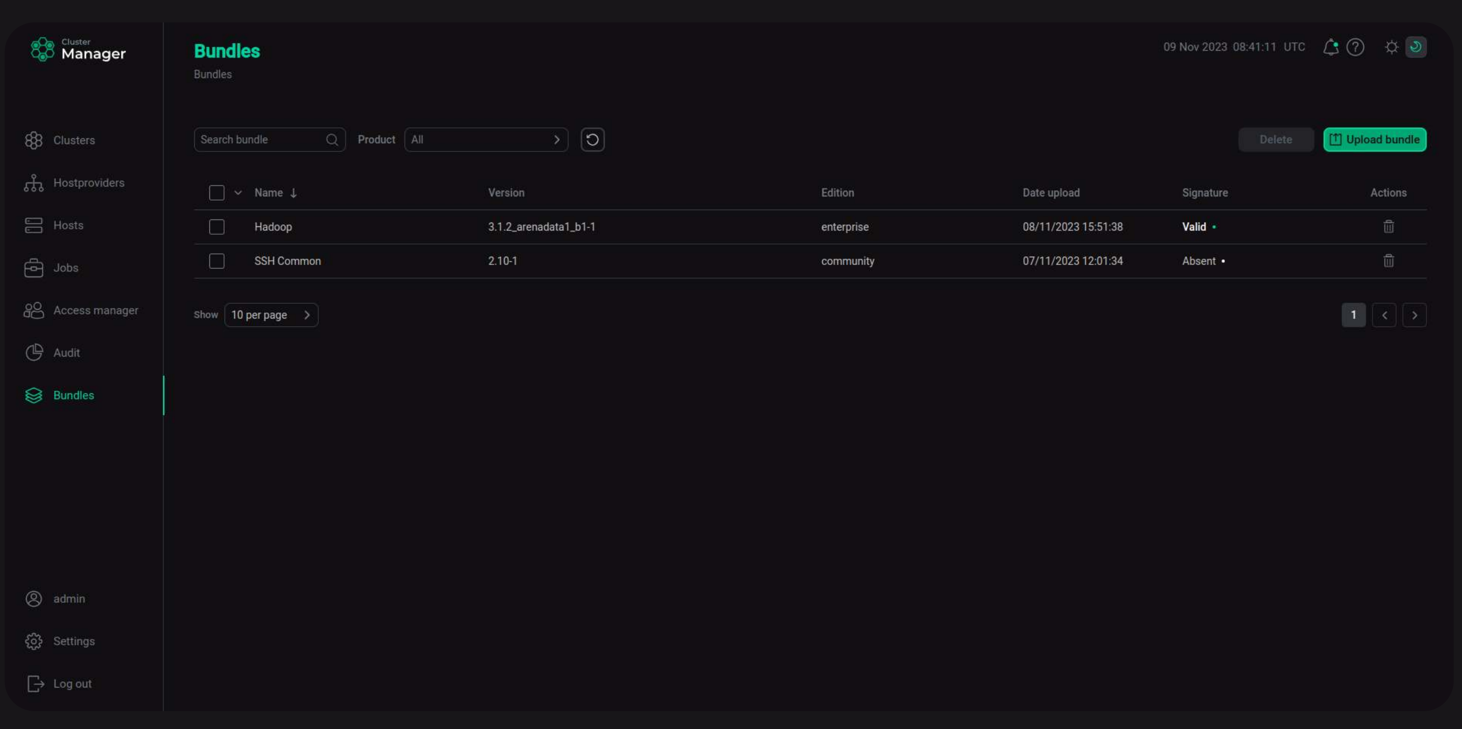


Manager
Bundles
Open Files
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Other Locations
jenny
Downloads
bundles
Delete
Upload bundle
Open files read-only
Custom Files
Cancel
Open

Name	Size	Type	Modified
adcm_cluster_hadoop_v3.1.2_arenadata1_b1-1_enterprise.tgz_signed.tgz	5,4 MB	Archive	25 Oct

Переход к загрузке бандла

3. После успешной загрузки бандл отображается на странице ***Bundles***.



Manager
Bundles
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Delete
Upload bundle

Name	Version	Edition	Date upload	Signature
Hadoop	3.1.2_arenadata1_b1-1	enterprise	08/11/2023 15:51:38	Valid
SSH Common	2.10-1	community	07/11/2023 12:01:34	Absent

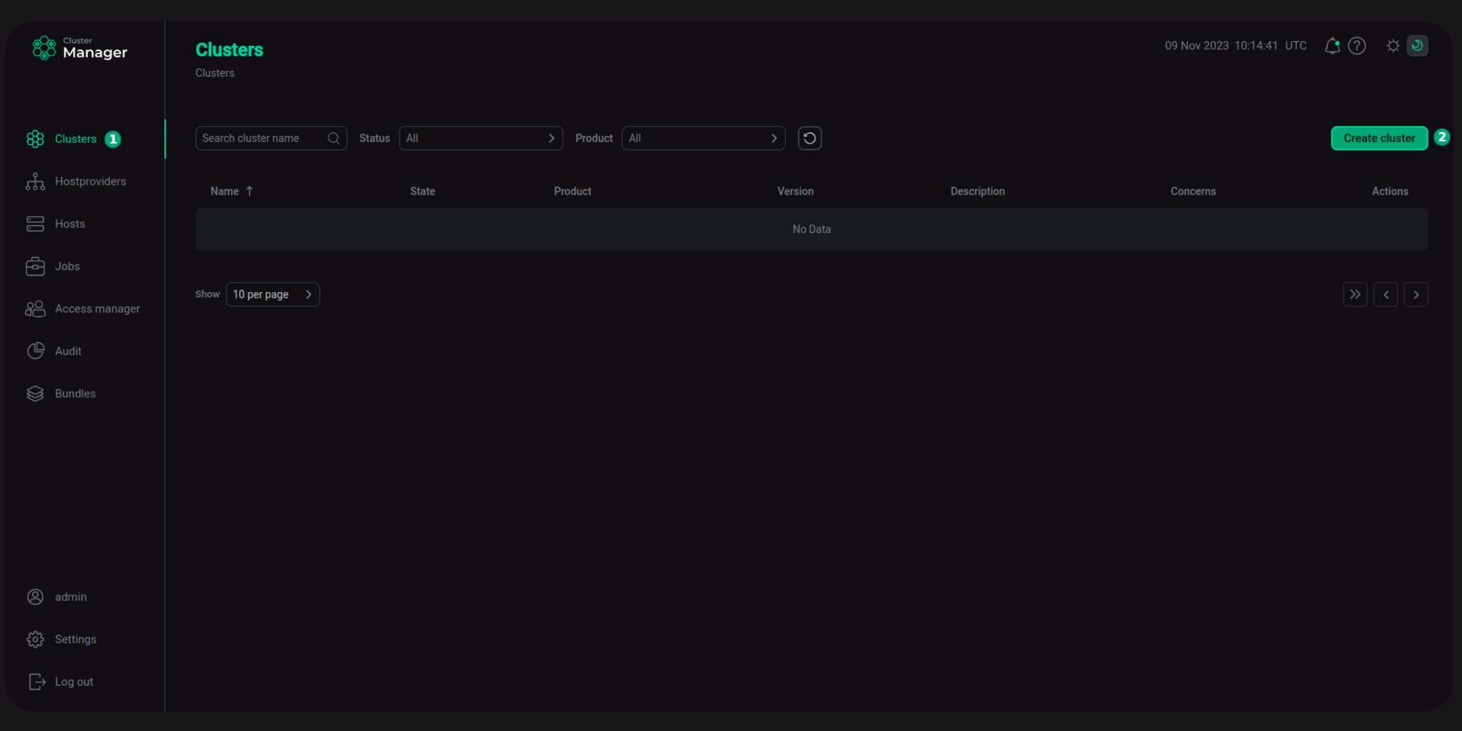
Результат успешной загрузки бандла

Шаг 3. Создание кластера на базе загруженного бандла

> ВАЖНО
>
> На этапе создания кластера через web-интерфейс ADCM генерируется только сам кластер. Это означает добавление записи о кластере в ADCM — сервисы на этом шаге еще не устанавливаются.

Для создания кластера необходимо выполнить шаги:

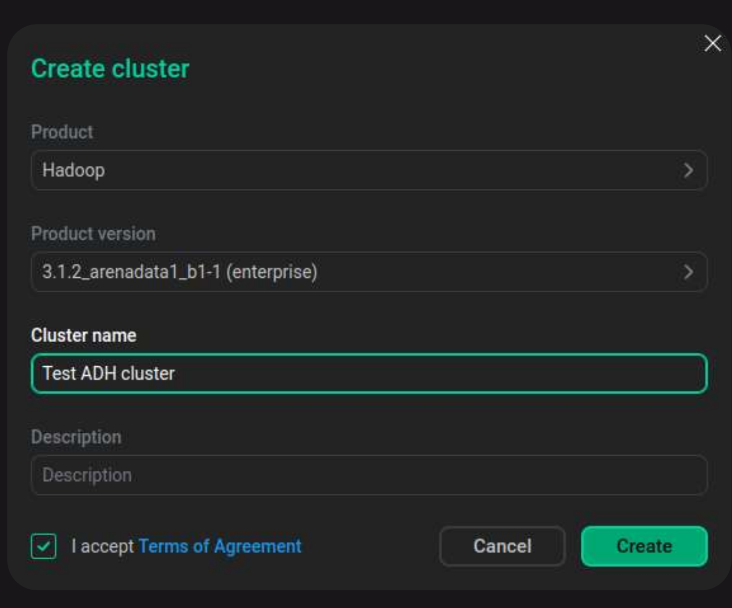
1. Выберите пункт ***Clusters*** в левом навигационном меню и нажмите ***Create cluster***.



Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Create cluster
Name
State
Product
Version
Description
Concerns
Actions
No Data

Переход к созданию кластера

2. В открывшемся окне требуется описать новый кластер:
   - Выберите загруженный бандл в поле ***Product***.
   - Выберите версию бандла в поле ***Product version***. Несколько версий становятся доступны после загрузки нескольких бандлов для одного типа продуктов.
   - Введите имя кластера в поле ***Cluster name***.
   - При необходимости добавьте описание кластера в поле ***Description***.
   - Установите флаг ***I accept Terms of Agreement*** для подписания лицензионного соглашения. Флаг отображается только для бандлов версии Enterprise. Чтобы ознакомиться с текстом соглашения, перейдите по ссылке ***Terms of Agreement***. Подписать соглашение также возможно на странице ***Bundles***, нажав на соответствующий продуктовый бандл.
   - Нажмите ***Create***.


Create cluster

Product: Hadoop

Product version: 3.1.2_arenadata1_b1-1 (enterprise)

Cluster name: Test ADH cluster

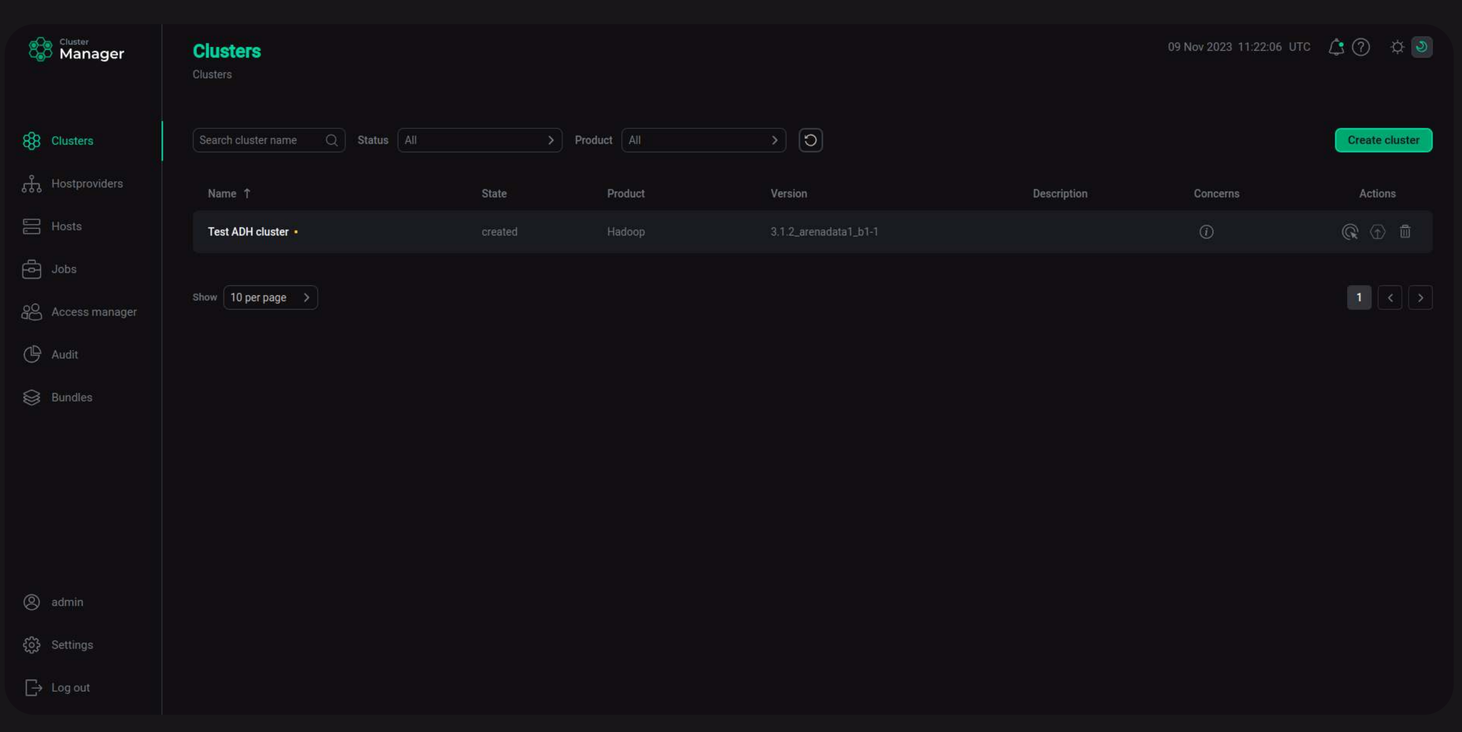
Description: Description

☑ I accept Terms of Agreement

Cancel
Create

Создание кластера

3. После успешного добавления кластер отображается на странице ***Clusters***.



Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Create cluster

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

Результат успешного добавления кластера

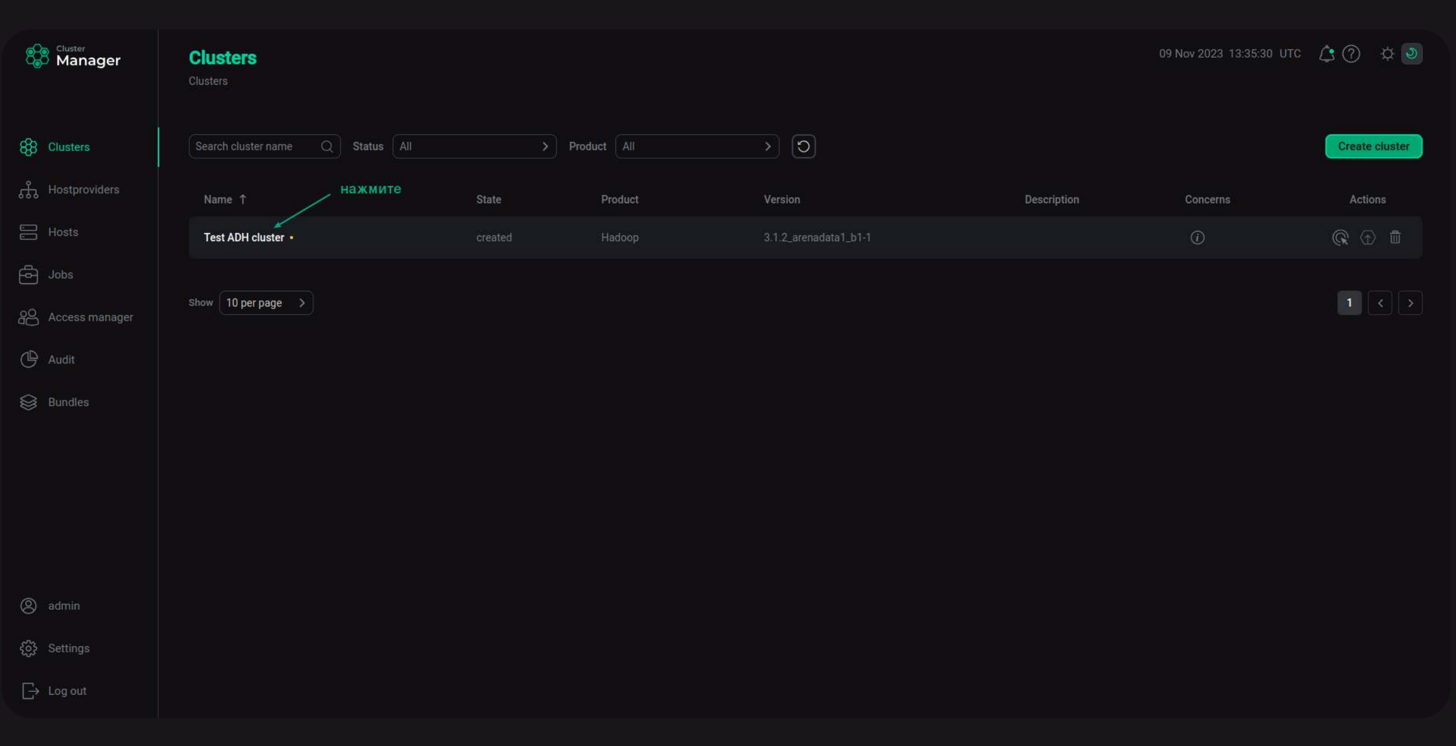
Добавление сервисов

Евгения Кузина, Дарья Барышева

Назад к содержанию

В ADCM **сервис** означает программное обеспечение, выполняющее некоторую функцию. Примеры сервисов в кластерах ADH: HDFS, HBase, Hive и другие. Для добавления сервисов в кластер:

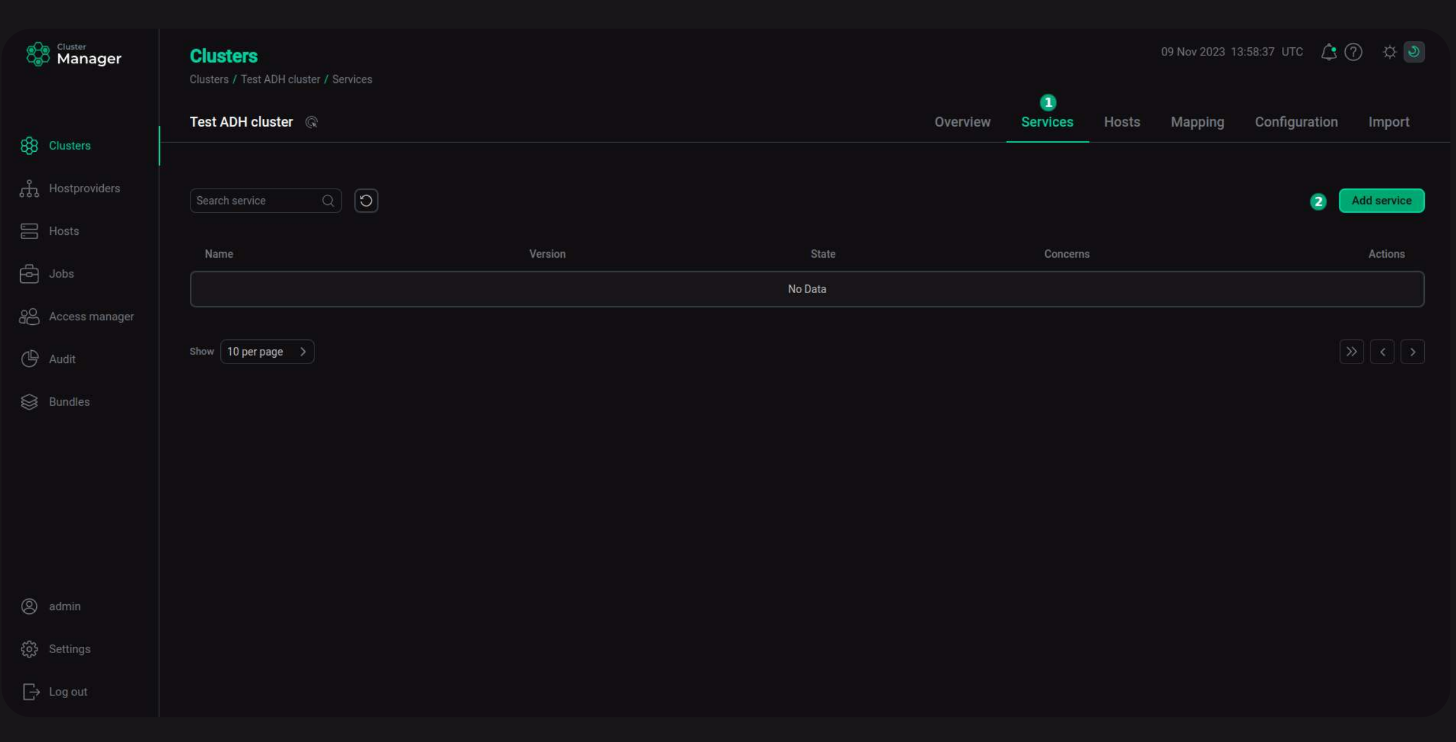
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.



Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

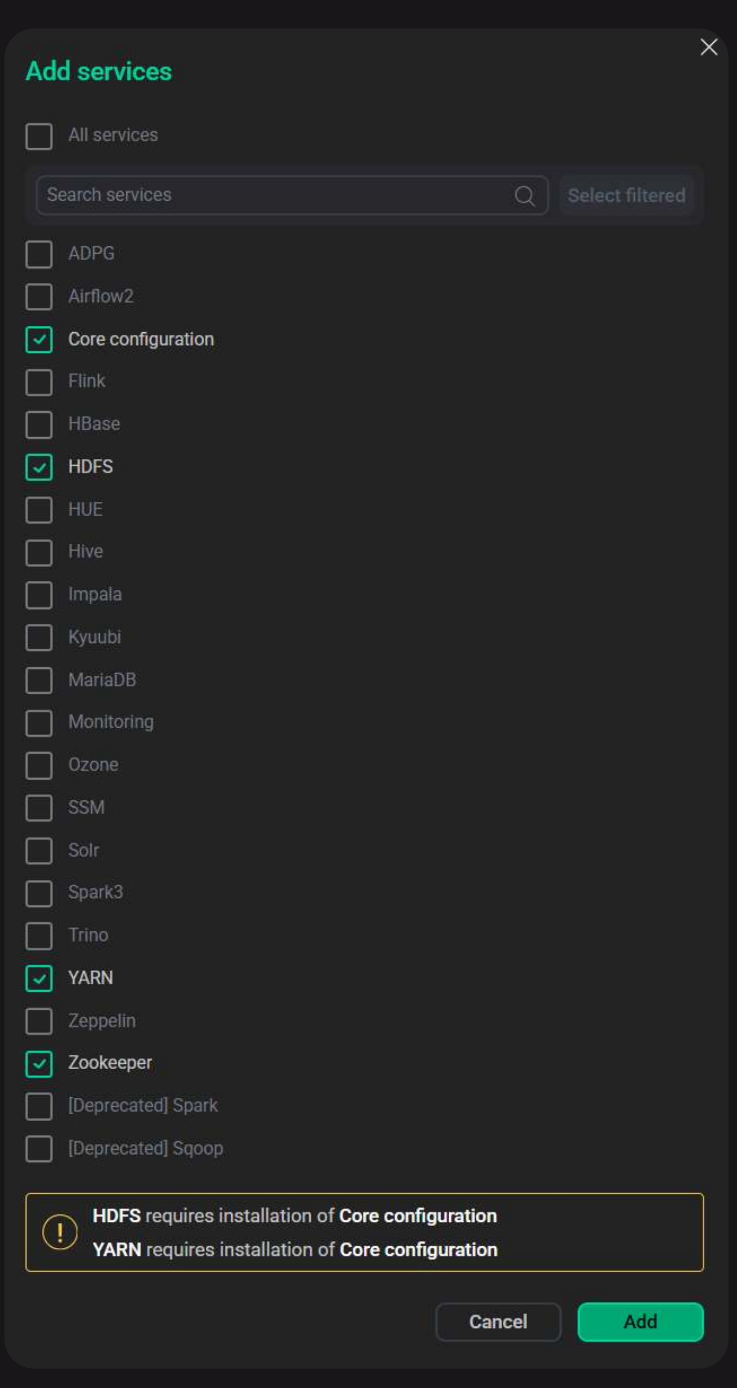
Выбор кластера

2. Откройте вкладку ***Services*** на странице кластера и нажмите ***Add service***.



Переход к добавлению сервисов

3. В открывшемся окне выберите сервисы для добавления в кластер и нажмите ***Add***.



- ☐ All services
- ☐ ADPG
- ☐ Airflow2
- ☑ Core configuration
- ☐ Flink
- ☐ HBase
- ☑ HDFS
- ☐ HUE
- ☐ Hive
- ☐ Impala
- ☐ Kyuubi
- ☐ MariaDB
- ☐ Monitoring
- ☐ Ozone
- ☐ SSM
- ☐ Solr
- ☐ Spark3
- ☐ Trino
- ☑ YARN
- ☐ Zeppelin
- ☑ Zookeeper
- ☐ [Deprecated] Spark
- ☐ [Deprecated] Sqoop

HDFS requires installation of **Core configuration**
YARN requires installation of **Core configuration**

Выбор сервисов

Краткое описание доступных сервисов приведено ниже.

Сервисы, которые могут быть добавлены в кластер ADH

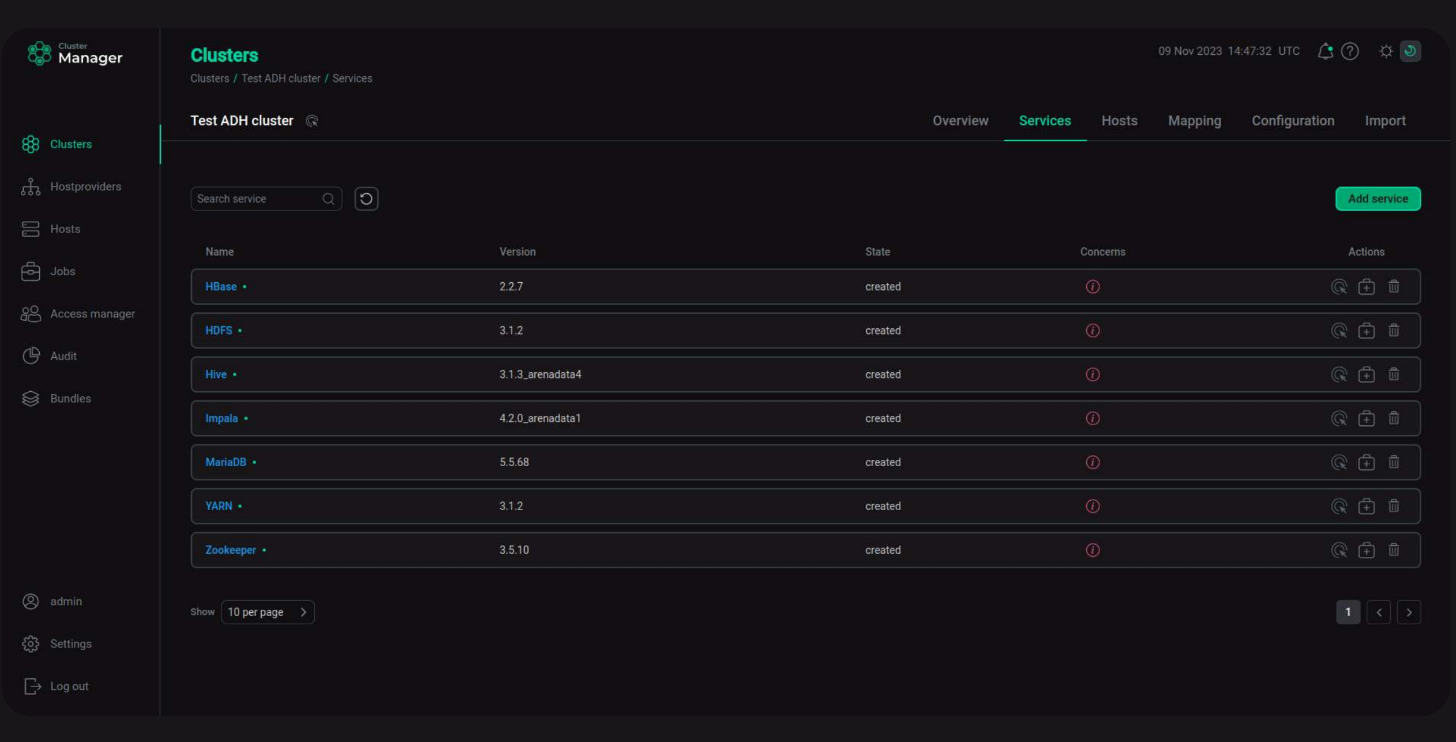
Сервис	Назначение
Airflow2	По сравнению с версией Airflow 1.x, Airflow2 предлагает дополнительные функции: высокая отказоустойчивость (High Availability), уменьшенная задержка выполнения задач, полный REST API, TaskFlow API, группы задач, независимые провайдеры и другие
Core configuration	Сервис, содержащий общие пакеты Hadoop, необходимые для всех остальных сервисов, кроме ADPG, Solr, Zookeeper и Monitoring. Этот сервис устанавливается на все узлы. Если уже установленный сервис Hadoop расширяется на другой хост, сервис Core configuration автоматически устанавливается на этот хост. Сервис Core configuration поддерживается начиная с версии ADH 3.3.6.2.b1. Конфигурационные параметры сервиса описаны в статье Конфигурационные параметры. После применения любых изменений в конфигурации сервиса необходимо выполнить его действие ***Update Core configuration***
Flink	Распределенная платформа, используемая в высоконагруженных приложениях Big Data для анализа данных, хранимых в кластерах Hyperwave. Наиболее частые сценарии использования: приложения, управляемые событиями (event-driven), потоковая и пакетная аналитика, конвейеры данных, ETL и другое
Flink2	Сервис Flink с версией 2.x. Сервис добавлен в режиме технологического превью, может иметь ряд ограничений и не предназначен для использования в производственных средах
HBase	Нереляционная распределенная база данных, написанная на Java и работающая поверх HDFS. Относится к классу колоночных СУБД, хранящих данные в формате key-value. Используется для произвольного доступа к большим данным на чтение и запись в режиме реального времени
HDFS	Распределенная файловая система, используемая в кластерах Hyperwave для хранения больших файлов. Обеспечивает возможность потокового доступа к информации, распределенной поблочно по нодам кластера
Hive	Сервис, предназначенный для создания корпоративных хранилищ данных (Data Warehouse, DWH) и анализа Big Data. Работает поверх HDFS и других совместимых систем, таких как Apache HBase. Использование Hive облегчает запись, чтение и управление большими наборами данных, хранимыми в распределенных системах
HUE	Apache HUE — это SQL-помощник с графическим интерфейсом, который предоставляет интерпретаторы для различных источников данных, таких как Impala, Hive, Kyuubi и многих других. Он обладает удобным редактором запросов с умной автоподстановкой, функцией "drag and drop", меню быстрой навигации и другими полезными функциями
Impala	Impala обеспечивает быстрые интерактивные SQL-запросы к данным, хранящимся в HDFS, HBase или S3-хранилище. В дополнение к унифицированной платформе хранения Impala также использует те же метаданные, синтаксис SQL (Hive SQL) и драйвер JDBC, что и Apache Hive. Это делает Impala унифицированной платформой для запросов в режиме реального времени или пакетных запросов
Kyuubi	Apache Kyuubi — это распределенный многопользовательский шлюз для предоставления SQL для DWH и DataLake. Kyuubi создает распределенные механизмы запросов SQL поверх различных видов современных вычислительных платформ, например, Apache Spark, Flink, Hive, Impala и т. д., чтобы получать и обрабатывать большие наборы распределенных данных из разнородных источников
Monitoring	Сервис, добавляемый в случаях, когда запланирована настройка мониторинга ADH через ADCM
Ozone	Apache Ozone (O3) — распределенное хранилище объектов, оптимизированное для работы как с сервисами Hadoop, так и с хранилищами S3. Ozone обладает высокой масштабируемостью, может использовать те же политики безопасности кластера, что и HDFS, и способен эффективно обрабатывать файлы любого размера
Solr	Поисковая платформа, основанная на проекте Apache Lucene. Основные функции Solr включают полнотекстовый поиск, фасетный поиск, выделение результатов поиска (highlighting), распределенное индексирование, интеграцию с базами данных, обработку документов со сложными форматами (Word, PDF и так далее), запросы с балансировкой нагрузки, централизованную настройку и другое
Spark3	Аналитический фреймворк, используемый для быстрой обработки больших объемов данных. Spark3 может работать в кластерах Hyperwave, используя standalone-режим YARN или Spark. Он может обрабатывать данные, поступающие в форматах HDFS, HBase, Cassandra, Hive и любых других форматах, поддерживаемых в Hadoop. Применяется для пакетной обработки и других задач: потоковой передачи, машинного обучения, интерактивных запросов и так далее. Версия Spark 3.x предлагает дополнительные функции: адаптивное выполнение Spark SQL, Dynamic Partition Pruning (DPP), обработку графов, расширенную поддержку Deep Learning и другое
Spark4	Сервис Spark с версией 4.x. Сервис добавлен в режиме технологического превью, может иметь ряд ограничений и не предназначен для использования в производственных средах
SSM	Smart Storage Manager — это сервис, цель которого — оптимизировать эффективность хранения и управления данными в Hadoop Distributed File System. SSM собирает данные о работе HDFS и информацию о состоянии системы и на основе собранных показателей может автоматически использовать такие методологии как кеш, политики хранения данных, управление гетерогенными хранилищами (HSM), сжатие данных и Erasure Coding. Кроме того, SSM предоставляет возможность настройки асинхронной репликации данных и пространства имен на резервный кластер с целью организации DR
Trino	Trino — это движок SQL-запросов с открытым исходным кодом, используемый для параллельной обработки данных, распределенных по хранилищам разного типа, таких как объектные хранилища (S3), базы данных и файловые системы
YARN	Сервис, необходимый для управления ресурсами кластера и планирования/мониторинга задач (jobs). Использует специальный демон (Resource Manager), абстрагирующий все вычислительные ресурсы кластера и управляющий их предоставлением распределенным приложениям
Zeppelin	Сервис, который играет роль web-блокнота и обеспечивает интерактивный анализ данных. Позволяет создавать запросы к данным в кластерах Hyperwave и отображать результаты в виде таблиц, графиков, диаграмм и так далее
Zookeeper	Сервис централизованного управления распределенными приложениями. Он используется в кластерах Hyperwave для обнаружения сбоев, выбора активных NameNode, мониторинга работоспособности (health checks), управления сеансами и так далее

Минимальный набор сервисов, рекомендуемый для кластеров ADH, приведен ниже:

- Core configuration;
- HDFS/Ozone;
- YARN;
- Zookeeper (опционально для Community-версии ADH).

Эти сервисы составляют ядро Hadoop и являются достаточными для организации распределенного хранения и обработки данных. Полный перечень сервисов будет зависеть от требований конкретного проекта.

4. В результате успешно добавленные сервисы отображаются на вкладке *Services*.



Name	Version	State	Concerns	Actions
HBase	2.2.7	created		
HDFS	3.1.2	created		
Hive	3.1.3_arenadata4	created		
Impala	4.2.0_arenadata1	created		
MariaDB	5.5.68	created		
YARN	3.1.2	created		
Zookeeper	3.5.10	created		

Результат успешного добавления сервисов в кластер

> **ПРИМЕЧАНИЕ**
>
> Опциональные сервисы могут быть добавлены в кластер позднее. Процесс добавления сервисов в уже развернутый кластер не отличается от описанного выше.

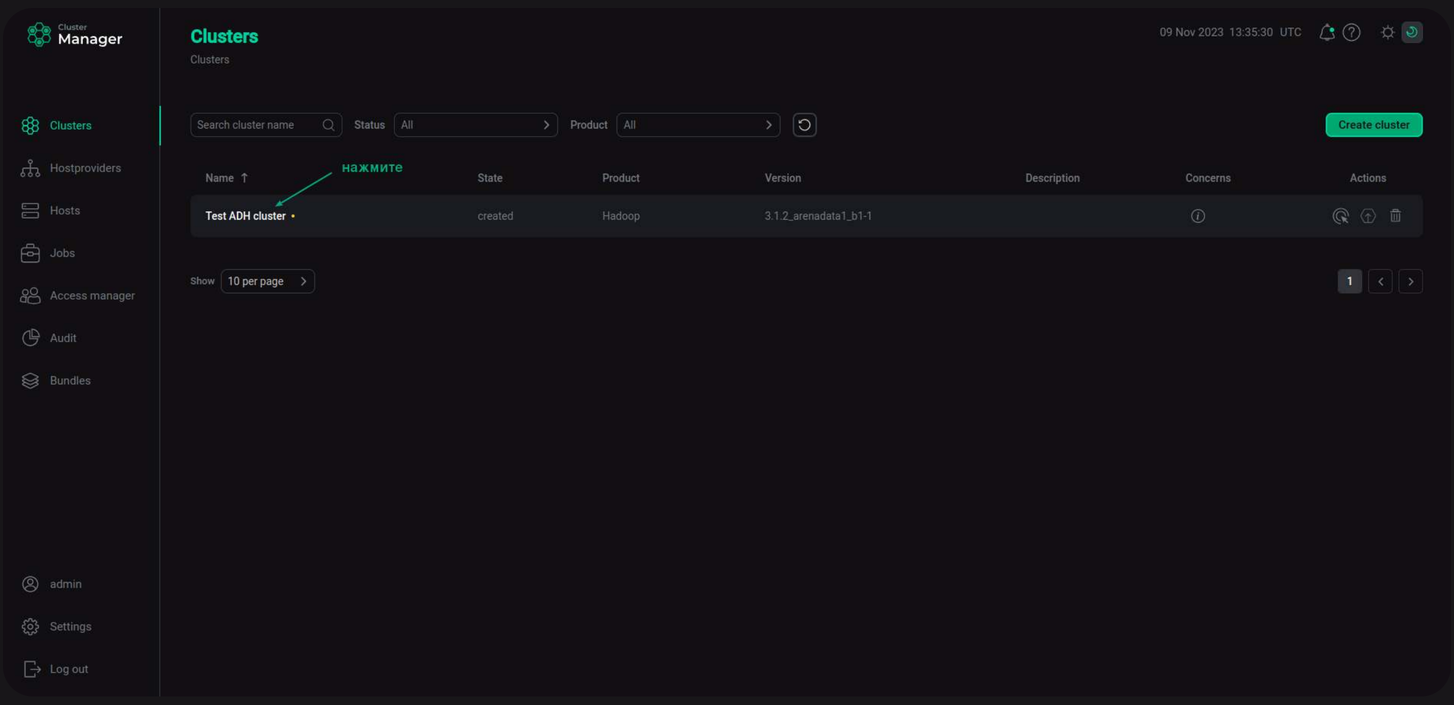
Добавление хостов в кластер

Евгения Кузина, Дарья Барышева

Назад к содержанию

Для добавления хостов в кластер:

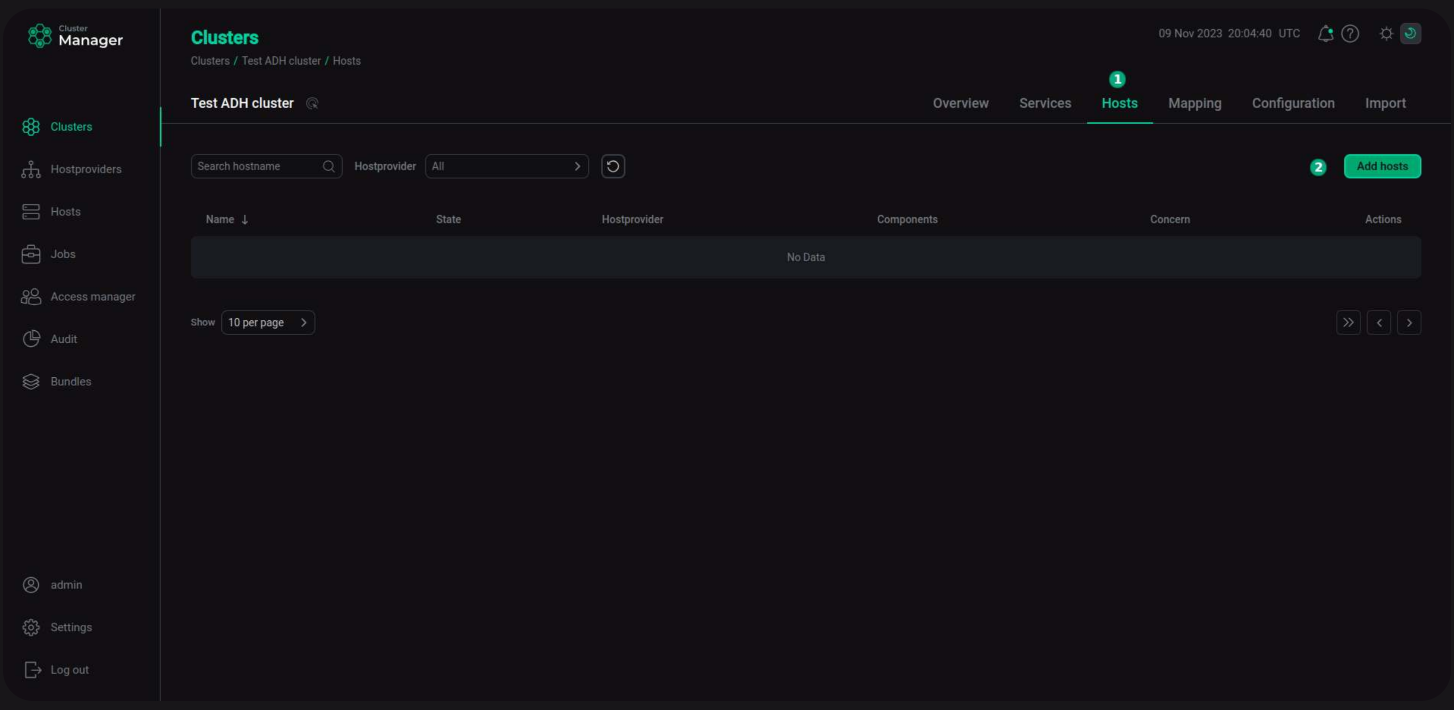
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Clusters
09 Nov 2023 13:35:30 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
Product
Create cluster
нажмите
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

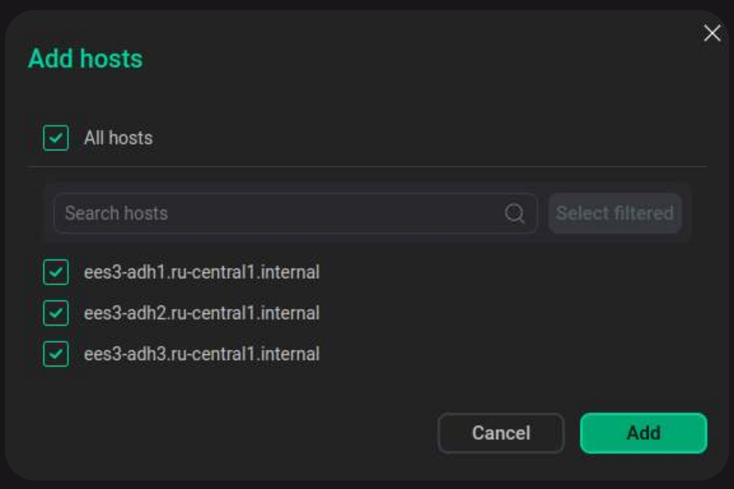
Выбор кластера

2. Откройте вкладку ***Hosts*** на странице кластера и нажмите ***Add hosts***.


Cluster Manager
Clusters
Clusters / Test ADH cluster / Hosts
09 Nov 2023 20:04:40 UTC
Test ADH cluster
Overview
Services
1
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search hostname
Hostprovider
All
2
Add hosts
Name
State
Hostprovider
Components
Concern
Actions
No Data
Show
10 per page

Переход к добавлению хостов

3. В открывшемся окне выберите хосты для добавления в кластер и нажмите ***Add***.


Add hosts
Search hosts
Select filtered
Cancel
Add

☑ All hosts

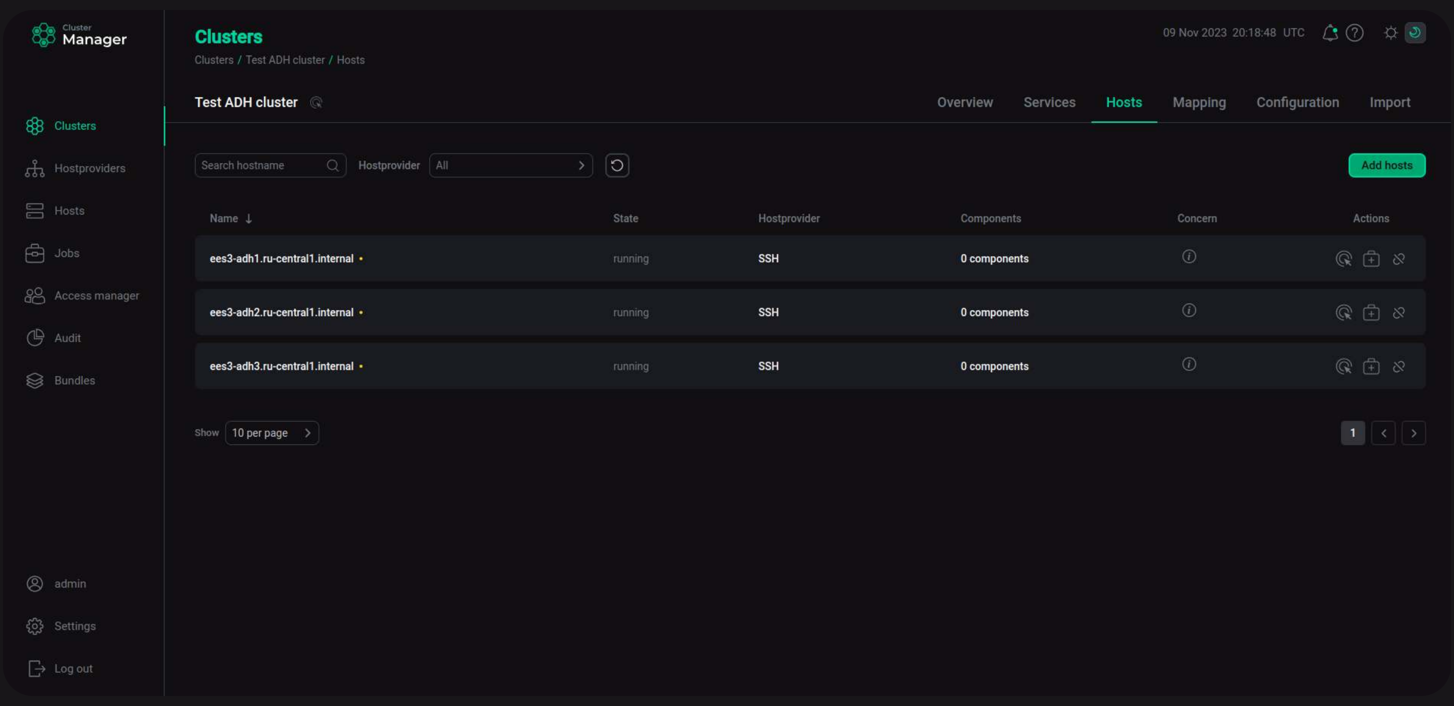
☑ ees3-adh1.ru-central1.internal

☑ ees3-adh2.ru-central1.internal

☑ ees3-adh3.ru-central1.internal

Выбор хостов

4. В результате успешно добавленные хосты отображаются на вкладке ***Hosts***.


Cluster Manager
Clusters
Clusters / Test ADH cluster / Hosts
09 Nov 2023 20:18:48 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search hostname
Hostprovider
All
Add hosts
Show
10 per page
1

Name	State	Hostprovider	Components	Concern	Actions
ees3-adh1.ru-central1.internal	running	SSH	0 components		
ees3-adh2.ru-central1.internal	running	SSH	0 components		
ees3-adh3.ru-central1.internal	running	SSH	0 components		

Результат успешного добавления хостов в кластер

Добавление компонентов

Евгения Кузина, Дарья Барышева

Назад к содержанию

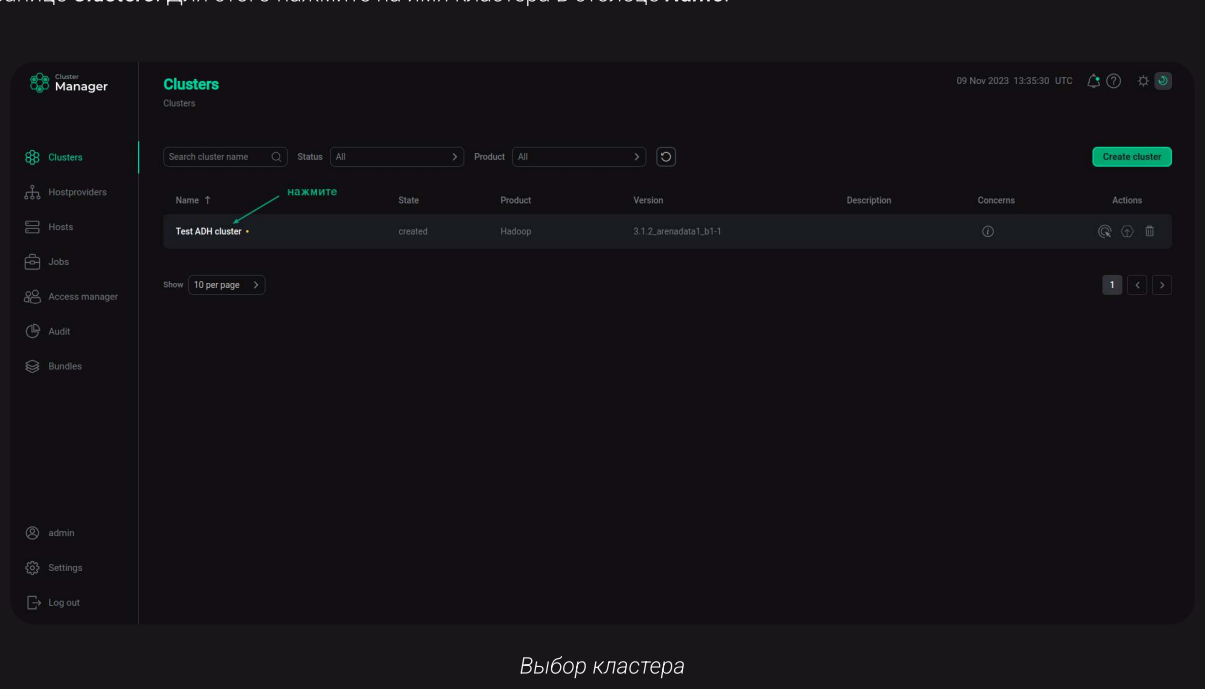
В ADCM **компонент** означает часть сервиса, которая должна быть развернута на одном или нескольких хостах кластера. Например, сервис HDFS состоит из таких компонентов, как DataNode, NameNode, Client и другие.

ВАЖНО

Изначально на хостах кластера нет ни одного компонента — распределение компонентов между хостами должно быть проведено вручную.

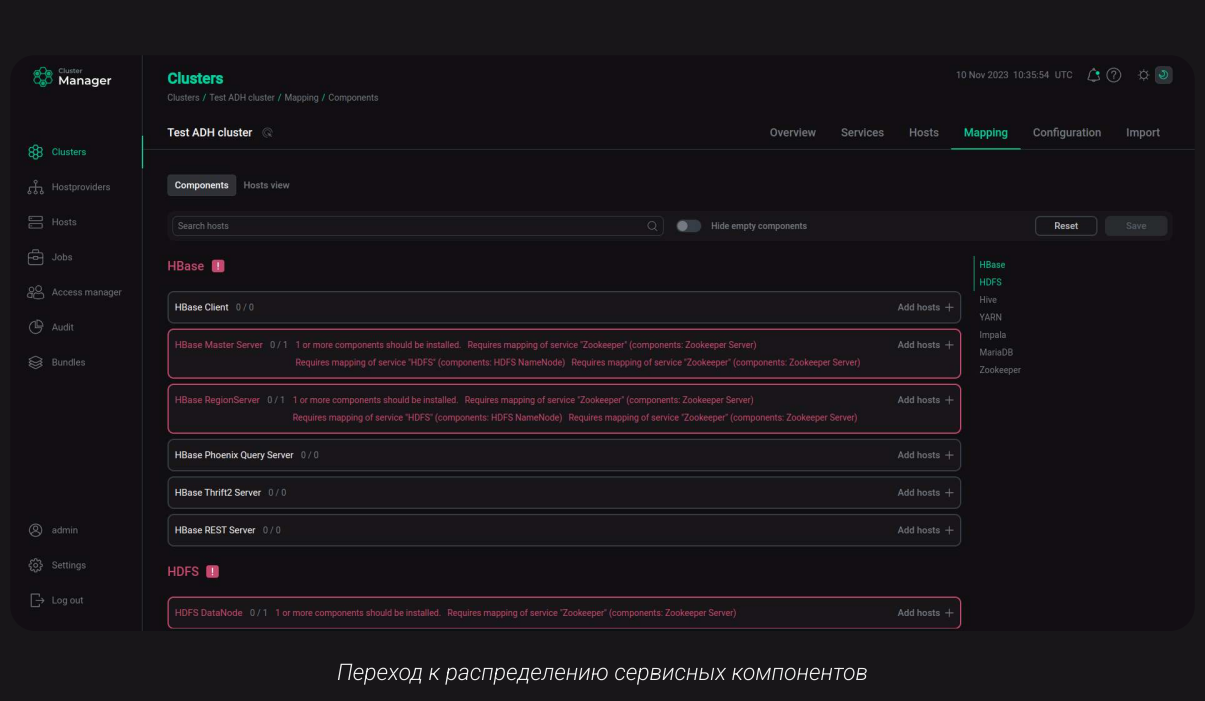
Для добавления компонентов на хосты кластера:

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Create cluster
Name
State
Product
Version
Description
Concerns
Actions
нажмите
Test ADH cluster
created
Hadoop
3.1.2_arenadata1_b1-1
10 per page

Выбор кластера

2. Откройте вкладку ***Mapping*** на странице кластера.


Manager
Clusters
Clusters / Test ADH cluster / Mapping / Components
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Components
Hosts view
Search hosts
Hide empty components
Reset
Save
HBase
HDFS
Hive
YARN
Impala
MariaDB
Zookeeper
Add hosts +

HBase Client 0 / 0

HBase Master Server 0 / 1 1 or more components should be installed. Requires mapping of service "Zookeeper" (components: Zookeeper Server) Requires mapping of service "HDFS" (components: HDFS NameNode) Requires mapping of service "Zookeeper" (components: Zookeeper Server)

HBase RegionServer 0 / 1 1 or more components should be installed. Requires mapping of service "Zookeeper" (components: Zookeeper Server) Requires mapping of service "HDFS" (components: HDFS NameNode) Requires mapping of service "Zookeeper" (components: Zookeeper Server)

HBase Phoenix Query Server 0 / 0

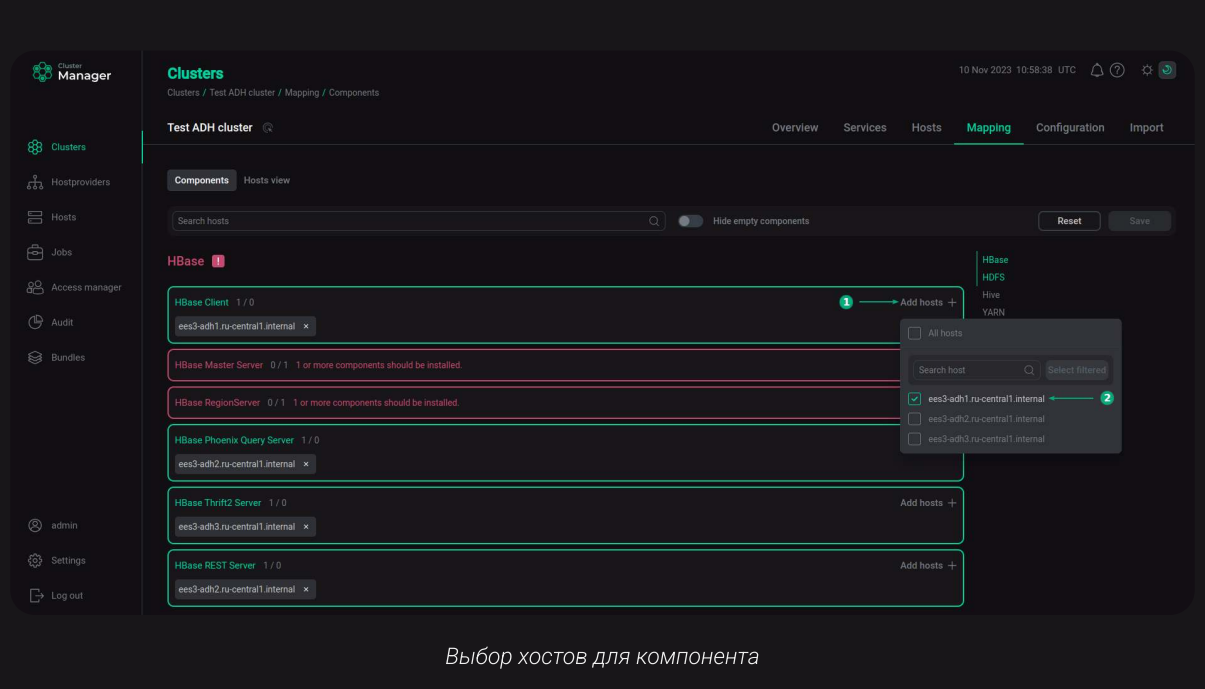
HBase Thrift2 Server 0 / 0

HBase REST Server 0 / 0

HDFS DataNode 0 / 1 1 or more components should be installed. Requires mapping of service "Zookeeper" (components: Zookeeper Server)

Переход к распределению сервисных компонентов

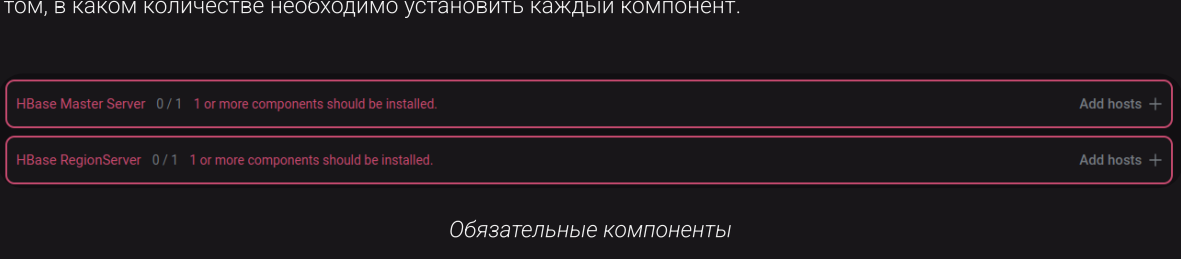
3. Распределите все компоненты между хостами кластера. Для назначения хостов компоненту нажмите ***Add hosts*** и выберите хосты в открывшемся окне.


Manager
Clusters
Clusters / Test ADH cluster / Mapping / Components
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Components
Hosts view
Hide empty components
Reset
Save
HBase
HBase Client 1 / 0
1
Add hosts +
All hosts
Search host
Select filtered
ees3-adh1.ru-central1.internal
2
ees3-adh2.ru-central1.internal
ees3-adh3.ru-central1.internal
HBase Master Server 0 / 1 1 or more components should be installed.
HBase RegionServer 0 / 1 1 or more components should be installed.
HBase Phoenix Query Server 1 / 0
HBase Thrift2 Server 1 / 0
HBase REST Server 1 / 0

Выбор хостов для компонента

Обязательные компоненты

Каждый сервис может иметь обязательные и опциональные компоненты. Обязательные подсвечиваются красным цветом. Также в интерфейсе ADCM для них выводится информация о том, в каком количестве необходимо установить каждый компонент.



HBase Master Server 0 / 1 1 or more components should be installed. Add hosts +

HBase RegionServer 0 / 1 1 or more components should be installed. Add hosts +

Обязательные компоненты

ВНИМАНИЕ

Без назначения нужного числа хостов обязательным компонентам сохранение общей схемы распределения невозможно.

Следующая таблица содержит примеры распределения компонентов в кластерах ADH, состоящих из 3, 6 и 9 хостов. Все примеры основаны на Enterprise-версии ADH.

Примеры распределения компонентов

3 хоста

Хост	Сервис	Компонент
Хост 1	Core configuration	Configuration server
	Flink	Flink Client Flink JobManager Flink TaskManager Flink SQL-Gateway
	HBase	HBase Client HBase RegionServer
	HDFS	HDFS Client HDFS DataNode HDFS JournalNode HDFS NameNode HDFS ZKFC
	Hive	Hive Client Hive Metastore Hive Tez
	Impala	Impala Daemon Impala Statestore Impala Catalog Service
	Monitoring	Grafana Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Recon Ozone HttpFS Ozone Datanode
	Solr	Solr Server
	Spark3	Spark3 Client Spark3 History server
	Trino	Trino Coordinator Trino Client
	YARN	MapReduce History server YARN Client YARN NodeManager
	ZooKeeper	ZooKeeper Server
Хост 2	Airflow2	Airflow2 Broker Airflow2 Flower Airflow2 Scheduler Airflow2 Server Airflow2 Worker
	Flink	Flink Client Flink History Server
	HBase	HBase Client HBase Master server HBase Phoenix query server HBase RegionServer
	HDFS	HDFS Client HDFS DataNode HDFS HttpFS server HDFS JournalNode
	Hive	Hive Client Hive HiveServer2 Hive Tez Hive TezUI
	Impala	Impala Daemon
	Monitoring	Prometheus Server Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Datanode
	Solr	Solr Server Solr prometheus exporter
	Spark3	Spark3 Client Spark3 Livy server
	Trino	Trino Worker Trino Client
	YARN	YARN Client YARN NodeManager YARN ResourceManager
	Zeppelin	Zeppelin Server
	ZooKeeper	ZooKeeper Server
Хост 3	Arenadata PostgreSQL	Arenadata PostgreSQL
	Flink	Flink Client
	HBase	HBase Client HBase RegionServer HBase Thrift2 server HBase REST Server
	HDFS	HDFS Client HDFS DataNode HDFS JournalNode HDFS NameNode HDFS ZKFC
	Hive	Hive Client Hive Tez
	HUE	HUE Server
	Impala	Impala Daemon Impala Client
	Kyuubi	Kyuubi Server
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone S3G Ozone Datanode
	Solr	Solr Server
	Spark3	Spark3 Client
	SSM	SSM Server SSM Agent
	Trino	Trino Worker Trino Client
	YARN	YARN Client YARN NodeManager YARN Timeline server
	ZooKeeper	ZooKeeper Server

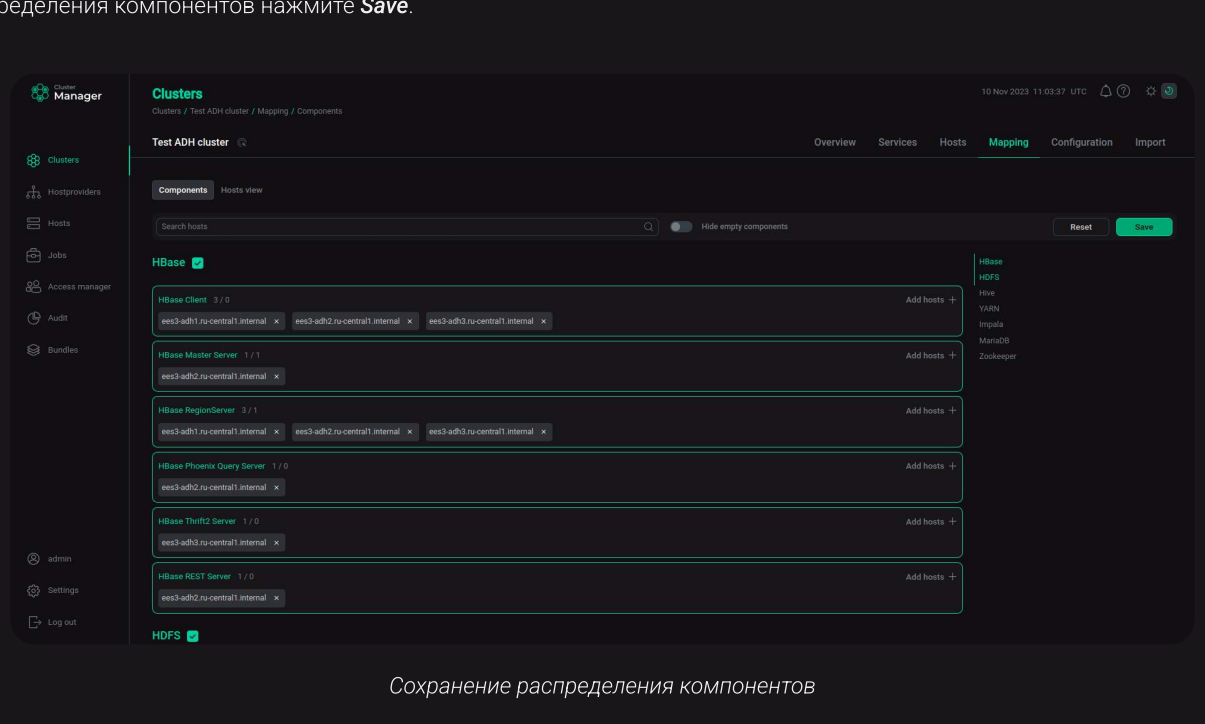
6 хостов

Хост	Сервис	Компонент
Хост 1	Core configuration	Configuration server
	Flink	Flink JobManager Flink TaskManager Flink SQL-Gateway
	HDFS	HDFS JournalNode HDFS NameNode HDFS ZKFC
	Hive	Hive Metastore
	Impala	Impala Statestore Impala Catalog Service
	Monitoring	Grafana Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Recon Ozone HttpFS
	Solr	Solr Server Solr prometheus exporter
	Spark3	Spark3 History server
	Trino	Trino Coordinator Trino Client
	YARN	MapReduce History server
	ZooKeeper	ZooKeeper Server
Хост 2	Airflow2	Airflow2 Broker Airflow2 Flower Airflow2 Scheduler Airflow2 Server Airflow2 Worker
	HBase	HBase Master server HBase Phoenix query server
	HDFS	HDFS JournalNode HDFS HttpFS server
	Hive	Hive HiveServer2 Hive TezUI
	Monitoring	Prometheus Server Node Exporter
	Solr	Solr Server
	Spark3	Spark3 Livy server
	Trino	Trino Worker Trino Client
	YARN	YARN ResourceManager
	Zeppelin	Zeppelin Server
	ZooKeeper	ZooKeeper Server
Хост 3	Arenadata PostgreSQL	Arenadata PostgreSQL
	HBase	HBase Thrift2 server HBase REST Server
	HDFS	HDFS NameNode HDFS ZKFC HDFS JournalNode
	HUE	HUE Server
	Kyuubi	Kyuubi Server
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Solr	Solr Server
	SSM	SSM Server SSM Agent
	YARN	YARN Timeline server
	ZooKeeper	ZooKeeper Server
Хост 4	Flink	Flink Client Flink History Server
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon Impala Client
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Datanode
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 5	Flink	Flink Client
	HBase	HBase Client HBase RegionServer
	HDFS	HDFS DataNode HDFS Client
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 6	Flink	Flink Client
	HBase	HBase Client HBase RegionServer
	HDFS	HDFS DataNode HDFS Client
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone S3G
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server

9 хостов

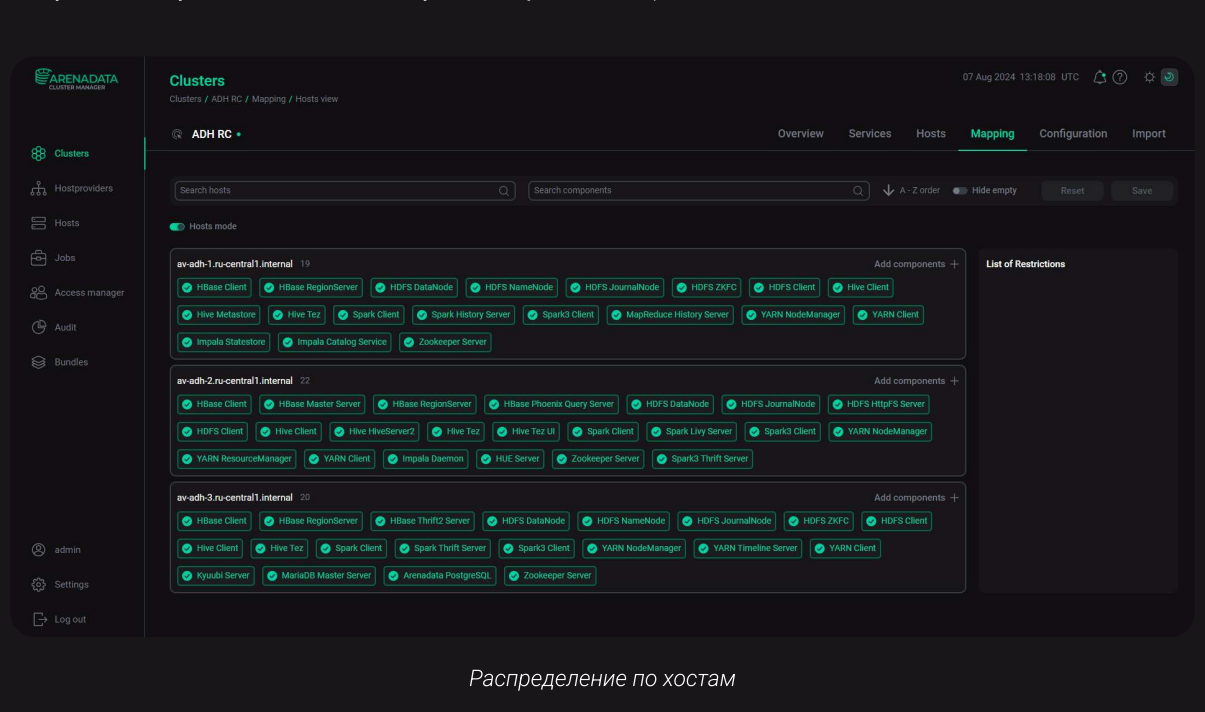
Хост	Сервис	Компонент
Хост 1	Core configuration	Configuration server
	Flink	Flink JobManager Flink TaskManager Flink SQL-Gateway
	HDFS	HDFS JournalNode HDFS NameNode HDFS ZKFC
	Hive	Hive Metastore
	Impala	Impala Statestore Impala Catalog Service
	Monitoring	Grafana Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Recon Ozone HttpFS
	Solr	Solr Server
	Spark3	Spark3 History server
	Trino	Trino Coordinator Trino Client
	YARN	MapReduce History server
	ZooKeeper	ZooKeeper Server
Хост 2	Airflow2	Airflow2 Broker Airflow2 Flower Airflow2 Scheduler Airflow2 Server Airflow2 Worker
	Flink	Flink TaskManager
	HBase	HBase Master server HBase Phoenix query server
	HDFS	HDFS JournalNode HDFS HttpFS server
	Hive	Hive HiveServer2 Hive TezUI
	Monitoring	Prometheus Server Node Exporter
	Solr	Solr Server Solr prometheus exporter
	Spark3	Spark3 Livy server
	YARN	YARN ResourceManager
	Zeppelin	Zeppelin Server
	ZooKeeper	ZooKeeper Server
Хост 3	HBase	HBase Thrift2 server HBase REST Server
	HDFS	HDFS NameNode HDFS ZKFC HDFS JournalNode
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Datanode
	Solr	Solr Server
	Trino	Trino Worker Trino Client
	YARN	YARN Timeline server
	Zeppelin	Zeppelin Server
	ZooKeeper	ZooKeeper Server
Хост 4	Arenadata PostgreSQL	Arenadata PostgreSQL
	HDFS	HDFS JournalNode HDFS NameNode HDFS ZKFC HDFS HttpFS server
	HBase	HBase Master server
	HUE	HUE Server
	Kyuubi	Kyuubi Server
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Solr	Solr Server
	SSM	SSM Server SSM Agent
	Trino	Trino Worker Trino Client
	ZooKeeper	ZooKeeper Server
Хост 5	HDFS	HDFS JournalNode HDFS NameNode HDFS ZKFC
	HBase	HBase Master server
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone S3G
	Solr	Solr Server
	Trino	Trino Worker Trino Client
	ZooKeeper	ZooKeeper Server
Хост 6	Flink	Flink Client Flink History Server
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon Impala Client
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Solr	Solr Server
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 7	Flink	Flink Client
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Solr	Solr Server
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 8	Flink	Flink Client
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 9	Flink	Flink Client
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server

4. После завершения распределения компонентов нажмите ***Save***.


Manager
Clusters
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Components
Hosts view
Reset
Save
HBase
HDFS
Add hosts +

Сохранение распределения компонентов

Начиная с версии 2.2.0 ADCM поддерживает альтернативный способ сопоставления — по хостам. Для этого переведите в активное положение переключатель ***Hosts mode*** и распределите компоненты между хостами путем нажатия на кнопку ***Add components*** напротив каждого хоста.


ARENADATA
Clusters
Clusters / ADH RC / Mapping / Hosts view
ADH RC
Overview
Services
Hosts
Mapping
Configuration
Import
Search hosts
Search components
Hosts mode
av-adh-1.ru-central1.internal
av-adh-2.ru-central1.internal
av-adh-3.ru-central1.internal
Add components +
List of Restrictions

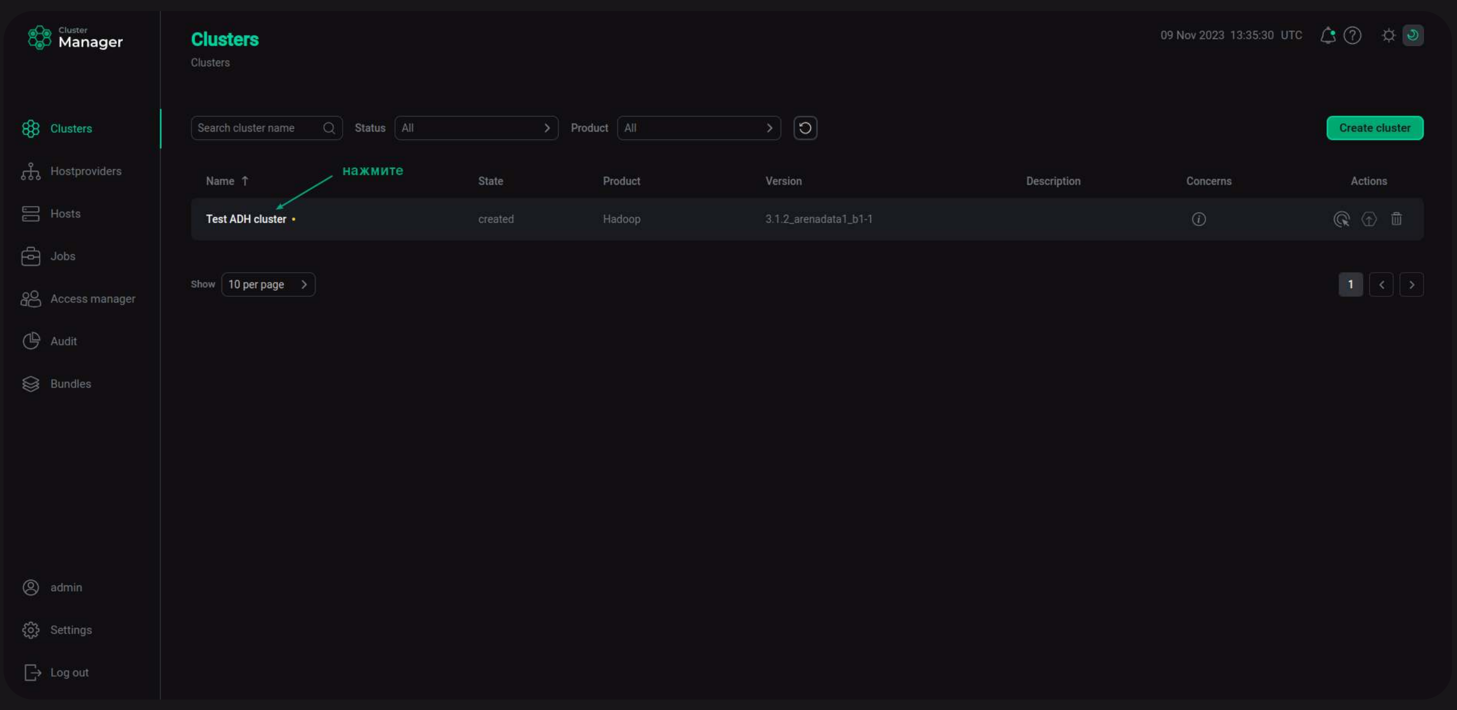
Распределение по хостам

Настройка сервисов

Евгения Кузина, Дарья Барышева

Назад к содержанию

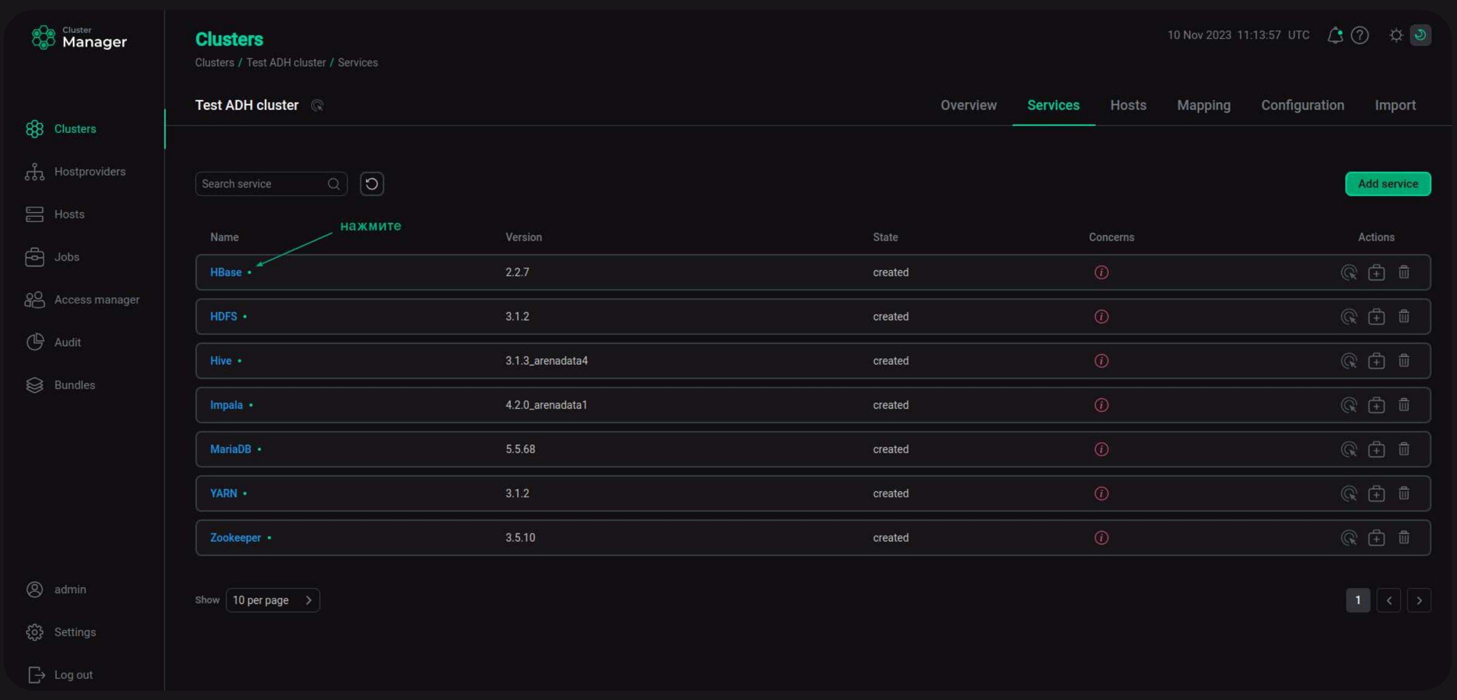
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster
Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
09 Nov 2023 13:35:30 UTC
Search cluster name
Status
All
Product
All
Create cluster
нажмите
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

Выбор кластера

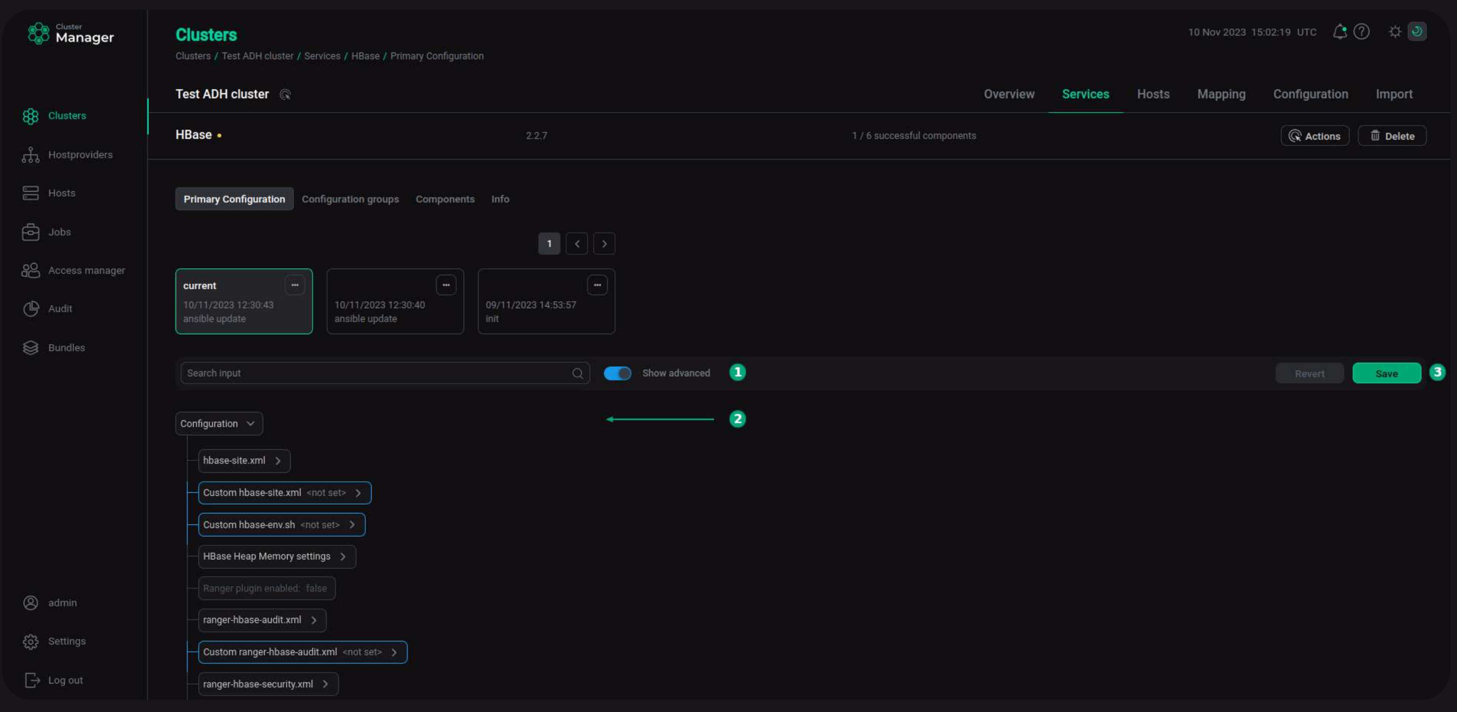
2. Откройте вкладку ***Services*** на странице кластера. Нажмите на имя сервиса в столбце ***Name***.


Cluster
Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test ADH cluster / Services
10 Nov 2023 11:13:57 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add service
нажмите
Show
10 per page
1

Name	Version	State	Concerns	Actions
HBase	2.2.7	created		
HDFS	3.1.2	created		
Hive	3.1.3_arenadata4	created		
Impala	4.2.0_arenadata1	created		
MariaDB	5.5.68	created		
YARN	3.1.2	created		
Zookeeper	3.5.10	created		

Переход к настройке сервиса

3. В открывшемся окне:
   - Откройте вкладку ***Primary Configuration***.
   - Переведите в активное состояние переключатель ***Show advanced***.
   - Заполните конфигурационные параметры выбранного сервиса. Поля, подсвеченные красным, обязательны для заполнения.
   - Нажмите ***Save***.


Cluster
Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test ADH cluster / Services / HBase / Primary Configuration
10 Nov 2023 15:02:19 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
HBase
2.2.7
1 / 6 successful components
Actions
Delete
Primary Configuration
Configuration groups
Components
Info
1
current
10/11/2023 12:30:43
ansible update
10/11/2023 12:30:40
ansible update
09/11/2023 14:53:57
init
Search input
Show advanced
1
Revert
Save
3
2
Configuration
hbase-site.xml
Custom hbase-site.xml <not set>
Custom hbase-env.sh <not set>
HBase Heap Memory settings
Ranger plugin enabled: false
ranger-hbase-audit.xml
Custom ranger-hbase-audit.xml <not set>
ranger-hbase-security.xml

Настройка сервиса

ПРИМЕЧАНИЕ

Более подробную информацию по всем конфигурационным параметрам можно получить в статье Конфигурационные параметры.

Настройка кластера

Евгения Кузина, Дарья Барышева

Содержание

Назад к содержанию

Primary configuration

Configuration groups

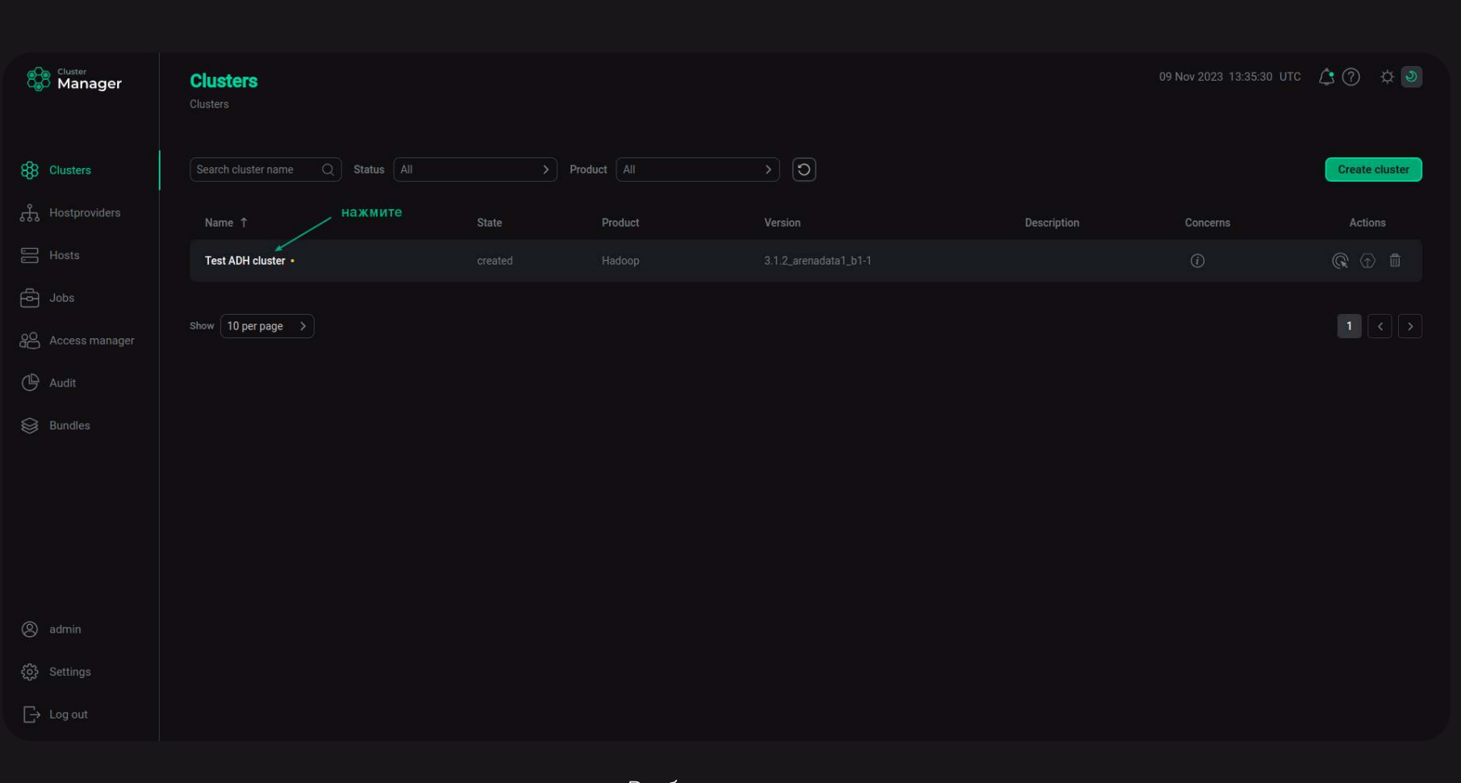
Ansible settings

ВАЖНО

В большинстве случаев кастомная настройка кластера не требуется — значения параметров можно оставить по умолчанию.

После добавления нового кластера можно выполнить его настройку. Для этого выполните шаги:

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.



Выбор кластера

2. Откройте вкладку ***Configuration*** на странице кластера. Вкладка ***Configuration*** состоит из трех подразделов: ***Primary configuration***, ***Configuration groups***, ***Ansible settings***.

3. Заполните необходимые конфигурационные параметры на нужной вкладке и нажмите ***Save***.

Primary configuration

Вкладка ***Configuration*** → ***Primary configuration*** содержит основные конфигурационные параметры кластера.

Доступные для редактирования параметры перечислены ниже:

- Раздел ***Reliability control***. Отображается, если активирован переключатель ***Show advanced***. Раздел содержит следующие настройки:
  - ***retries*** — количество попыток для операций в кластере (по умолчанию 20);
  - ***delay*** — задержка между попытками в секундах (по умолчанию 10);
  - ***timeout*** — тайм-аут проверки сетевых портов в секундах (по умолчанию 60).
- Пути к репозиториям, которые будут использованы в процессе установки в зависимости от операционной системы кластера (CentOS 7, Red Hat 7, Ubuntu 22, ALT Linux 8, Astra Linux, RedOS 7):
  - ***monitoring*** — репозиторий для установки мониторинга;
  - ***arenadata*** — YUM-репозиторий Arenadata;
  - ***arenadata_postgres*** — YUM-репозиторий Arenadata PostgreSQL;
  - ***prometheus_repo*** — YUM-репозиторий Prometheus;
  - ***arenadata_enterprise*** — YUM-репозиторий Arenadata Enterprise;
- Параметр ***DockerRegistry***, указывающий адрес Docker Registry. Значение по умолчанию — *hub.adsw.io* — Docker Registry из репозитория Arenadata.
- Раздел ***Kerberos***, содержащий параметры, перечисленные в таблице ниже. Некоторые параметры становятся видимыми только при установленном флажке ***Show advanced***.

Параметр	Описание	Значение по умолчанию
Enable Kerberos	Включает аутентификацию Kerberos	False
Authentication on WEB UIs	Включает аутентификацию Kerberos для веб-интерфейсов	False
SPNEGO Signature Value	Секретный ключ, используемый для генерации и проверки подписи в протоколе HTTP-аутентификации SPNEGO (в формате Base64)	—
Kerberos KDC type	Тип KDC. За дополнительной информацией обратитесь к статье Kerberos	MIT
KDC hosts	Один или несколько хостов KDC с запущенными серверами FreeIPA	—
Realm	Kerberos realm для подключения к серверу FreeIPA	—
Domains	Один или несколько доменов, ассоциированных с FreeIPA	—
Kadmin server	Хост, на котором работает `kadmin`	—
Kadmin principal	Имя принципала, используемое для подключения через `kadmin`, например `admin@RU-CENTRAL1.INTERNAL`	—
Kadmin password	Пароль администратора IPA	—
Keytabs directory	Каталог с keytab-файлами	/etc/security/keytabs
Additional realms	Дополнительные Kerberos realms	—
Delay between kinit invocation attempts	Задержка между попытками вызова `kinit`	5
Number of retries for kinit invocation attempts	Количество попыток вызова `kinit`	10
Trusted Active Directory server	Сервер Active Directory для обеспечения односторонних доверительных отношений между realms из MIT Kerberos KDC	—
Trusted Active Directory realm	Active Directory realm для обеспечения односторонних доверительных отношений между realms из MIT Kerberos KDC	—
Custom krb5.conf	Активирует добавление кастомных параметров в файл *krb5.conf*	False
krb5.conf	Дополнительные параметры для записи в файл *krb5.conf*	—
Admin DN	Полное отличительное имя (DistinguishedName) пользователя admin с правами `create/modify/delete/pwdchange` над пользовательскими учётными записями в целевом Organizational Unit	—
LDAP URL	LDAP URL, который состоит из префикса протокола `ldap://` или `ldaps://`, имени хоста или IP-адреса и порта AD-сервера. Например, `ldaps://192.168.4.2:636`	—
Container DN	Отличительное имя (DistinguishedName) контейнера	—
TLS CA certificate Path	Путь к сертификату CA в файловой системе хоста, который предварительно размещён или будет записан из поля ***TLS CA certificate***	—
TLS CA certificate (optional)	Сертификат CA, который записывается на хост по пути из ***TLS CA certificate Path*** во время выполнения действия ***Manage Kerberos***. Сертификат не сохраняется в ADCM	—
Custom ldap.conf	Активирует добавление кастомных параметров в файл *ldap.conf*	False
ldap.conf	Дополнительные параметры для записи в файл *ldap.conf*	—
IpaClient No NTP Autoconf	Отключает автоконфигурирование NTP во время установки клиента IPA	False
IpaClient No DNS Lookup	Отключает поиск DNS для сервера FreeIPA во время установки клиента IPA	True
Custom smb.conf	Использование кастомного файла конфигурации *smb.conf*	False
smb.conf	Кастомный файл конфигурации *smb.conf*	—
Set up Kerberos utils	Активирует возможность устанавливать и удалять клиенты и утилиты Kerberos. Данный параметр учитывается в действиях ***Expand*** и ***Install***	True
Configure Kerberos on hosts	Активирует возможность конфигурировать кластер, включая *krb5.conf*, *ldap.conf*	True
Set up principals and keytabs	Активирует возможность создавать, пересоздавать и удалять принципалов и keytab-файлы. Пароли для принципалов генерируются случайным образом перед созданием keytab-файлов. Данный параметр учитывается в действиях ***Expand*** и ***Install***. Бандл ADCM установит owner и разрешения для keytab-файлов только если этот флаг установлен. В случае отсутствия прав администратора пользователь должен предоставить подготовленный keytab-файл с правильно установленными owner и разрешениями (см. Рекомендации для пользовательских keytab-файлов)	True
Configure services and clients	Активирует возможность обновлять конфигурации сервисов и клиентов	True
Run service checks	Активирует возможность проводить проверку сервисов	True

- Раздел ***Credential encryption*** с настройками провайдера шифрования:
  - ***Credstore password*** — пароль от провайдера шифрования;
  - ***Credstore options*** — способ хранения пароля: пароль не используется (`no password`), хранить пароль в файле (`password-file`) или в переменной среды (`password in the environment`).
- Раздел ***SSL*** с настройками, перечисленными в таблице ниже.

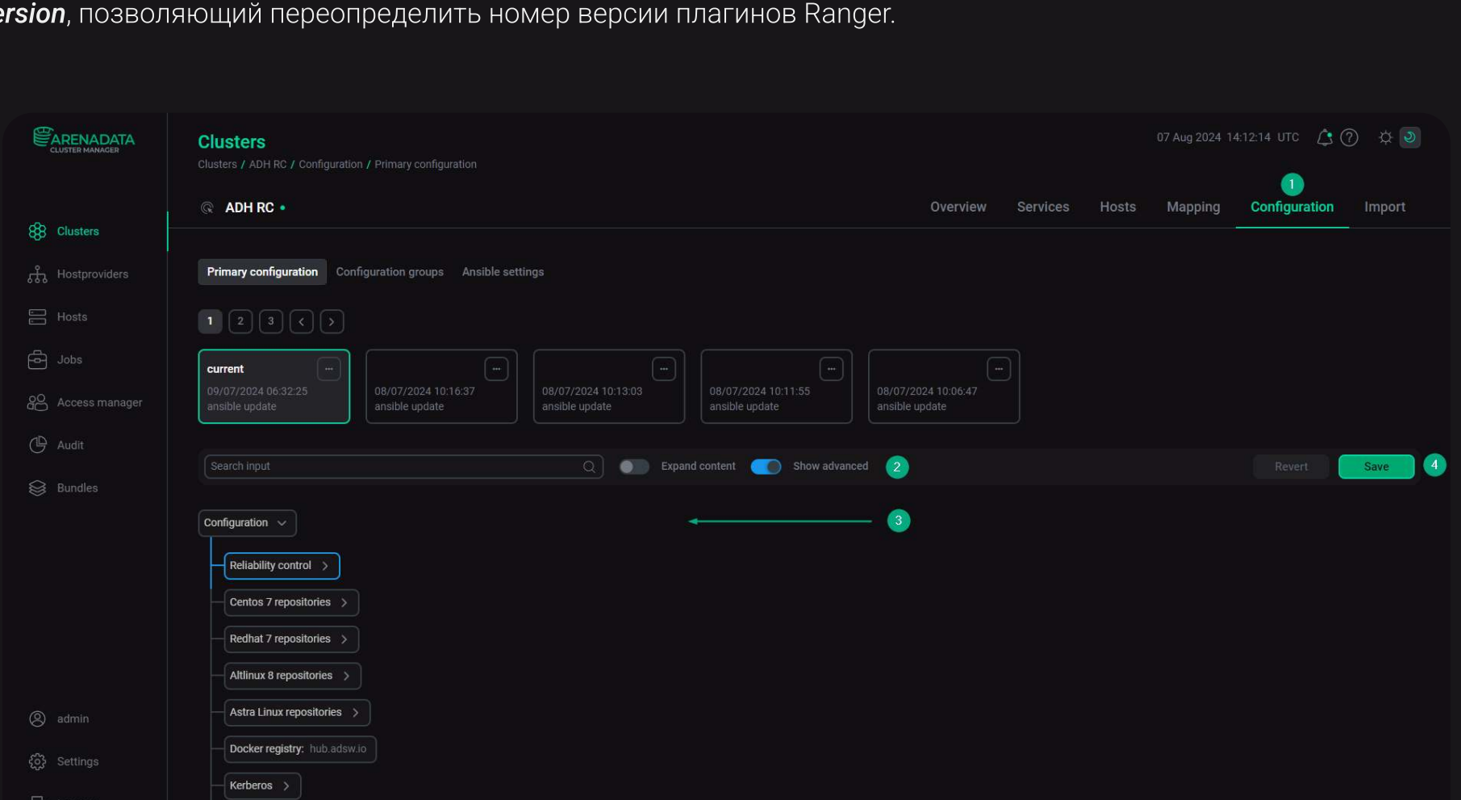
Параметр	Описание	Значение по умолчанию
Enable SSL	Включает использование SSL	false
Keystore path	Путь к keystore-файлу	/etc/ssl/keystore.jks
Keystore password	Пароль для keystore-файла	—
Truststore path	Путь к truststore-файлу	/etc/ssl/truststore.jks
Truststore password	Пароль для truststore-файла	—
TLS Version	Версия протокола TLS. См. Установка версии TLS для кластера ADH	TLSv1.2
Run service checks	Активирует возможность проводить проверку сервисов	true

- Параметр ***java.io.tmpdir***, указывающий переменную среды для управления `java.io.tmpdir` для YARN, HBase, Hive, Spark3.
- Параметр ***ad-runtime-utils*** содержит шаблон файла конфигурации, который задает параметры пути установки runtime detection для заданной среды выполнения (Java, Python и т. д.). Шаблон содержит настройки кластера, а также настройки, специфичные для сервисов, таких как Trino, Flink и Ozone. Этот параметр можно использовать для явного указания версии Java. Подробнее читайте на странице Software Requirements — Java.
  - Раздел `default` настраивает поведение кластера по умолчанию.
  - Раздел `autodetect` определяет алгоритм автоматического определения различных версий Java. Для каждой версии (8, 17, 21, 23, 24) настраивается следующее:
    - `env_var` — переменная окружения, указывающая версию (например, `JAVA8_HOME`, `JAVA17_HOME`);
    - `paths` — список путей для поиска установленной Java.
  - Раздел `services` определяет, для каких сервисов требуются конкретные версии Java.

ad-runtime-utils

```
default:
  runtimes:
    java:
      version: "8"
      env_var: JAVA_HOME

autodetect:
  runtimes:
    java:
      "8":
        env_var: JAVA8_HOME
        paths:
          - /usr/java/jdk1.8
          - /usr/java/jre1.8
          - /usr/lib/jvm/j2sdk1.8-oracle
          - /usr/lib/jvm/j2sdk1.8-oracle/jre
          - /usr/lib/jvm/java-8-oracle
          - /usr/lib/jdk8-latest
          - /usr/lib/jvm/java-1.8.0
          - /usr/lib/jvm/java-1.8.0-oracle
          - /usr/lib/jvm/bellsoft-java8-amd64
          - /usr/lib/jvm/java-arenadata-openjdk-8

      "17":
        env_var: JAVA17_HOME
        paths:
          - /usr/lib/jvm/bellsoft-java17-amd64
          - /usr/lib/jvm/bellsoft-java17
          - /usr/lib/jvm/liberica-jdk-17*
          - /usr/lib/jvm/liberica-jre-17*
          - /usr/lib/jvm/zulu-17*
          - /usr/lib/jvm/temurin-17*
          - /usr/lib/jvm/adoptopenjdk-17*
          - /usr/lib/jvm/jdk-17*
          - /usr/lib/jvm/java-17*
          - /usr/lib/jvm/openjdk-17*
          - /usr/lib/jvm/openjdk-17*
          - /opt/java/bellsoft-java17-amd64
          - /opt/java/temurin-17*
          - /usr/local/java/bellsoft-java17-amd64
          - /usr/lib/jvm/java-arenadata-openjdk-17
          - /usr/lib64/jvm/java-17*

      "21":
        env_var: JAVA21_HOME
        paths:
          - /usr/lib/jvm/java-21*
          - /usr/lib/jvm/jdk-21*
          - /usr/lib/jvm/java-arenadata-openjdk-21

      "23":
        env_var: JAVA23_HOME
        paths:
          - /usr/lib/jvm/bellsoft-java23-amd64
          - /usr/lib/jvm/bellsoft-java23
          - /usr/lib/jvm/liberica-jdk-23*
          - /usr/lib/jvm/liberica-jre-23*
          - /usr/lib/jvm/zulu-23*
          - /usr/lib/jvm/temurin-23*
          - /usr/lib/jvm/adoptopenjdk-23*
          - /usr/lib/jvm/jdk-23*
          - /usr/lib/jvm/java-23-openjdk
          - /usr/lib/jvm/openjdk-23*
          - /opt/java/bellsoft-java23-amd64
          - /opt/java/temurin-23*
          - /usr/local/java/bellsoft-java23-amd64
          - /usr/lib/jvm/java-arenadata-openjdk-23

      "24":
        env_var: JAVA24_HOME
        paths:
          - /usr/lib/jvm/bellsoft-java24-amd64
          - /usr/lib/jvm/bellsoft-java24
          - /usr/lib/jvm/liberica-jdk-24*
          - /usr/lib/jvm/liberica-jre-24*
          - /usr/lib/jvm/zulu-24*
          - /usr/lib/jvm/temurin-24*
          - /usr/lib/jvm/adoptopenjdk-24*
          - /usr/lib/jvm/jdk-24*
          - /usr/lib/jvm/java-24-openjdk
          - /usr/lib/jvm/openjdk-24*
          - /opt/java/bellsoft-java24-amd64
          - /opt/java/temurin-24*
          - /usr/local/java/bellsoft-java24-amd64
          - /usr/lib/jvm/java-arenadata-openjdk-24

services:
  TRINO:
      path: /etc/trino/conf/trino-java.yaml
  FLINK2:
      path: /etc/flink2/conf/flink2-java.yaml
  OZONE:
      path: /etc/ozone/conf/ozone-java.yaml
  SPARK4:
      path: /etc/spark4/conf/spark4-java.yaml
```

- Параметр ***Precheck packages***, активирующий возможность проверять наличие пакетов в репозитории перед установкой. Значение по умолчанию — `false`.
- ***Cluster UUID*** — отображает универсальный уникальный идентификатор кластера (universally unique identifier).
- Параметр ***Ranger plugins version***, позволяющий переопределить номер версии плагинов Ranger.

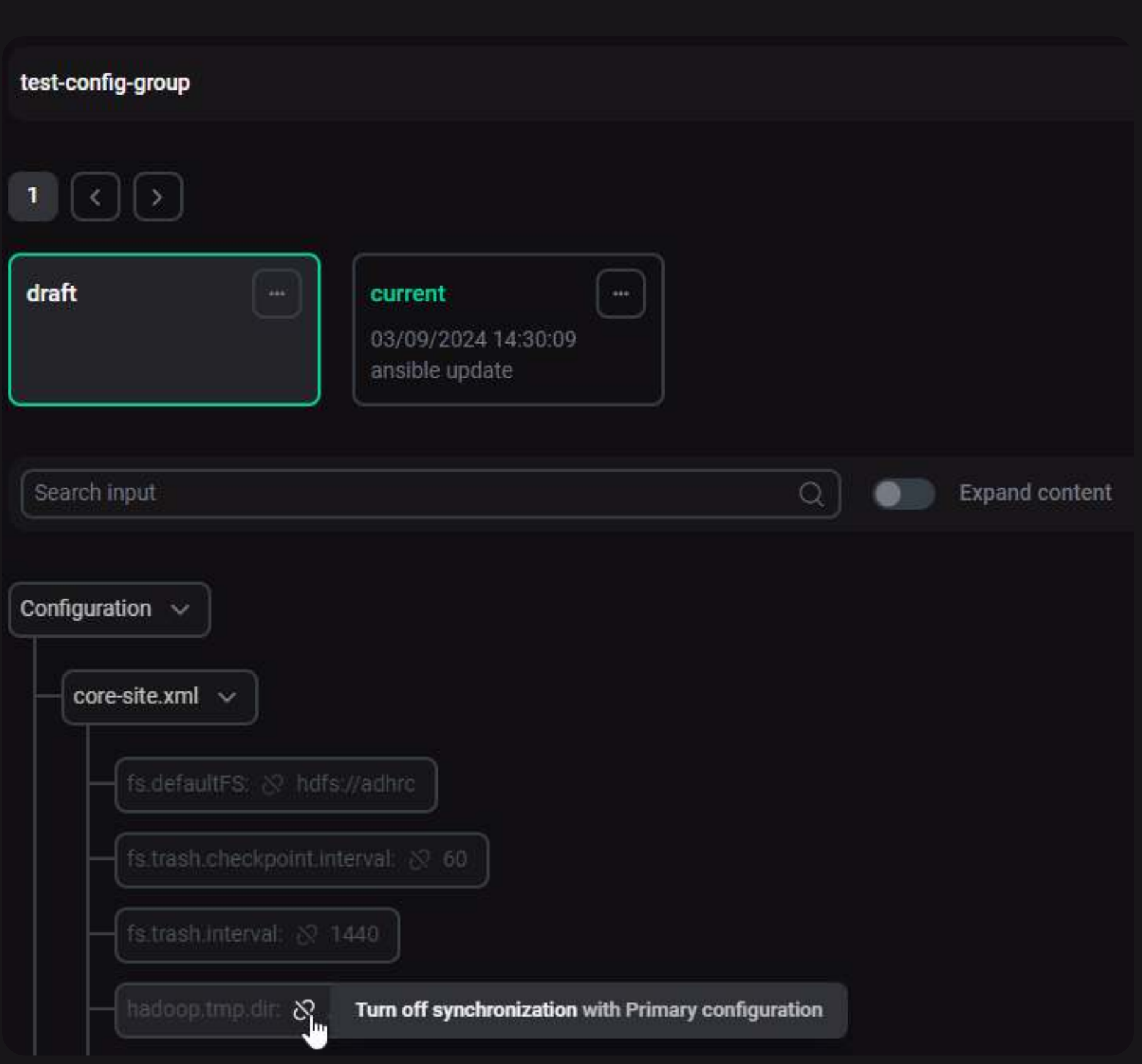


Конфигурирование кластера

Configuration groups

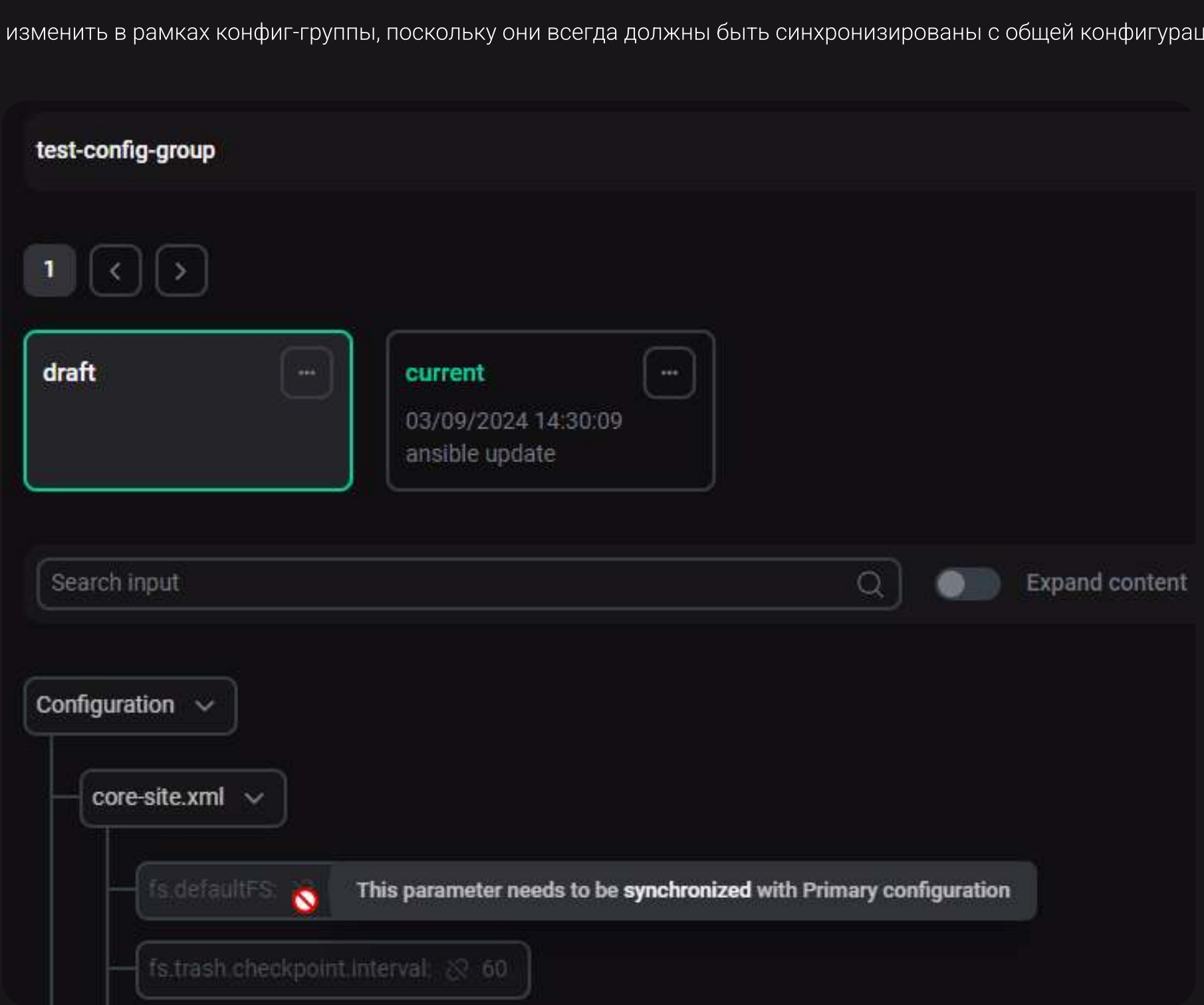
Вкладка ***Configuration*** → ***Configuration groups*** предназначена для настройки конфиг-групп кластера.

Чтобы настроить параметры в конфиг-группе, выберите параметры, которые вы хотите настроить иначе, нажмите иконку ***Turn off synchronization***, отредактируйте их и нажмите ***Save***.



Вкладка Configuration на странице конфиг-группы

Некоторые параметры нельзя изменить в рамках конфиг-группы, поскольку они всегда должны быть синхронизированы с общей конфигурацией.



Параметр, привязанный к общей конфигурации

Ansible settings

Вкладка ***Configuration*** → ***Ansible settings*** предназначена для ввода конфигурационных настроек Ansible на уровне кластера. Вкладка доступна начиная с версии ADCM 2.2.0.

Параметры Ansible

Имя	Описание	По умолчанию
forks	Число параллельных процессов, генерируемых при коммуникации с удаленными хостами	5

Установка кластера

Евгения Кузина, Дарья Барышева

Содержание

Назад к содержанию

- Шаг 1. Запуск установки
  - Установка всего кластера
  - Установка отдельных сервисов
- Шаг 2. Просмотр результатов установки
- Шаг 3. Проверка статуса кластера после установки

Шаг 1. Запуск установки

Существует два способа установки:

1. **Установка всего кластера**. В этом методе все сервисы кластера устанавливаются автоматически друг за другом после применения действия ***Install*** к кластеру.
2. **Установка отдельных сервисов**. В этом методе каждый сервис устанавливается вручную путем применения к нему одного или нескольких действий. Этот способ также можно использовать для добавления новых сервисов в уже развернутый кластер.

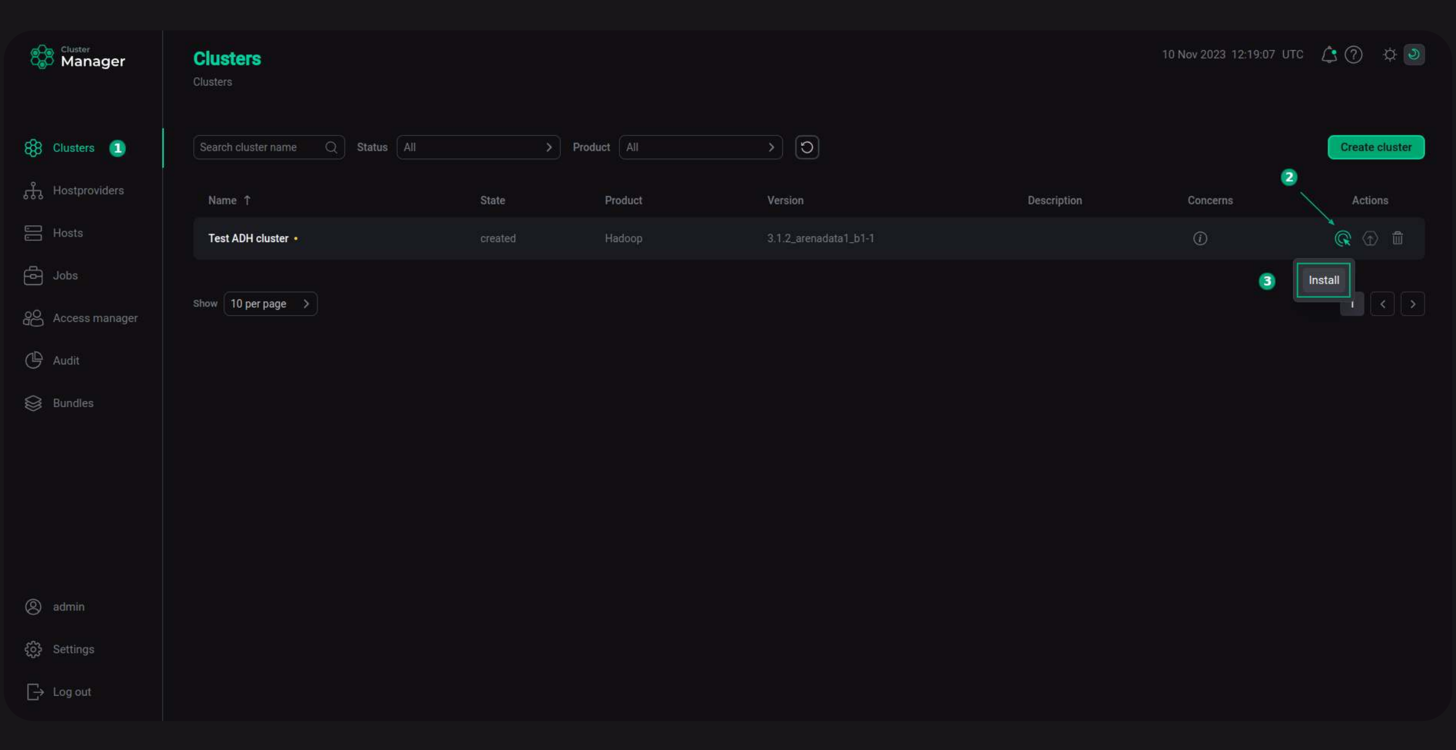
В обоих методах каждый сервис запускается автоматически после установки.

> **ВАЖНО**
>
> Установка должна производиться на чистые хосты без предустановленного подобного ПО.

Установка всего кластера

Чтобы установить все сервисы кластера за одну операцию, выполните шаги:

1. Примените действие ***Install*** к кластеру, выбранному на странице ***Clusters***, нажав на иконку в столбце ***Actions***.

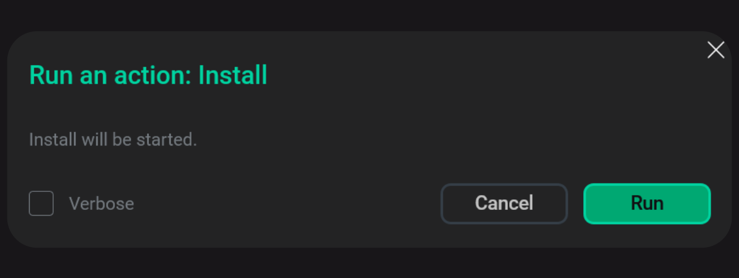


Cluster Manager
Clusters
Clusters
10 Nov 2023 12:19:07 UTC
Clusters
1
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
2
3
Install
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

Переход к установке всего кластера

2. Подтвердите действие в открывшемся окне.



Run an action: Install

Install will be started.

☐ Verbose

Cancel Run

Подтверждение действия

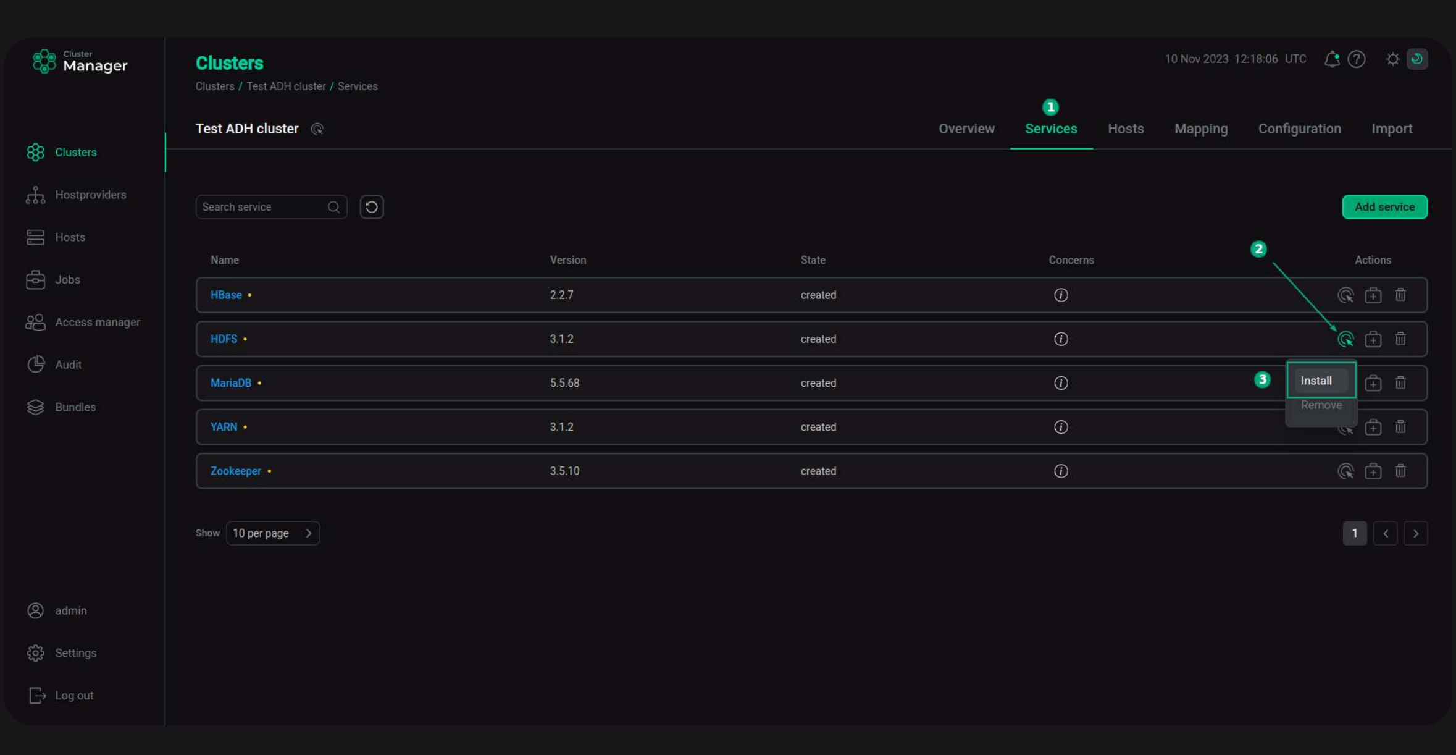
3. Дождитесь завершения установки.

Установка отдельных сервисов

Чтобы установить один или несколько сервисов вручную, необходимо предварительно добавить эти сервисы в кластер и затем применить к ним действия, описанные ниже.

> **ВНИМАНИЕ**
>
> Обратите внимание, что сервисы устанавливаются по очереди друг за другом. Не пытайтесь установить следующий сервис до окончания установки предыдущего.

1. Откройте вкладку ***Services*** на странице кластера. Для сервиса, который должен быть установлен, нажмите на иконку в столбце ***Actions*** и выберите действие ***Install*** из выпадающего списка.

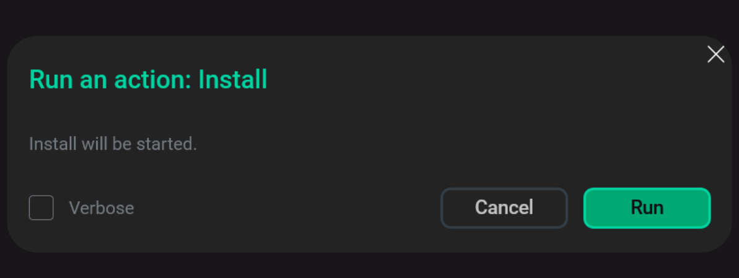


Cluster Manager
Clusters
Clusters / Test ADH cluster / Services
10 Nov 2023 12:18:06 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
1
2
3
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
Add service
Install
Remove
Show
10 per page

Name	Version	State	Concerns	Actions
HBase	2.2.7	created		
HDFS	3.1.2	created		
MariaDB	5.5.68	created		
YARN	3.1.2	created		
Zookeeper	3.5.10	created		

Переход к установке отдельных сервисов

2. Подтвердите действие в открывшемся окне.



Run an action: Install

Install will be started.

☐ Verbose

Cancel Run

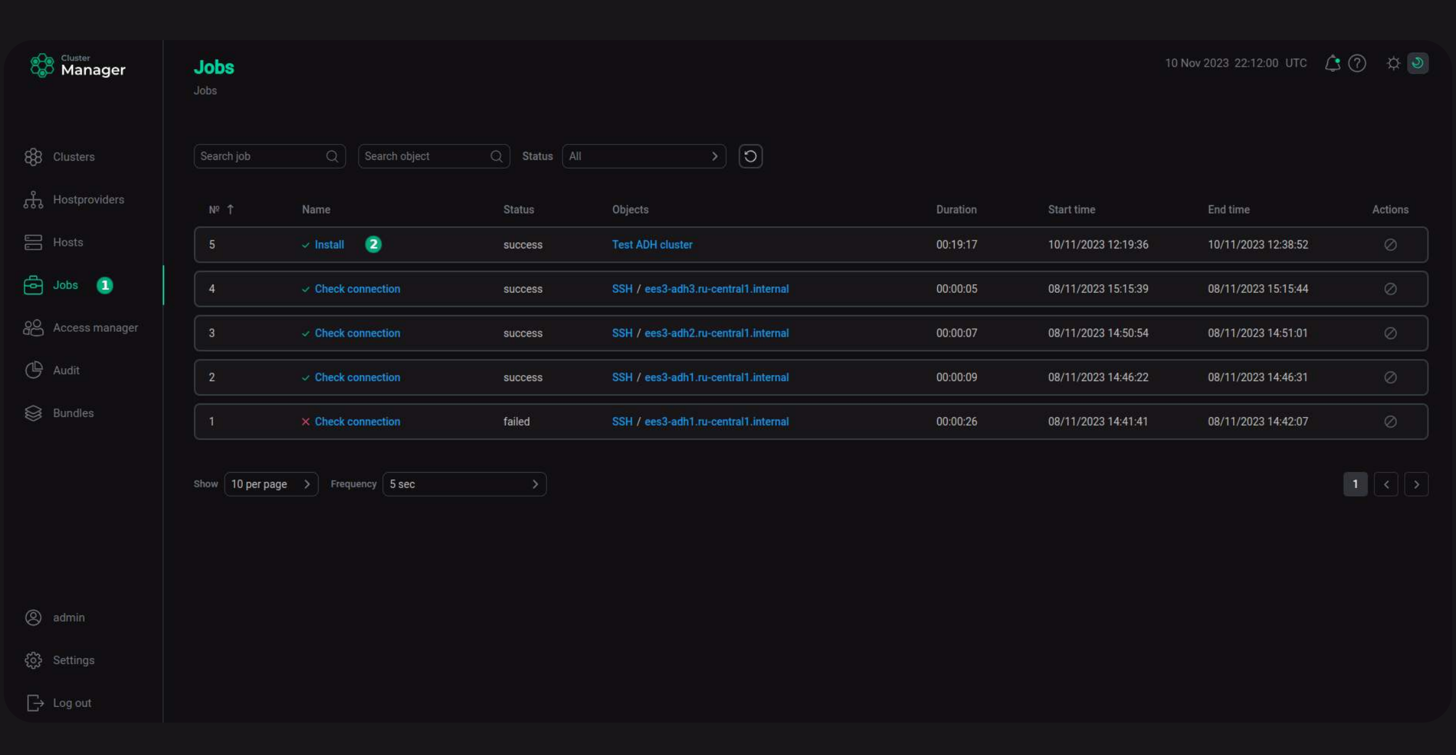
Подтверждение действия

3. Дождитесь завершения установки и убедитесь, что статус сервиса изменился с `created` на `installed`.
4. Повторите предыдущие шаги для всех сервисов, которые необходимо установить.

> **ВАЖНО**
>
> При установке сервисов вручную важно соблюдать следующий порядок:
>
> 1. Zookeeper
> 2. HDFS
> 3. YARN
> 4. HBase
> 5. Hive
> 6. Spark3
> 7. Monitoring

Шаг 2. Просмотр результатов установки

ADCM запускает отдельную задачу (job) для установки, как и для любого другого процесса. Для получения информации о статусе задач предназначена страница ***Jobs***.

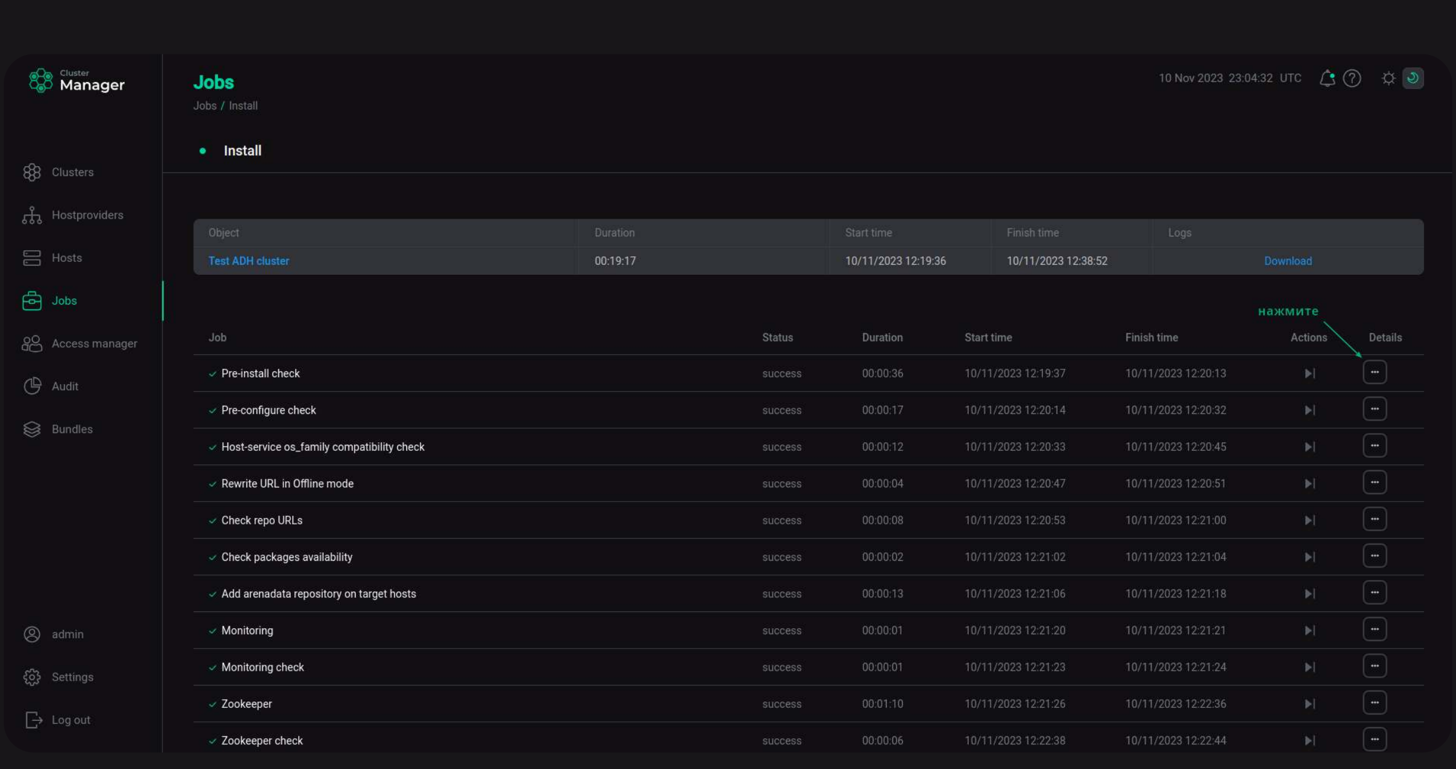


Cluster Manager
Jobs
Jobs
10 Nov 2023 22:12:00 UTC
Clusters
Hostproviders
Hosts
Jobs
1
2
Access manager
Audit
Bundles
admin
Settings
Log out
Search job
Search object
Status
All
Show
10 per page
Frequency
5 sec

№	Name	Status	Objects	Duration	Start time	End time	Actions
5	Install	success	Test ADH cluster	00:19:17	10/11/2023 12:19:36	10/11/2023 12:38:52	
4	Check connection	success	SSH / ees3-adh3.ru-central1.internal	00:00:05	08/11/2023 15:15:39	08/11/2023 15:15:44	
3	Check connection	success	SSH / ees3-adh2.ru-central1.internal	00:00:07	08/11/2023 14:50:54	08/11/2023 14:51:01	
2	Check connection	success	SSH / ees3-adh1.ru-central1.internal	00:00:09	08/11/2023 14:46:22	08/11/2023 14:46:31	
1	Check connection	failed	SSH / ees3-adh1.ru-central1.internal	00:00:26	08/11/2023 14:41:41	08/11/2023 14:42:07	

Страница Jobs

Если установка завершается успешно, соответствующая ей задача на странице ***Jobs*** переходит из статуса `running` в `success`; в противном случае — в статус `failed`. Чтобы получить более детальную информацию о возникших ошибках, можно перейти на страницу задачу, кликнув по ней в списке задач на странице ***Jobs***. Откроется окно с шагами выполнения задачи. Кликните многоточие в столбце ***Details***, чтобы получить дополнительную информацию об определенном шаге.



Cluster Manager
Jobs
Jobs / Install
10 Nov 2023 23:04:32 UTC
Install
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
нажмите

Object	Duration	Start time	Finish time	Logs
Test ADH cluster	00:19:17	10/11/2023 12:19:36	10/11/2023 12:38:52	Download

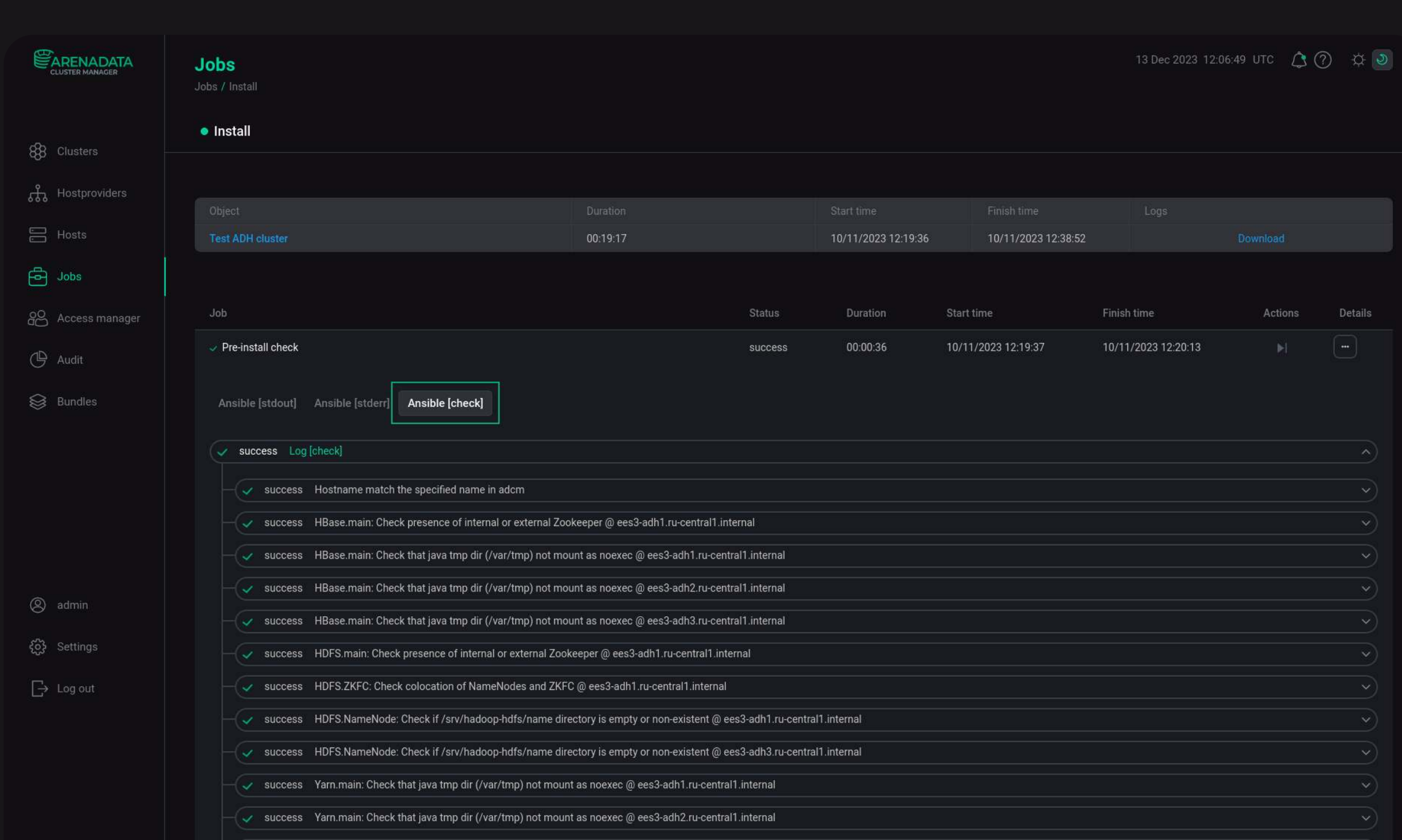
Job	Status	Duration	Start time	Finish time	Actions	Details
Pre-install check	success	00:00:36	10/11/2023 12:19:37	10/11/2023 12:20:13		
Pre-configure check	success	00:00:17	10/11/2023 12:20:14	10/11/2023 12:20:32		
Host-service os_family compatibility check	success	00:00:12	10/11/2023 12:20:33	10/11/2023 12:20:45		
Rewrite URL in Offline mode	success	00:00:04	10/11/2023 12:20:47	10/11/2023 12:20:51		
Check repo URLs	success	00:00:08	10/11/2023 12:20:53	10/11/2023 12:21:00		
Check packages availability	success	00:00:02	10/11/2023 12:21:02	10/11/2023 12:21:04		
Add arenadata repository on target hosts	success	00:00:13	10/11/2023 12:21:06	10/11/2023 12:21:18		
Monitoring	success	00:00:01	10/11/2023 12:21:20	10/11/2023 12:21:21		
Monitoring check	success	00:00:01	10/11/2023 12:21:23	10/11/2023 12:21:24		
Zookeeper	success	00:01:10	10/11/2023 12:21:26	10/11/2023 12:22:36		
Zookeeper check	success	00:00:06	10/11/2023 12:22:38	10/11/2023 12:22:44		

Переход на страницу с подробностями выполнения

Страница задачи содержит два основных раздела: ***ansible [stdout]*** и ***ansible [stderr]***. Это технические логи, содержащие информацию из стандартных I/O-потоков stdout/stderr.

В дополнение к ним может выводиться третий опциональный раздел ***ansible [check]***. Он содержит краткие описания наиболее часто возникающих ошибок. По сравнению с предыдущими разделами, эти описания, как правило, более простые и конкретные.

При возникновении ошибок в процессе установки рекомендуется изучить логи всех трех разделов.



ARENADATA CLUSTER MANAGER
Jobs
Jobs / Install
13 Dec 2023 12:06:49 UTC
Install
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Ansible [stdout]
Ansible [stderr]
Ansible [check]
Download

Object	Duration	Start time	Finish time	Logs
Test ADH cluster	00:19:17	10/11/2023 12:19:36	10/11/2023 12:38:52	Download

Job	Status	Duration	Start time	Finish time	Actions	Details
Pre-install check	success	00:00:36	10/11/2023 12:19:37	10/11/2023 12:20:13		

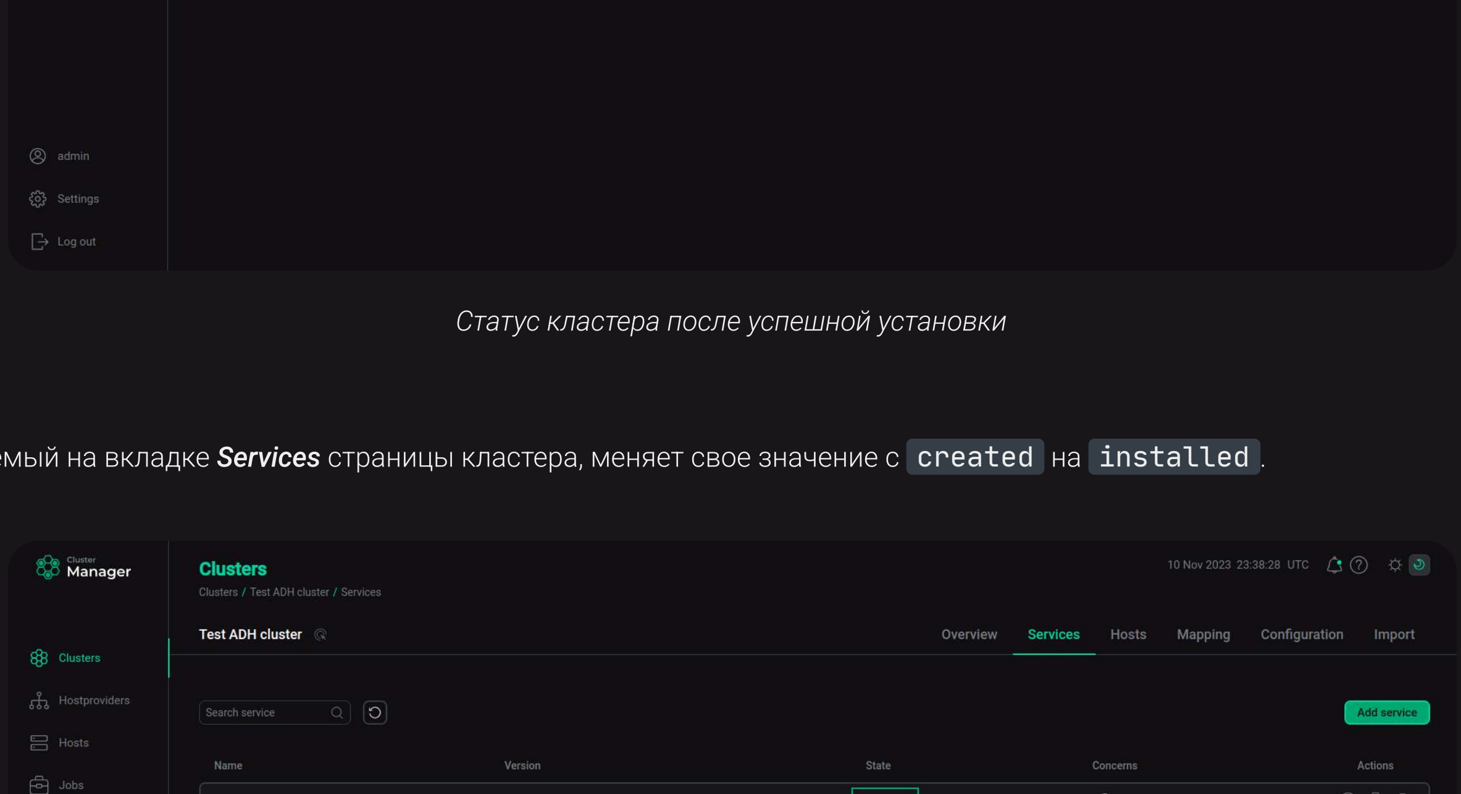
```
success Log [check]
  success Hostname match the specified name in adcm
  success HBase.main: Check presence of internal or external Zookeeper @ ees3-adh1.ru-central1.internal
  success HBase.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh1.ru-central1.internal
  success HBase.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh2.ru-central1.internal
  success HBase.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh3.ru-central1.internal
  success HDFS.main: Check presence of internal or external Zookeeper @ ees3-adh1.ru-central1.internal
  success HDFS.ZKFC: Check colocation of NameNodes and ZKFC @ ees3-adh1.ru-central1.internal
  success HDFS.NameNode: Check if /srv/hadoop-hdfs/name directory is empty or non-existent @ ees3-adh1.ru-central1.internal
  success HDFS.NameNode: Check if /srv/hadoop-hdfs/name directory is empty or non-existent @ ees3-adh3.ru-central1.internal
  success Yarn.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh1.ru-central1.internal
  success Yarn.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh2.ru-central1.internal
  success Yarn.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh3.ru-central1.internal
  success YARN.main: Enable HA ResourceManager @ ees3-adh2.ru-central1.internal
```

Подробности выполнения задачи

Шаг 3. Проверка статуса кластера после установки

В результате успешной установки происходит обновление статуса кластера и его сервисов:

- Статус кластера, отображаемый в столбце ***State*** на странице ***Clusters***, меняет свое значение с `created` на `installed`.

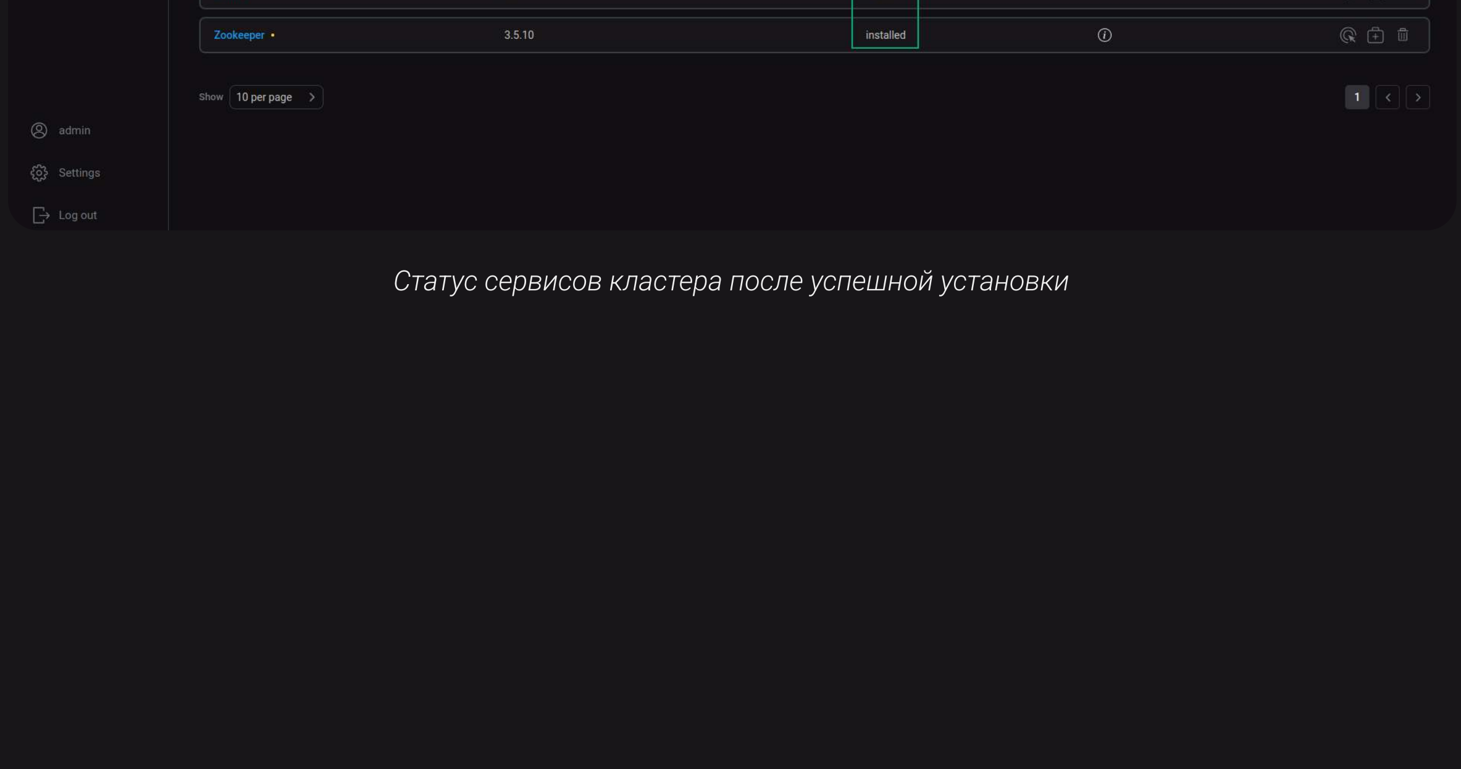


Cluster Manager
Clusters
Clusters
10 Nov 2023 23:29:12 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1			

Статус кластера после успешной установки

- Статус сервисов, отображаемый на вкладке ***Services*** страницы кластера, меняет свое значение с `created` на `installed`.



Cluster Manager
Clusters
Clusters / Test ADH cluster / Services
10 Nov 2023 23:38:28 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
Add service
Show
10 per page

Name	Version	State	Concerns	Actions
HBase	2.2.7	installed		
HDFS	3.1.2	installed		
Hive	3.1.3_arenadata4	installed		
MariaDB	5.5.68	installed		
YARN	3.1.2	installed		
Zookeeper	3.5.10	installed		

Статус сервисов кластера после успешной установки

Установка мониторинга

Дарья Барышева

Назад к содержанию

Мониторинг — это опциональный, но рекомендуемый компонент ADH, который можно настроить, используя один из следующих подходов:

- добавить в кластер ADH сервис Monitoring (доступен, начиная с версии ADH 4.0.0);
- создать кластер мониторинга и интегрировать его с кластером ADH (поддержка этого способа установки мониторинга будет прекращена в будущих релизах ADH).

Сервис мониторинга

Елена Дворядкина, Елена Костюченко

Содержание

Назад к содержанию

Обзор

Шаг 1. Добавление сервиса

Шаг 2. Добавление компонентов

Шаг 3. Настройка сервиса

Шаг 4. Установка сервиса

Шаг 5. Проверка результатов

- Метрики в формате Prometheus
- Веб-интерфейс Prometheus
- Веб-интерфейс Grafana

Сервис Monitoring разворачивает в ADH свой сервер Prometheus, который реализует сбор и хранение метрик мониторинга кластера ADH, а также поддерживает возможность использования веб-интерфейса Grafana для визуализации и анализа информации. В данной статье описываются шаги, необходимые для подключения этого сервиса.

Обзор

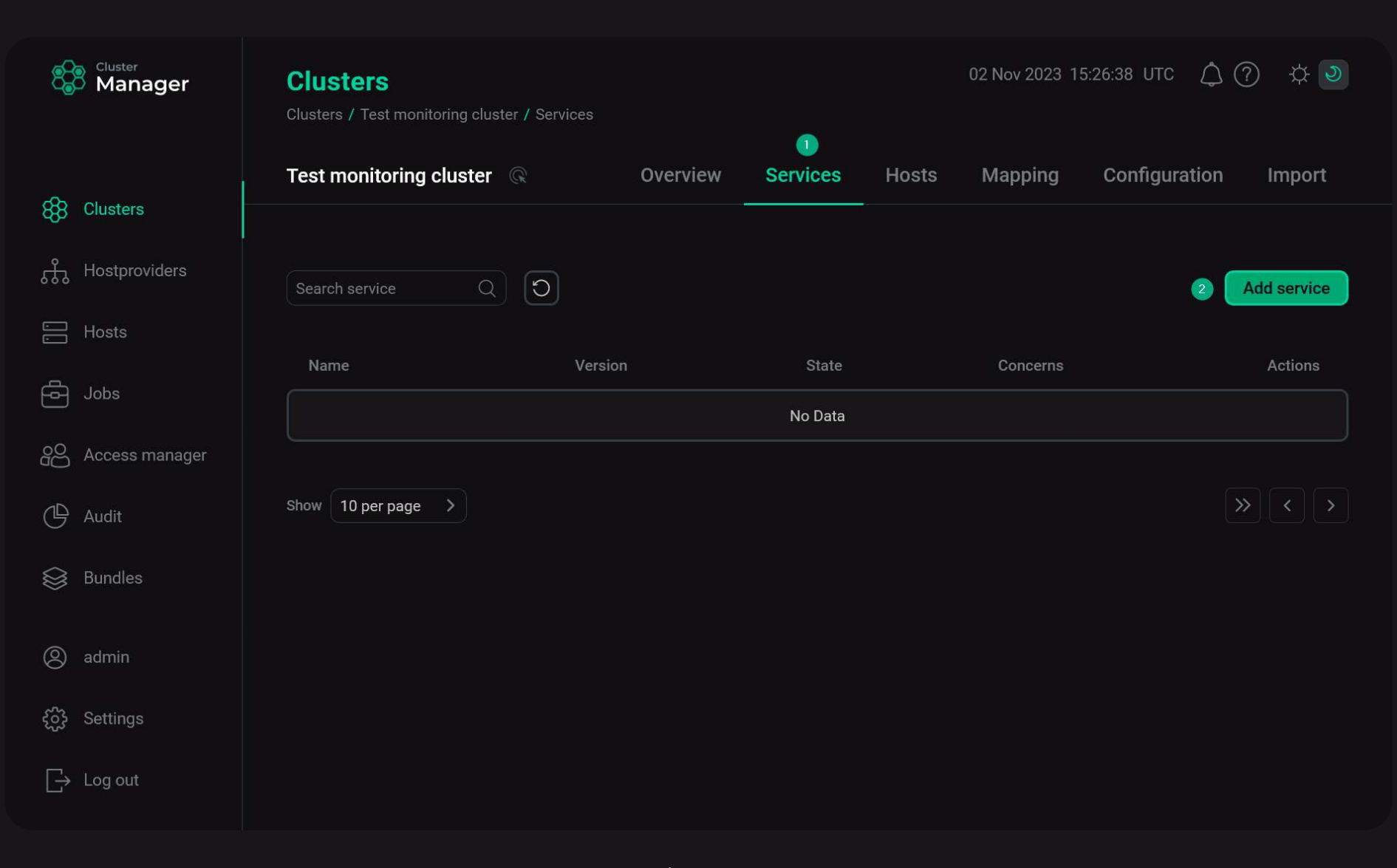
При добавлении сервиса Monitoring в ADH на все или выборочные хосты кластера можно установить Node Exporter — агент мониторинга, считывающий с хоста системные метрики, которые будет собирать Prometheus. Prometheus также может собирать метрики сервисов ADH. Эти метрики будут доступны в формате Prometheus на портах и конечных точках (endpoints), указанных в параметрах сервисных конфигураций. Просматривать и анализировать данные, которые собирает сервис мониторинга, можно также в веб-интерфейсе Prometheus или Grafana.

ПРИМЕЧАНИЕ

- Если у вас уже настроена совместимая с Prometheus система мониторинга (например, свой сервер Prometheus или VictoriaMetrics), можно использовать ее, чтобы собирать метрики ADH. Для этого в своей системе мониторинга укажите настройки доступа к метрикам ADH, используя параметры установленного в ADH Prometheus, которые конфигурируются на странице сервиса Monitoring в интерфейсе ADCM.
- Вы также можете использовать механизм Federation, чтобы перенести все метрики из сервера Prometheus, развернутого в ADH, в свой Prometheus.

Шаг 1. Добавление сервиса

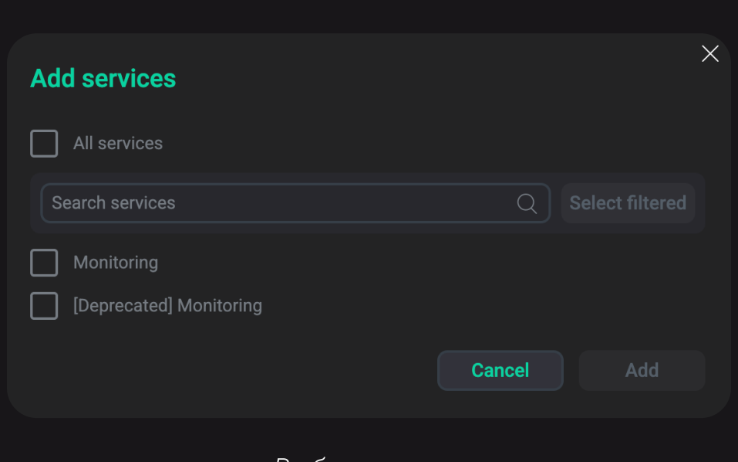
1. В интерфейсе ADCM откройте страницу *Clusters* и кликните по имени кластера ADH. Затем на открывшейся странице кластера перейдите на вкладку *Services* и нажмите *Add services*.



Cluster Manager
Clusters
Clusters / Test monitoring cluster / Services
02 Nov 2023 15:26:38 UTC
Test monitoring cluster
Overview
Services
Hosts
Mapping
Configuration
Import
1
2
Search service
Add service
Name
Version
State
Concerns
Actions
No Data
Show 10 per page
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out

Переход к добавлению сервисов

2. В открывшемся окне выберите сервис *Monitoring* и нажмите *Add*.



Add services
All services
Search services
Select filtered
Cancel
Add

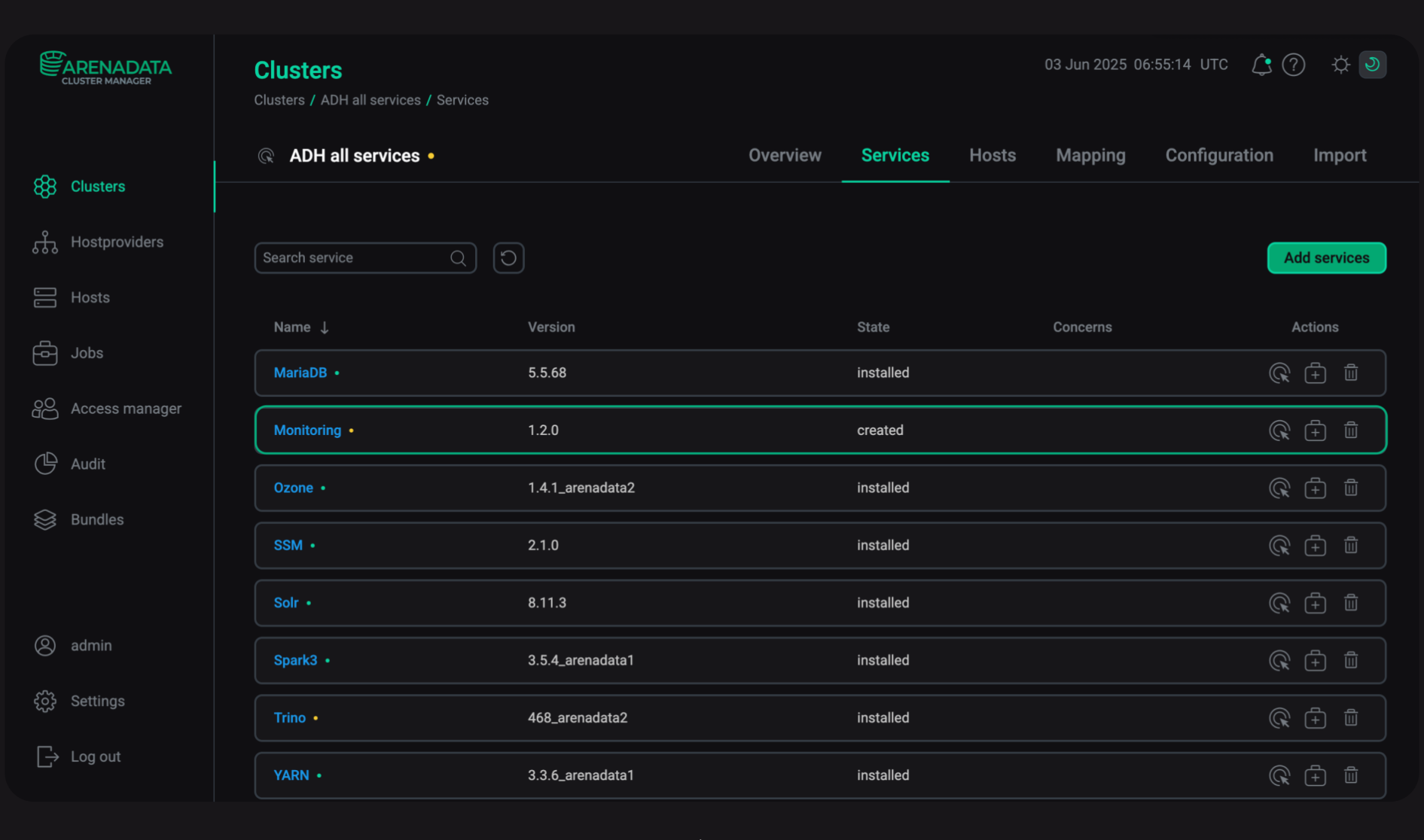
☐ All services

☐ Monitoring

☐ [Deprecated] Monitoring

Выбор сервиса

В результате добавленный сервис отобразится на вкладке *Services*.



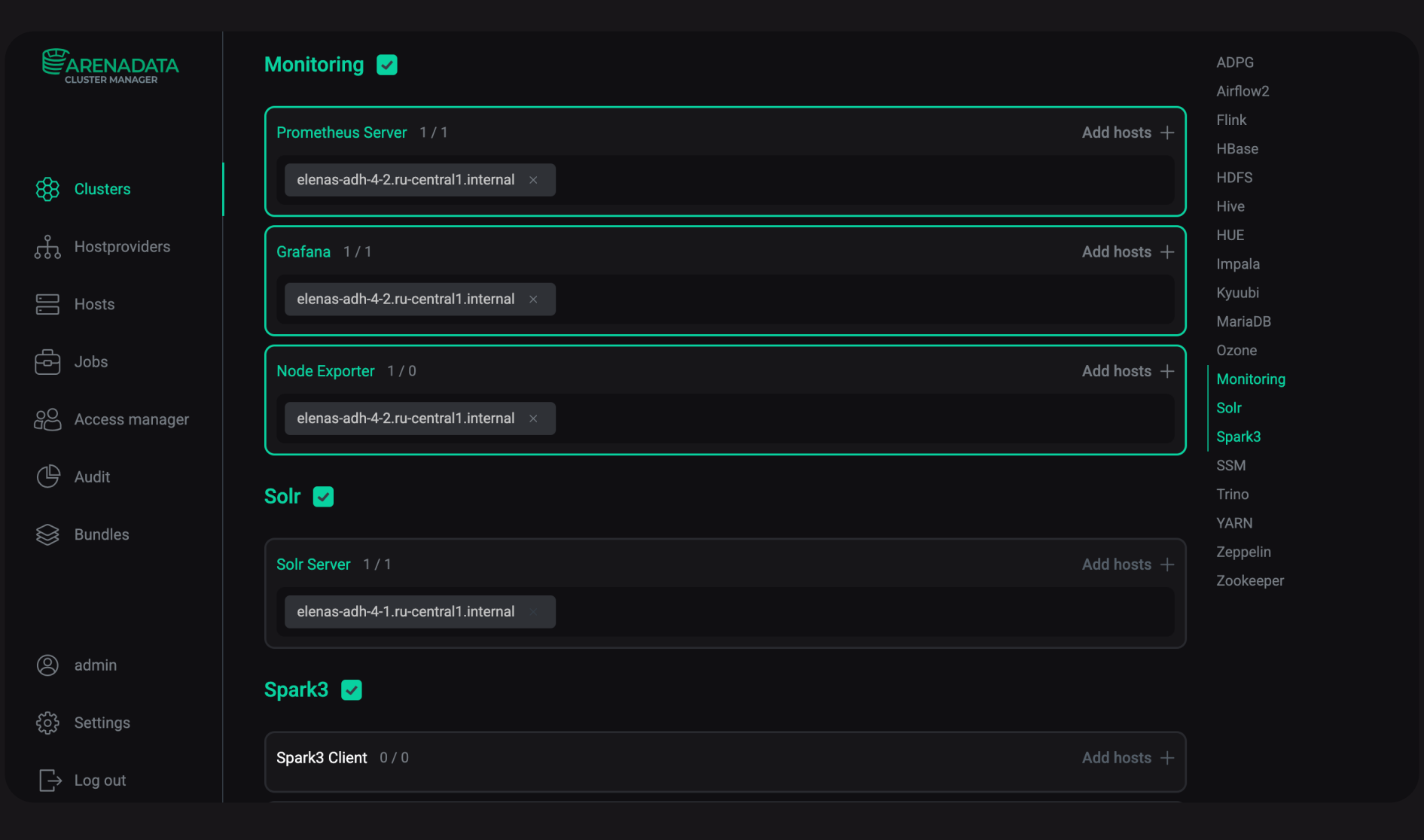
ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH all services / Services
03 Jun 2025 06:55:14 UTC
ADH all services
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add services
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out

Name	Version	State	Concerns	Actions
MariaDB	5.5.68	installed		
Monitoring	1.2.0	created		
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Solr	8.11.3	installed		
Spark3	3.5.4_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

Результат успешного добавления сервиса в кластер

Шаг 2. Добавление компонентов

1. На странице кластера откройте вкладку *Mapping*, чтобы перейти к распределению сервисных компонентов между хостами кластера.



ARENADATA CLUSTER MANAGER
Monitoring
Solr
Spark3
Add hosts
ADPG
Airflow2
Flink
HBase
HDFS
Hive
HUE
Impala
Kyuubi
MariaDB
Ozone
Monitoring
Solr
Spark3
SSM
Trino
YARN
Zeppelin
Zookeeper
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out

Prometheus Server 1 / 1 — elenas-adh-4-2.ru-central1.internal

Grafana 1 / 1 — elenas-adh-4-2.ru-central1.internal

Node Exporter 1 / 0 — elenas-adh-4-2.ru-central1.internal

Solr Server 1 / 1 — elenas-adh-4-1.ru-central1.internal

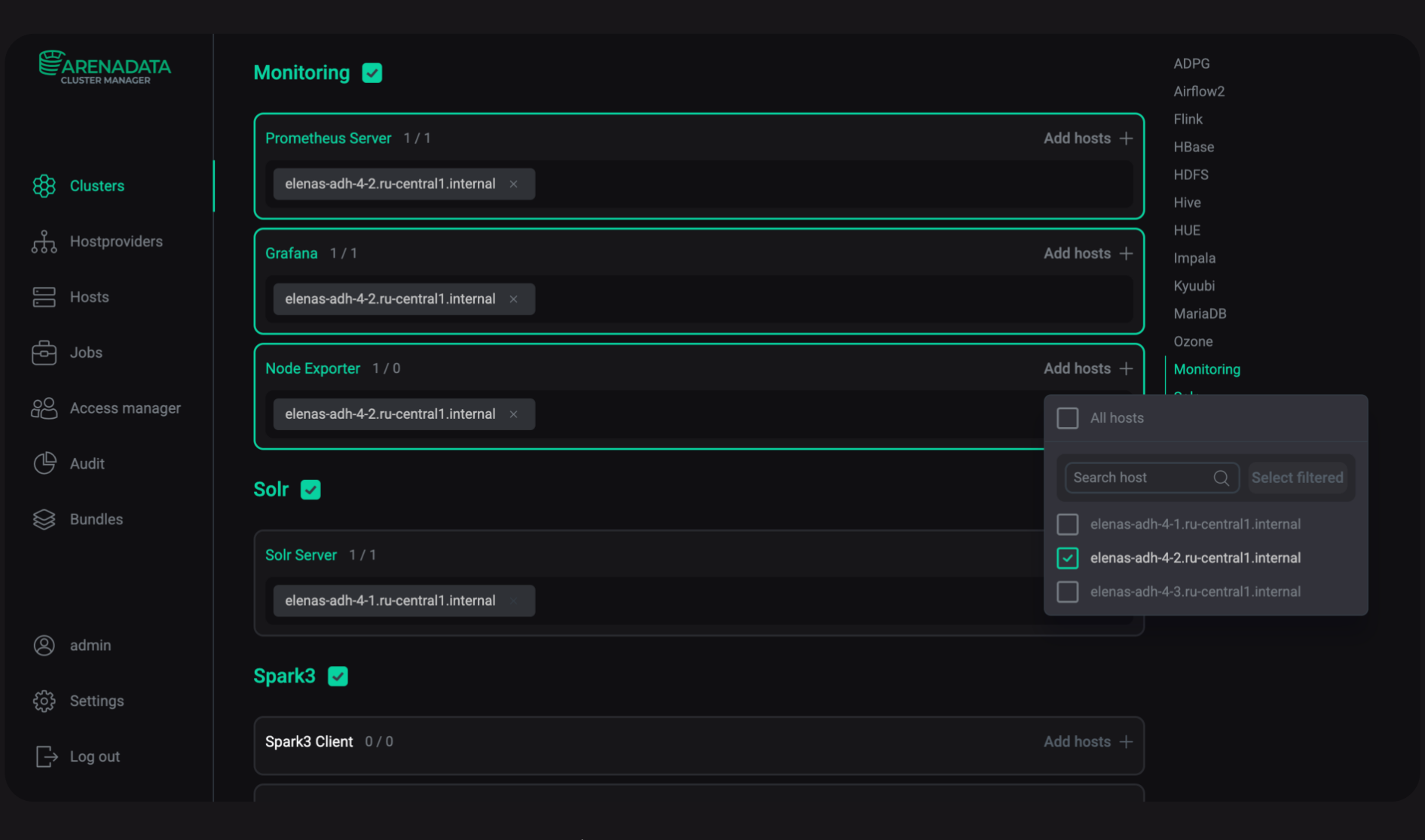
Spark3 Client 0 / 0

Переход к распределению сервисных компонентов

Компоненты сервиса Monitoring

Компонент	Описание
Prometheus Server	Хранит все метрики кластера ADH и позволяет Grafana запрашивать их
Grafana	Позволяет визуализировать метрики ADH в виде графиков и диаграмм, организованных в дашборды
Node Exporter	Собирает системные метрики и отправляет их серверу Prometheus

2. Назначьте хосты компонентам сервиса Monitoring — нажмите *Add hosts* и выберите нужные хосты в открывшемся окне.



ARENADATA CLUSTER MANAGER
Monitoring
Prometheus Server 1 / 1
Grafana 1 / 1
Node Exporter 1 / 0
Solr
Solr Server 1 / 1
Spark3
Spark3 Client 0 / 0
Add hosts
All hosts
Search host
Select filtered
elenas-adh-4-1.ru-central1.internal
elenas-adh-4-2.ru-central1.internal
elenas-adh-4-3.ru-central1.internal

Выбор хоста для компонента

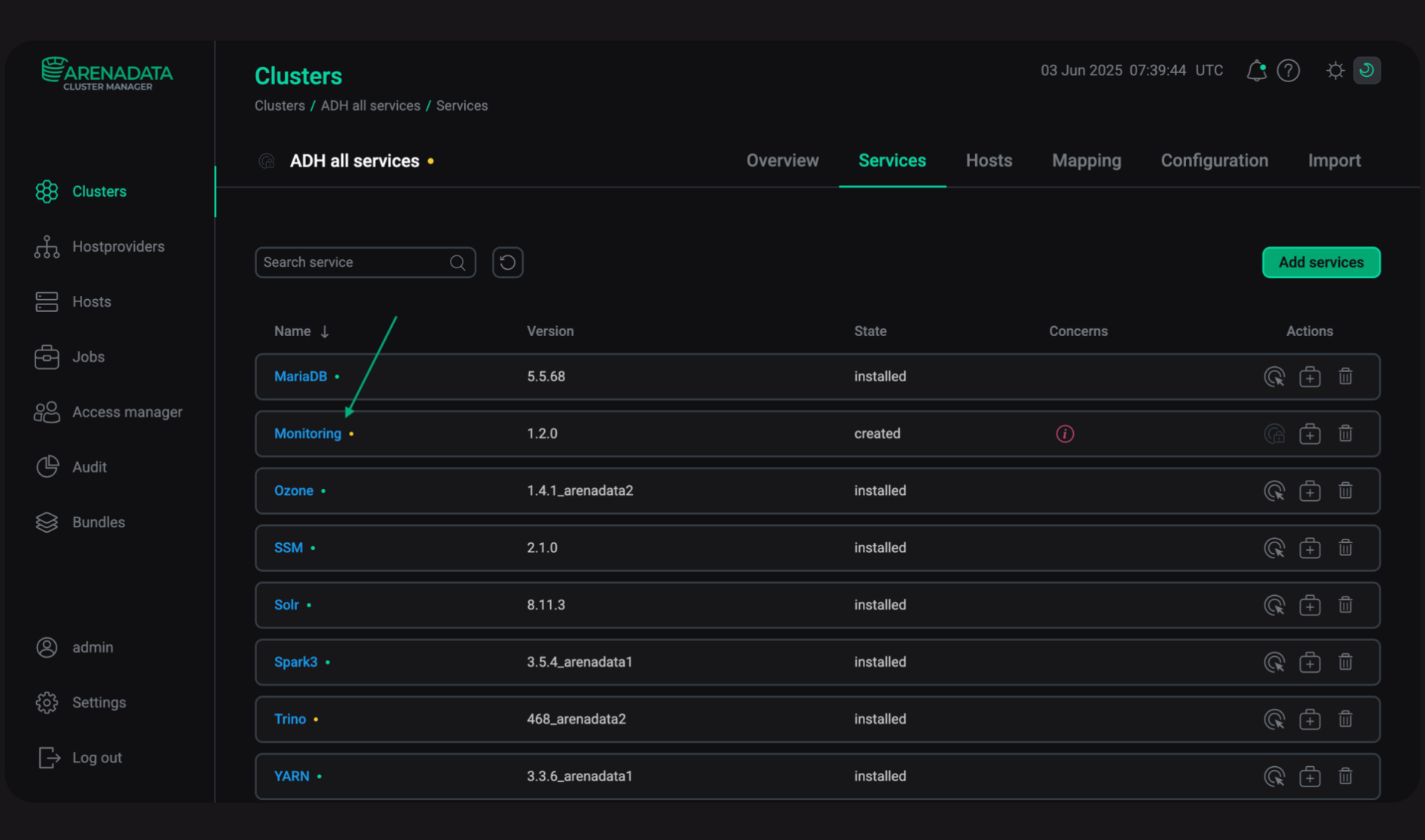
ВНИМАНИЕ

Компонент Prometheus Server не рекомендуется устанавливать на хосты с ADH — используйте для него отдельные хосты. Иначе при выходе из строя хоста ADH и/или критически большой нагрузке на него информация о соответствующих проблемах не будет сохранена.

3. После завершения распределения компонентов нажмите *Save*, чтобы сохранить изменения.

Шаг 3. Настройка сервиса

1. Откройте вкладку *Services* на странице кластера и кликните по имени сервиса *Monitoring* в столбце *Name*.

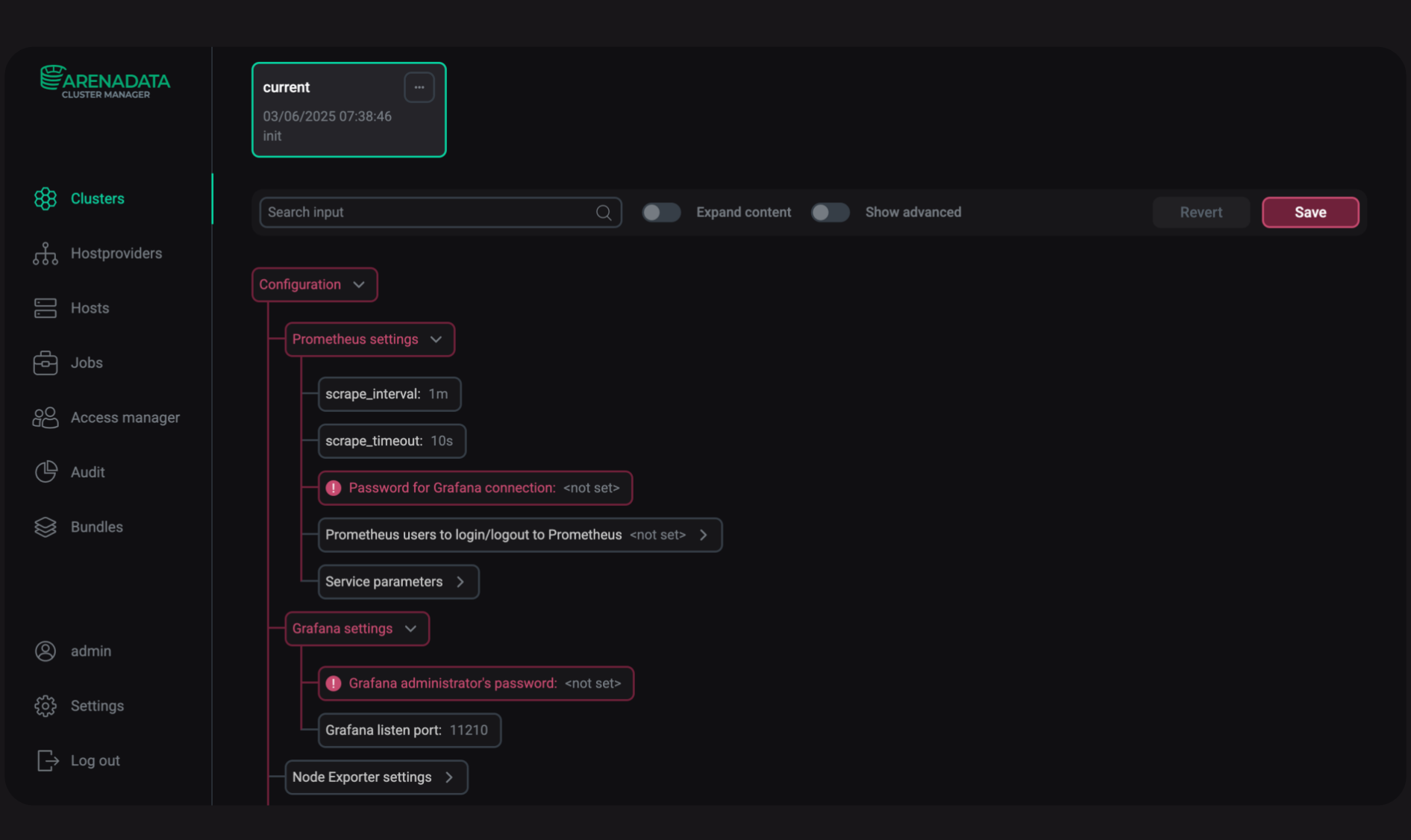


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH all services / Services
03 Jun 2025 07:39:44 UTC
ADH all services
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add services

Name	Version	State	Concerns	Actions
MariaDB	5.5.68	installed		
Monitoring	1.2.0	created	ⓘ	
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Solr	8.11.3	installed		
Spark3	3.5.4_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

Переход к настройке сервиса

2. В открывшемся окне заполните конфигурационные параметры сервиса — описания параметров приведены в разделе Monitoring. Поля, подсвеченные красным, обязательны для заполнения.



ARENADATA CLUSTER MANAGER
current
03/06/2025 07:38:46
init
Search input
Expand content
Show advanced
Revert
Save
Configuration
Prometheus settings
Grafana settings
Node Exporter settings

scrape_interval: 1m

scrape_timeout: 10s

Password for Grafana connection: <not set>

Prometheus users to login/logout to Prometheus <not set>

Service parameters

Grafana administrator's password: <not set>

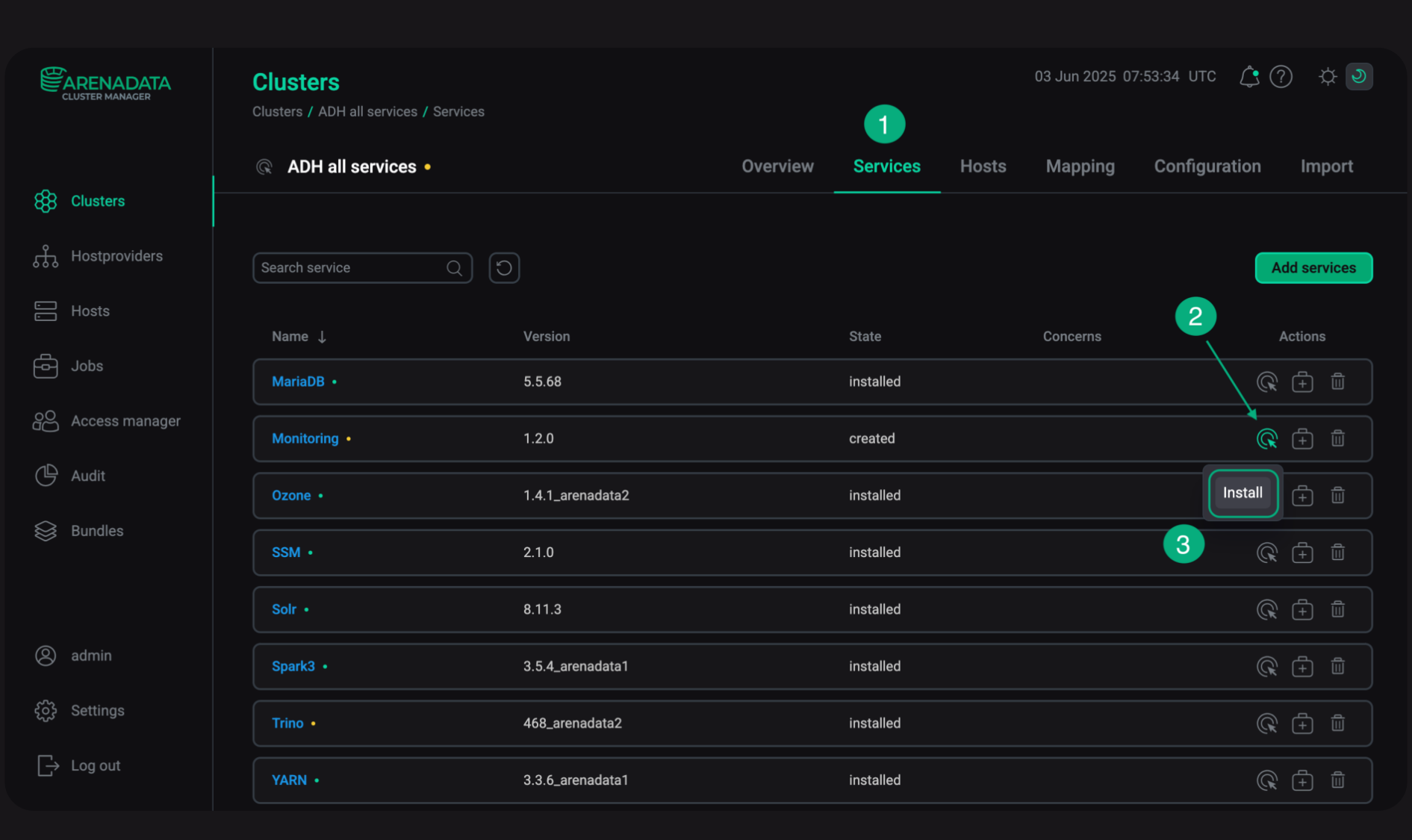
Grafana listen port: 11210

Настройка сервиса Monitoring

После указания всех необходимых параметров нажмите *Save*.

Шаг 4. Установка сервиса

1. На вкладке *Services* в строке сервиса Monitoring кликните по иконке в столбце *Actions* и запустите действие *Install*.

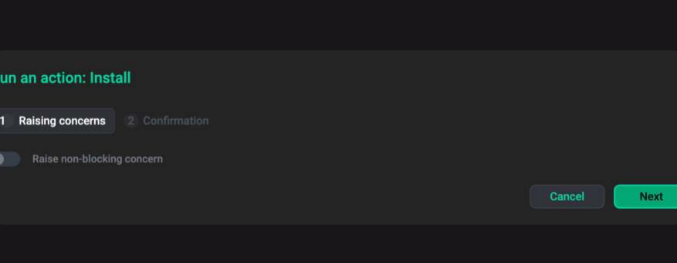


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH all services / Services
03 Jun 2025 07:53:34 UTC
ADH all services
Overview
Services
Hosts
Mapping
Configuration
Import
1
2
3
Install
Search service
Add services

Name	Version	State	Concerns	Actions
MariaDB	5.5.68	installed		
Monitoring	1.2.0	created		
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Solr	8.11.3	installed		
Spark3	3.5.4_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

Переход к установке сервиса

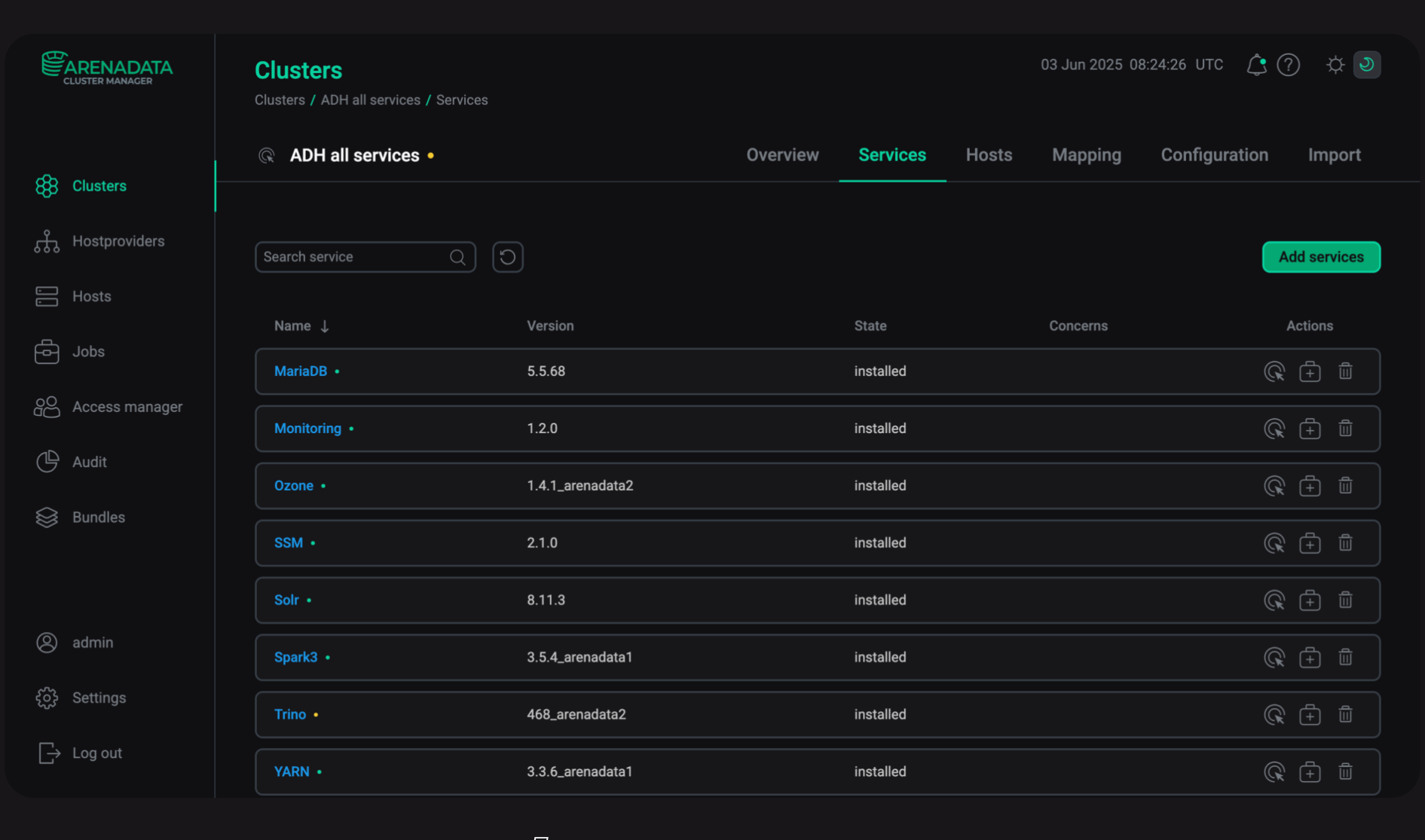
2. Подтвердите действие в открывшемся окне.



Run an action: Install
1 Raising concerns
2 Confirmation
Raise non-blocking concern
Cancel
Next

Подтверждение действия

3. Дождитесь завершения установки и убедитесь, что статус сервиса изменился с `created` на `installed`.

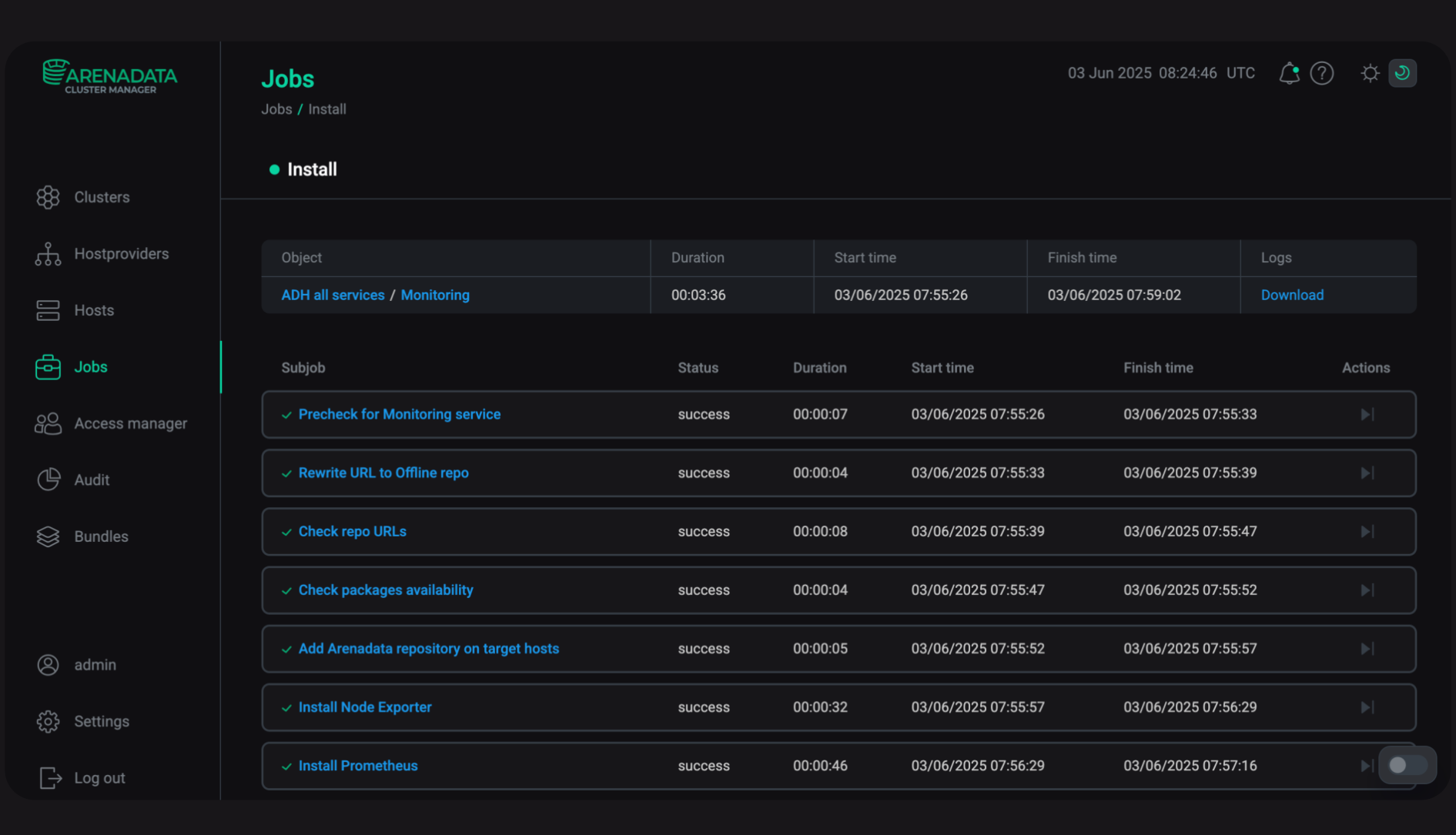


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH all services / Services
03 Jun 2025 08:24:26 UTC
ADH all services
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add services

Name	Version	State	Concerns	Actions
MariaDB	5.5.68	installed		
Monitoring	1.2.0	installed		
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Solr	8.11.3	installed		
Spark3	3.5.4_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

Процесс установки завершен

Чтобы посмотреть процесс выполнения установки сервиса и проанализировать ошибки в случае их возникновения, выберите пункт *Jobs* в левом навигационном меню и кликните по имени задачи *Install* в списке задач *Jobs*.



ARENADATA CLUSTER MANAGER
Jobs
Jobs / Install
03 Jun 2025 08:24:46 UTC
Install

Object	Duration	Start time	Finish time	Logs
ADH all services / Monitoring	00:03:36	03/06/2025 07:55:26	03/06/2025 07:59:02	Download

Subjob	Status	Duration	Start time	Finish time	Actions
✓ Precheck for Monitoring service	success	00:00:07	03/06/2025 07:55:26	03/06/2025 07:55:33	
✓ Rewrite URL to Offline repo	success	00:00:04	03/06/2025 07:55:33	03/06/2025 07:55:39	
✓ Check repo URLs	success	00:00:08	03/06/2025 07:55:39	03/06/2025 07:55:47	
✓ Check packages availability	success	00:00:04	03/06/2025 07:55:47	03/06/2025 07:55:52	
✓ Add Arenadata repository on target hosts	success	00:00:05	03/06/2025 07:55:52	03/06/2025 07:55:57	
✓ Install Node Exporter	success	00:00:32	03/06/2025 07:55:57	03/06/2025 07:56:29	
✓ Install Prometheus	success	00:00:46	03/06/2025 07:56:29	03/06/2025 07:57:16	

Страница задачи Install

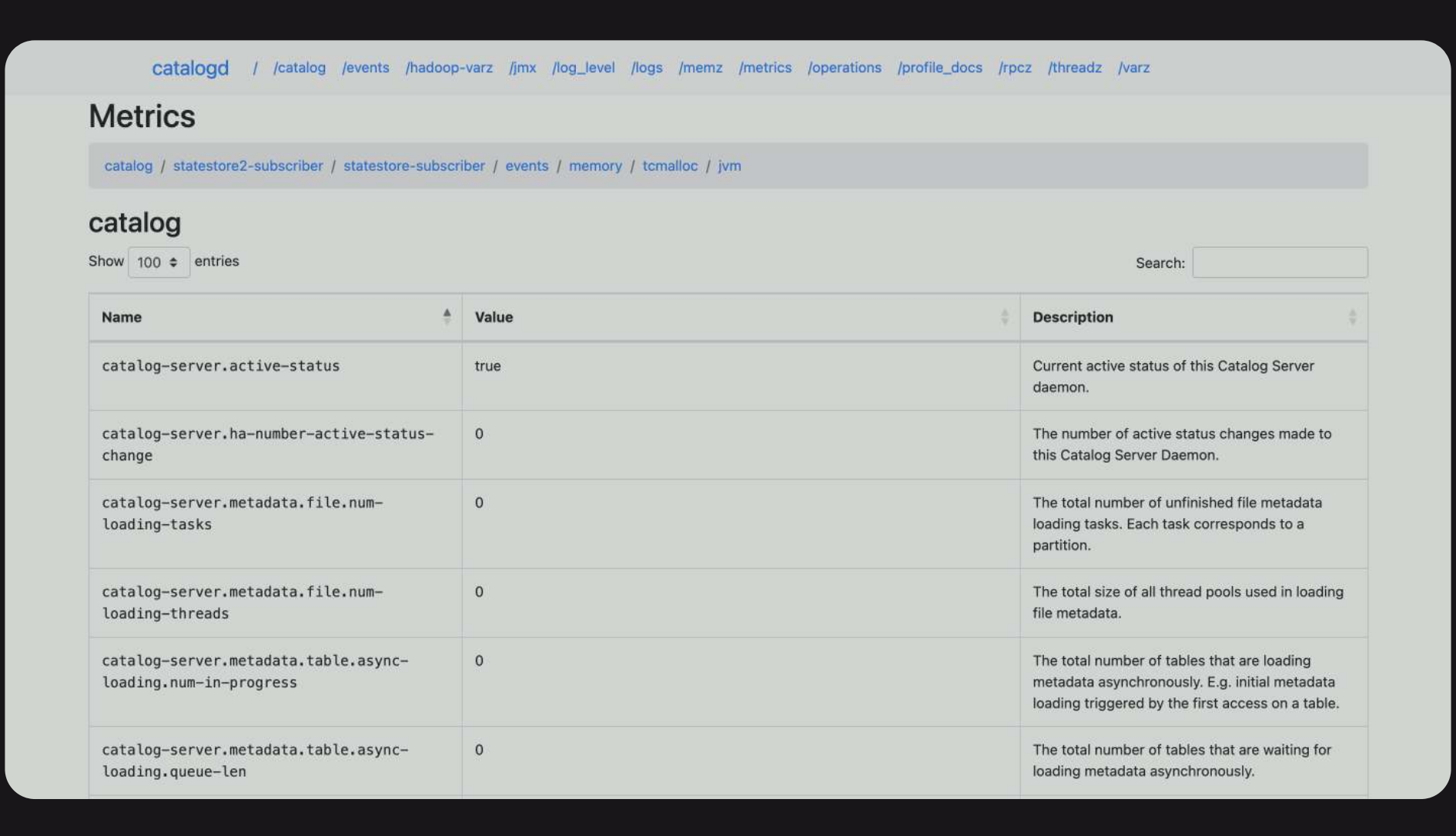
Шаг 5. Проверка результатов

Сервис Monitoring запускается автоматически после установки. По завершении установки необходимо убедиться, что с хостов кластера поступают системные метрики и метрики сервисов ADH. Чтобы это проверить, можно в браузере посмотреть метрики в формате Prometheus, а также использовать веб-интерфейсы Prometheus и Grafana.

Метрики в формате Prometheus

1. В адресной строке браузера введите адрес хоста кластера ADH, а также укажите порт и конечную точку доступа (endpoint) к сервисным или системным метрикам, которые определяются параметрами:
   - в конфигурации сервисов ADH — настройки доступа к метрикам мониторинга этих сервисов;
   - в секции **Node Exporter settings** на странице сервиса Monitoring — настройки доступа к системным метрикам хоста кластера ADH.

   Пример адреса для просмотра метрик сервера Impala на хосте с IP `10.92.42.57`: *http://10.92.42.57:25020/metrics*.
2. На открывшейся странице будут показаны метрики мониторинга с указанного хоста кластера в формате Prometheus.



catalogd / /catalog /events /hadoop-varz /jmx /log_level /logs /memz /metrics /operations /profile_docs /rpcz /threadz /varz
catalog / statestore2-subscriber / statestore-subscriber / events / memory / tcmalloc / jvm
Show 100 entries
Search:

Metrics

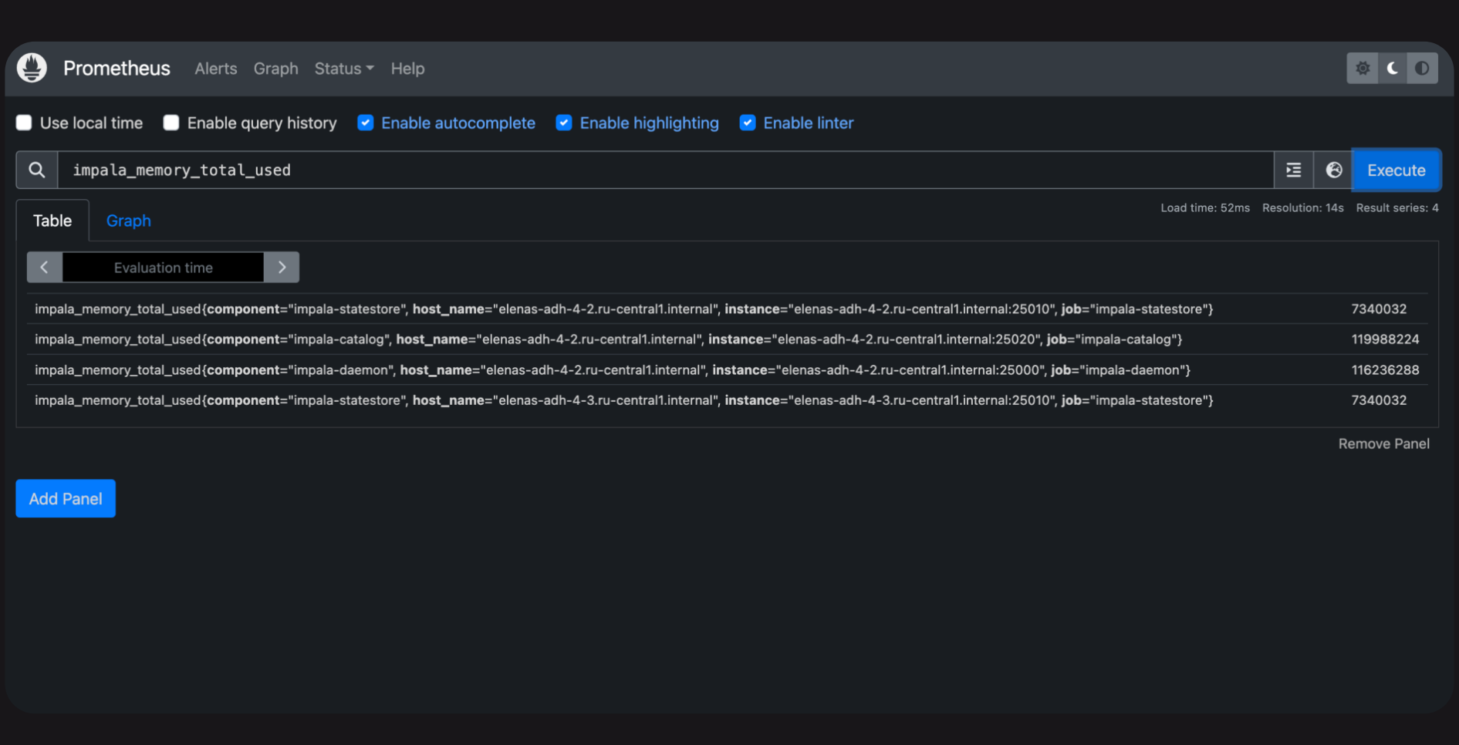
catalog

Name	Value	Description
catalog-server.active-status	true	Current active status of this Catalog Server daemon.
catalog-server.ha-number-active-status-change	0	The number of active status changes made to this Catalog Server Daemon.
catalog-server.metadata.file.num-loading-tasks	0	The total number of unfinished file metadata loading tasks. Each task corresponds to a partition.
catalog-server.metadata.file.num-loading-threads	0	The total size of all thread pools used in loading file metadata.
catalog-server.metadata.table.async-loading.num-in-progress	0	The total number of tables that are loading metadata asynchronously. E.g. initial metadata loading triggered by the first access on a table.
catalog-server.metadata.table.async-loading.queue-len	0	The total number of tables that are waiting for loading metadata asynchronously.

Метрики сервера Impala в формате Prometheus

Веб-интерфейс Prometheus

1. В адресной строке браузера введите IP-адрес хоста, где установлен компонент Prometheus Server сервиса Monitoring. Укажите номер порта, который определяется параметром **web.listen-address** в секции **Prometheus settings** в группе **Service parameters** конфигурации сервиса Monitoring (по умолчанию, `11200`). Пример адреса: *http://10.92.40.107:11200*. Для входа в интерфейс Prometheus используйте имя пользователя и пароль, которые также указываются в секции **Prometheus settings** — настройка **Prometheus users to login/logout to Prometheus**.
2. В поле *Expression* можно ввести название метрики и нажать *Execute* — в интерфейсе будут показаны значения этой метрики на всех хостах кластера ADH.



Prometheus
Alerts
Graph
Status
Help
Use local time
Enable query history
Enable autocomplete
Enable highlighting
Enable linter
Execute
Table
Graph
Evaluation time
Remove Panel
Add Panel

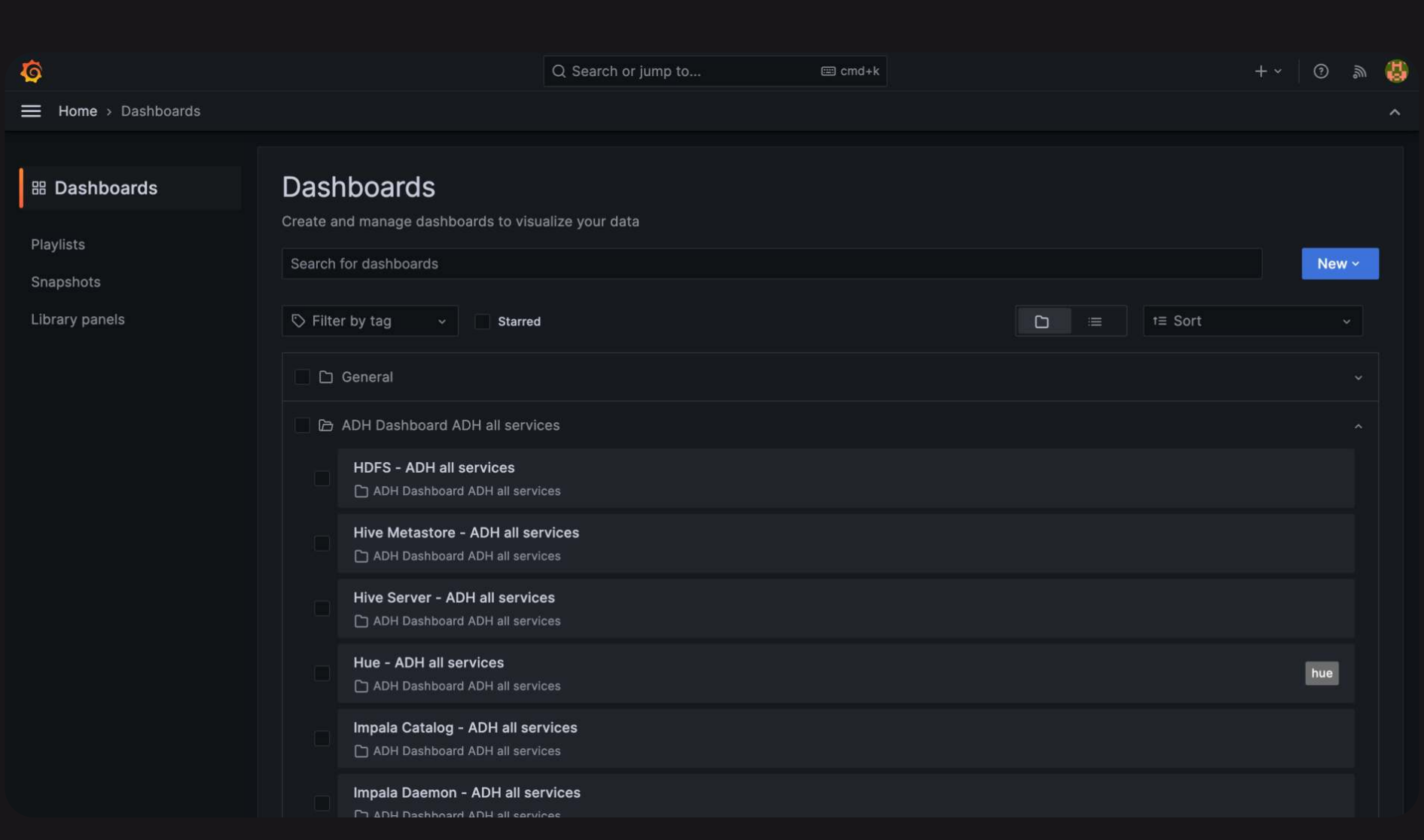
`impala_memory_total_used`

```
Load time: 52ms   Resolution: 14s   Result series: 4
impala_memory_total_used{component="impala-statestore", host_name="elenas-adh-4-2.ru-central1.internal", instance="elenas-adh-4-2.ru-central1.internal:25010", job="impala-statestore"}   7340032
impala_memory_total_used{component="impala-catalog", host_name="elenas-adh-4-2.ru-central1.internal", instance="elenas-adh-4-2.ru-central1.internal:25020", job="impala-catalog"}   119988224
impala_memory_total_used{component="impala-daemon", host_name="elenas-adh-4-2.ru-central1.internal", instance="elenas-adh-4-2.ru-central1.internal:25000", job="impala-daemon"}   116236288
impala_memory_total_used{component="impala-statestore", host_name="elenas-adh-4-3.ru-central1.internal", instance="elenas-adh-4-3.ru-central1.internal:25010", job="impala-statestore"}   7340032
```

Веб-интерфейс Prometheus

Веб-интерфейс Grafana

1. В адресной строке браузера введите адрес хоста, на котором развернут компонент Grafana сервиса Monitoring. В качестве номера порта укажите значение параметра **Grafana listen port** из раздела **Grafana settings** в настройках конфигурации сервиса мониторинга (по умолчанию `11210`). Например, *http://10.92.40.107:11210*. Для входа используйте `admin` в качестве имени пользователя и значение параметра **Grafana administrator's password** (находится также в разделе настроек **Grafana settings**) в качестве пароля.
2. В открывшемся окне выберите пункт меню **Home → Dashboards** и разверните секцию **ADH Dashboard <ADH_cluster_name>**. В этой секции можно выбрать один из доступных дашбордов для просмотра сервисных или системных метрик, поступающих из вашего кластера ADH.



Search or jump to...
cmd+k
Home › Dashboards
Dashboards
Playlists
Snapshots
Library panels
Create and manage dashboards to visualize your data
Search for dashboards
New
Filter by tag
Starred
Sort
General
hue

ADH Dashboard ADH all services

- HDFS - ADH all services
- Hive Metastore - ADH all services
- Hive Server - ADH all services
- Hue - ADH all services
- Impala Catalog - ADH all services
- Impala Daemon - ADH all services

Переход к метрикам кластера ADH в Grafana

Кластер мониторинга

Елена Костюченко

Назад к содержанию

Чтобы настроить мониторинг для ADH, можно сконфигурировать кластер мониторинга, используя сервисы Diamond, Graphite и Grafana, и интегрировать его с кластером ADH.

Для процесса **online**-установки необходим отдельный кластер мониторинга. Шаги для установки мониторинга в этом случае приведены ниже. Их можно выполнить до и после установки кластера ADH, кроме заключительного шага Интеграция с кластером ADH, который может быть выполнен только на базе добавленного кластера ADH.

- Создание кластера
- Добавление сервисов
- Добавление хостов в кластер
- Добавление компонентов
- Настройка сервисов
- Установка кластера
- Интеграция с кластером ADH

Создание кластера

Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Загрузка бандла

Шаг 2. Загрузка бандла в ADCM

Шаг 3. Создание кластера на базе загруженного бандла

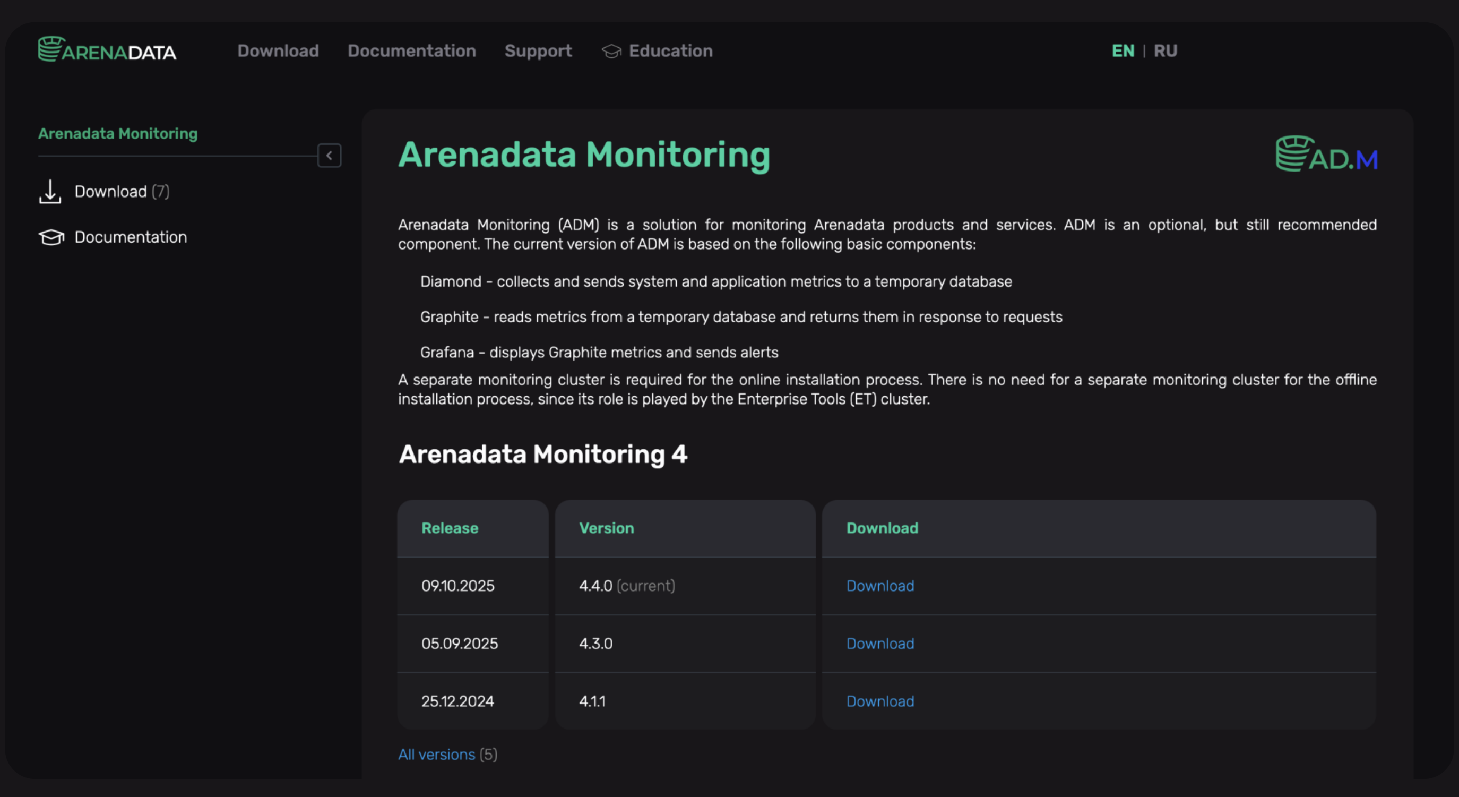
В ADCM **кластер** означает набор сервисов, запущенных на группе хостов.

Дистрибутивы кластеров для ADCM поставляются в виде бандлов. В общем случае бандл представляет обычный архив, описывающий логику развертывания кластера.

Шаги для добавления нового кластера приведены ниже.

Шаг 1. Загрузка бандла

1. Зайдите на сайт https://network.arenadata.io/ и выберите раздел ***Arenadata Monitoring***.
2. На следующей странице загрузите необходимый бандл.



ARENADATA
Download
Documentation
Support
Education
EN | RU
Arenadata Monitoring
Download (7)
Documentation
AD.M

Arenadata Monitoring

Arenadata Monitoring (ADM) is a solution for monitoring Arenadata products and services. ADM is an optional, but still recommended component. The current version of ADM is based on the following basic components:

Diamond - collects and sends system and application metrics to a temporary database

Graphite - reads metrics from a temporary database and returns them in response to requests

Grafana - displays Graphite metrics and sends alerts

A separate monitoring cluster is required for the online installation process. There is no need for a separate monitoring cluster for the offline installation process, since its role is played by the Enterprise Tools (ET) cluster.

Arenadata Monitoring 4

Release	Version	Download
09.10.2025	4.4.0 (current)	Download
05.09.2025	4.3.0	Download
25.12.2024	4.1.1	Download

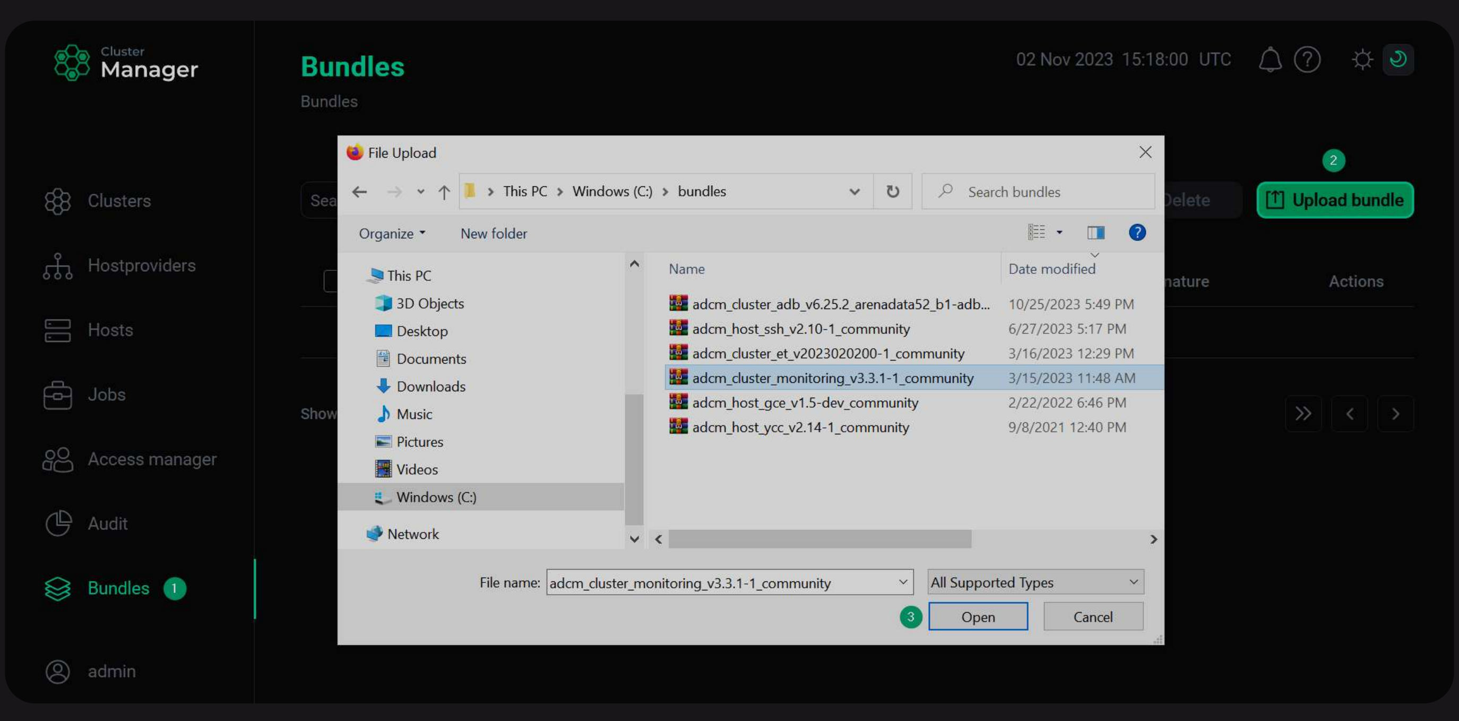
All versions (5)

Страница загрузки бандлов Arenadata Monitoring

Шаг 2. Загрузка бандла в ADCM

Чтобы загрузить бандл в ADCM:

1. Выберите пункт ***Bundles*** в левом навигационном меню и нажмите ***Upload bundle***.
2. Выберите бандл в открывшемся диалоговом окне.



Cluster Manager
Bundles
Bundles
02 Nov 2023 15:18:00 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
1
admin
2
Upload bundle
File Upload
This PC > Windows (C:) > bundles
Search bundles
Organize
New folder
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
Network
3
Open
Cancel

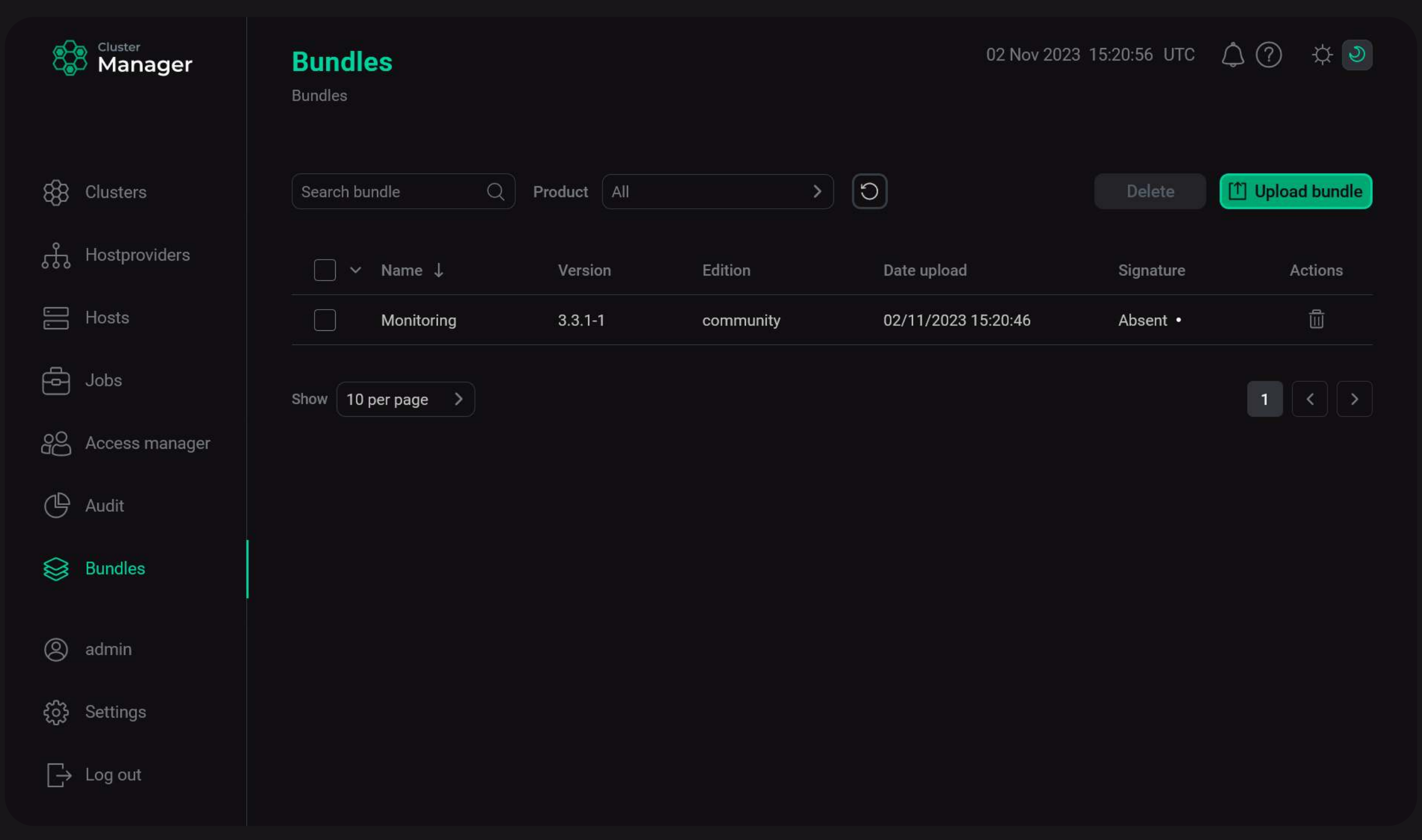
Name	Date modified
adcm_cluster_adb_v6.25.2_arenadata52_b1-adb...	10/25/2023 5:49 PM
adcm_host_ssh_v2.10-1_community	6/27/2023 5:17 PM
adcm_cluster_et_v2023020200-1_community	3/16/2023 12:29 PM
adcm_cluster_monitoring_v3.3.1-1_community	3/15/2023 11:48 AM
adcm_host_gce_v1.5-dev_community	2/22/2022 6:46 PM
adcm_host_ycc_v2.14-1_community	9/8/2021 12:40 PM

File name: adcm_cluster_monitoring_v3.3.1-1_community

All Supported Types

Переход к загрузке бандла

3. После успешной загрузки бандл отображается на странице ***Bundles***.



Cluster Manager
Bundles
Bundles
02 Nov 2023 15:20:56 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search bundle
Product
All
Delete
Upload bundle
Show
10 per page
1

Name	Version	Edition	Date upload	Signature	Actions
Monitoring	3.3.1-1	community	02/11/2023 15:20:46	Absent •	

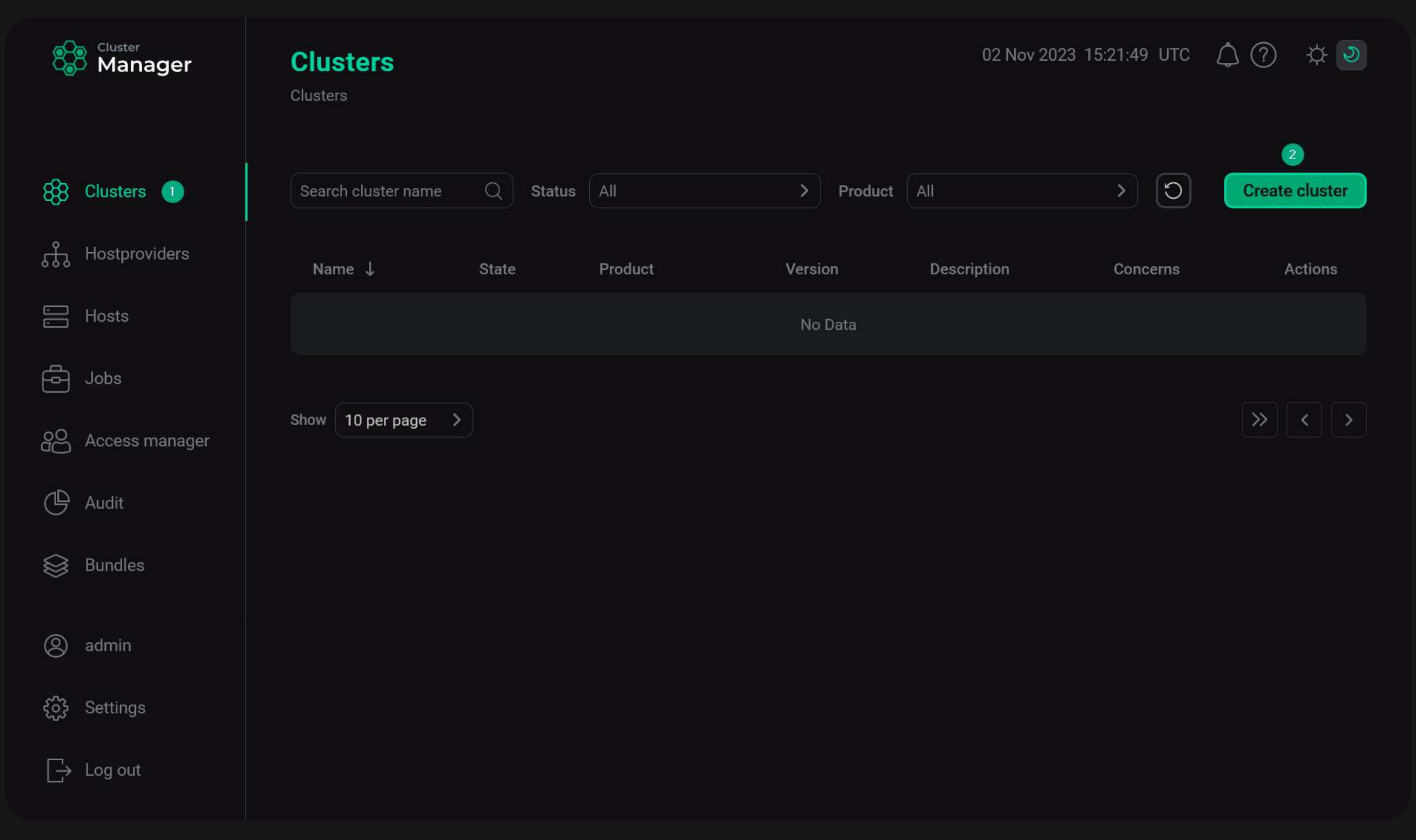
Результат успешной загрузки бандла

Шаг 3. Создание кластера на базе загруженного бандла

> **ВАЖНО**
>
> На этапе создания кластера через web-интерфейс ADCM генерируется только сам кластер. Это означает добавление записи о кластере в ADCM — сервисы на этом шаге еще не устанавливаются.

Для создания кластера необходимо выполнить шаги:

1. Выберите пункт ***Clusters*** в левом навигационном меню и нажмите ***Create cluster***.

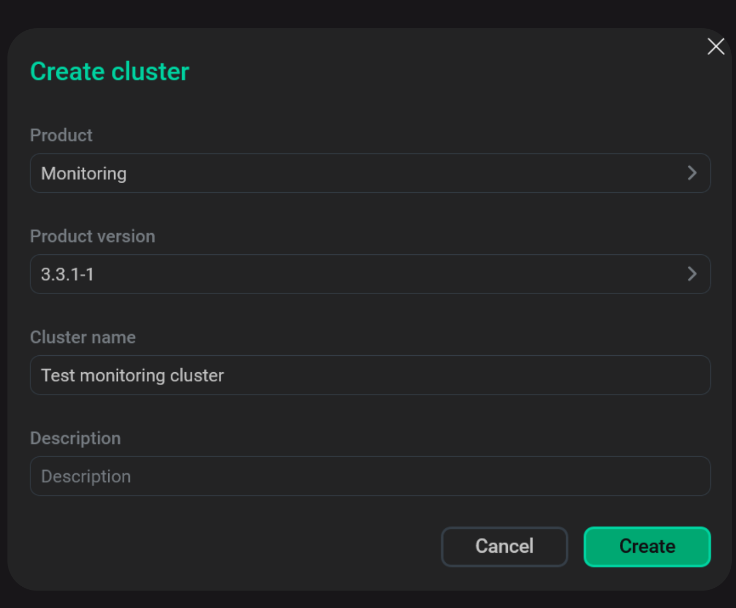


Cluster Manager
Clusters
Clusters
02 Nov 2023 15:21:49 UTC
Clusters
1
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
2
Create cluster
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
No Data						

Переход к созданию кластера

2. В открывшемся окне требуется описать новый кластер:
   - Выберите загруженный бандл в поле ***Product***.
   - Выберите версию бандла в поле ***Product version***. Несколько версий становятся доступны после загрузки нескольких бандлов для одного типа продуктов.
   - Введите имя кластера в поле ***Cluster name***.
   - При необходимости добавьте описание кластера в поле ***Description***.
   - Установите флаг ***I accept Terms of Agreement*** для подписания лицензионного соглашения. Флаг отображается только для бандлов версии Enterprise. Чтобы ознакомиться с текстом соглашения, перейдите по ссылке ***Terms of Agreement***. Подписать соглашение также возможно на странице ***Bundles***, нажав на соответствующий продуктовый бандл.
   - Нажмите ***Create***.



Create cluster
Cancel
Create

Product: Monitoring

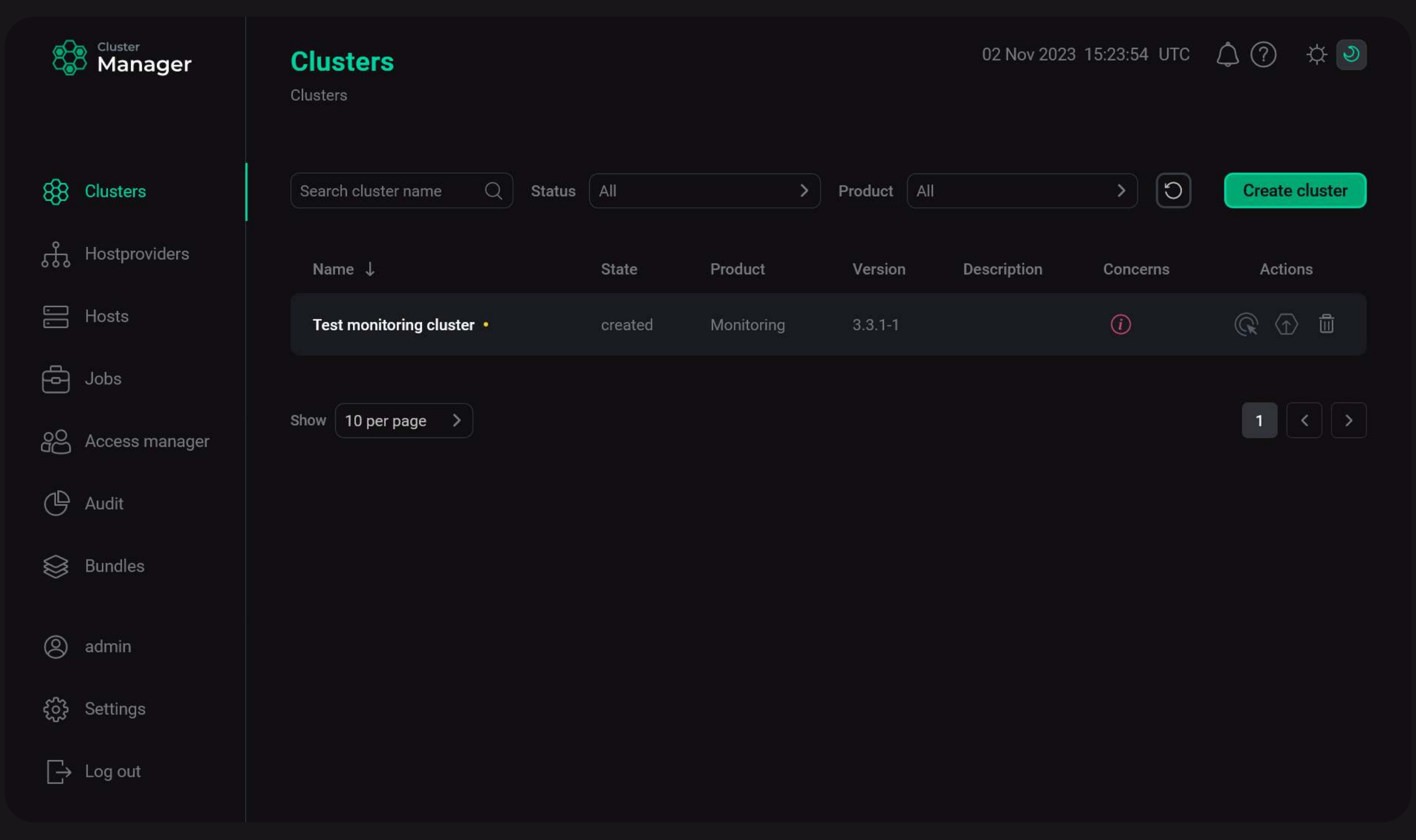
Product version: 3.3.1-1

Cluster name: Test monitoring cluster

Description: Description

Создание кластера

3. После успешного добавления кластер отображается на странице ***Clusters***.



Cluster Manager
Clusters
Clusters
02 Nov 2023 15:23:54 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test monitoring cluster •	created	Monitoring	3.3.1-1		ⓘ	

Результат успешного добавления кластера

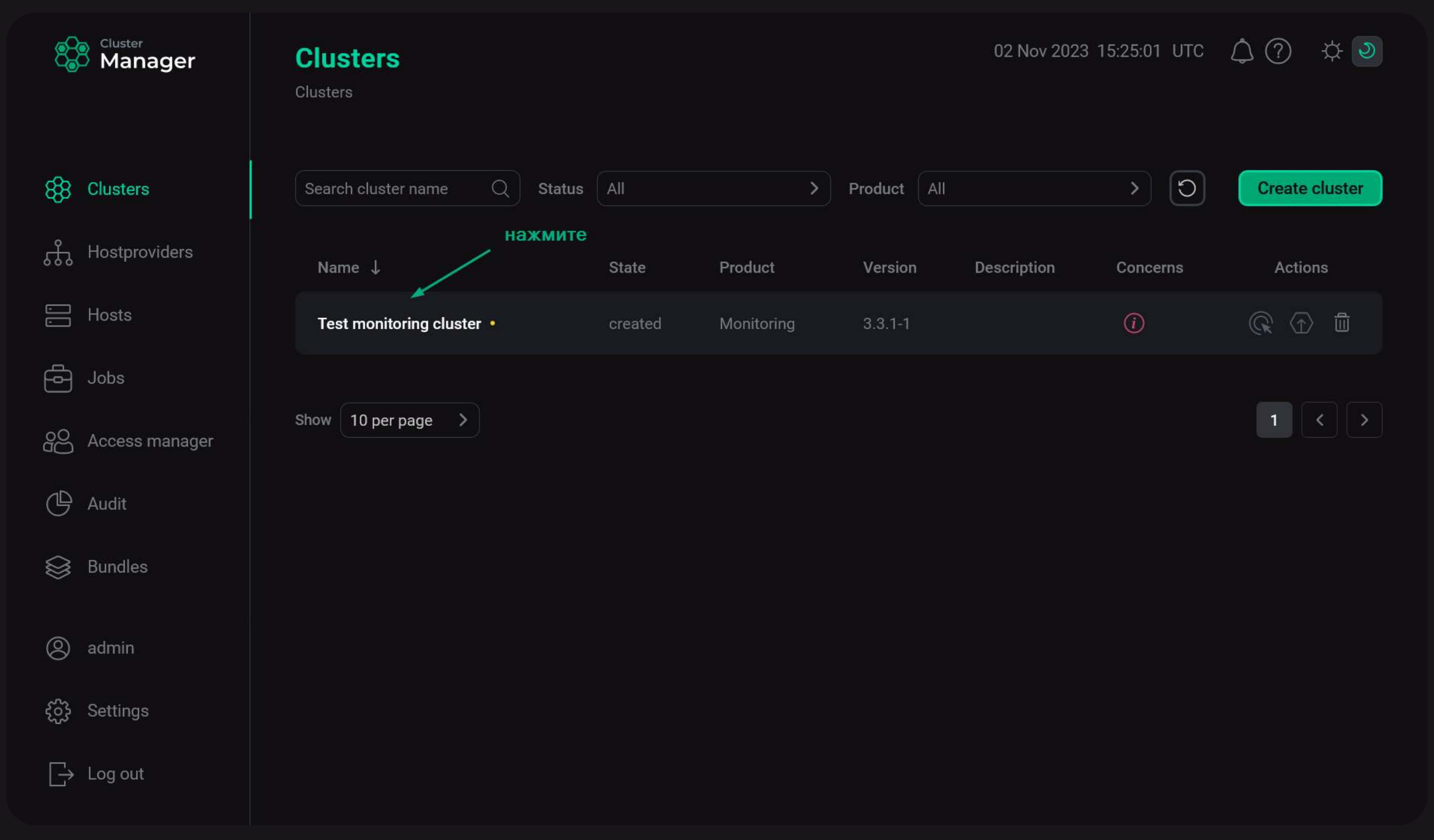
Добавление сервисов

Дарья Барышева

Назад к содержанию

В ADCM **сервис** означает программное обеспечение, выполняющее некоторую функцию. Для добавления сервисов в кластер:

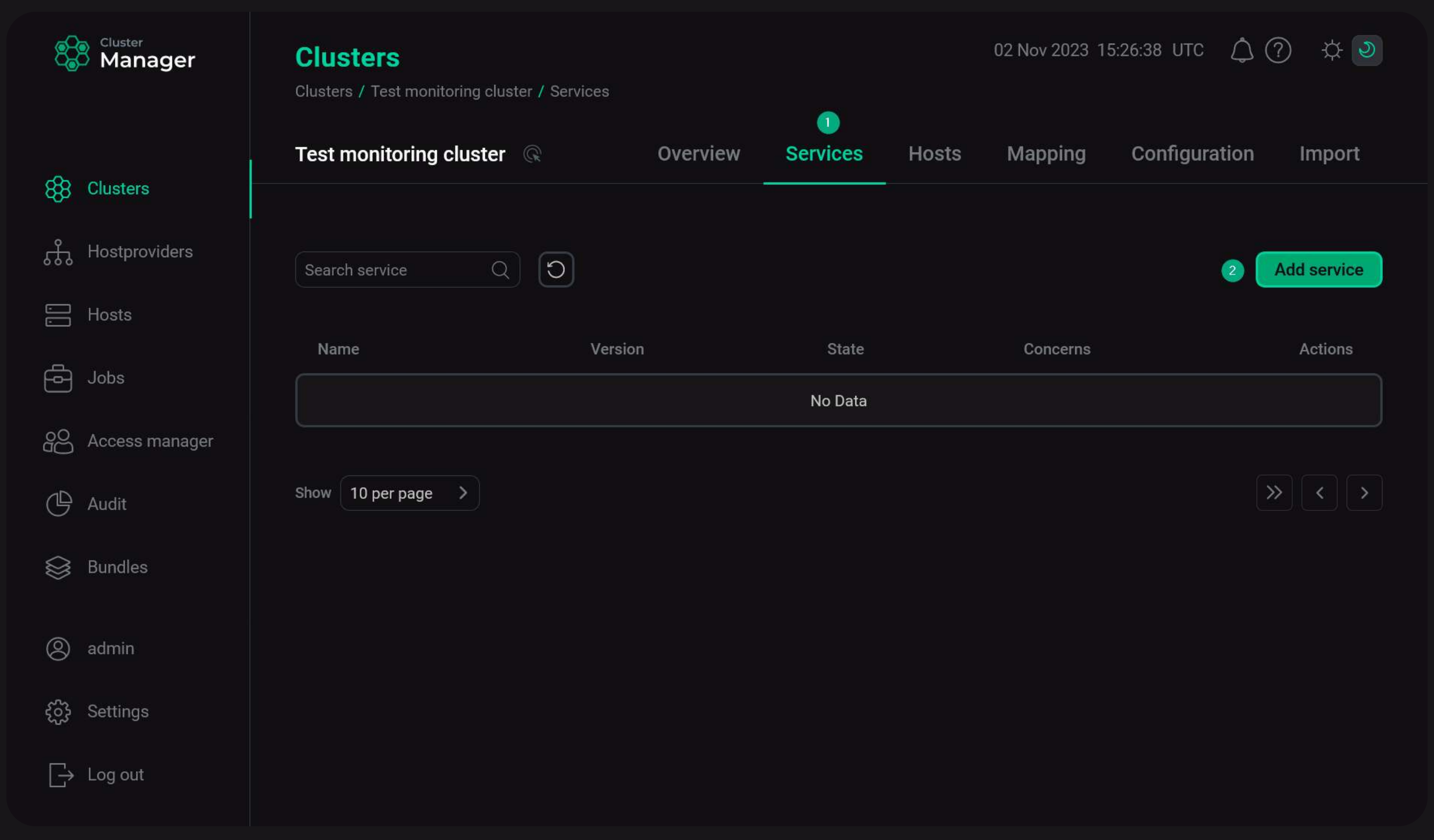
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster
Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
02 Nov 2023 15:25:01 UTC
Search cluster name
Status
All
Product
All
Create cluster
нажмите
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test monitoring cluster	created	Monitoring	3.3.1-1			

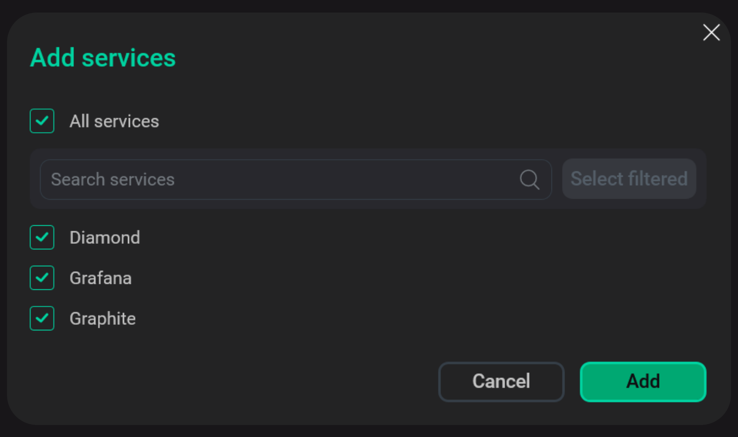
Выбор кластера

2. Откройте вкладку ***Services*** на странице кластера и нажмите ***Add service***.


Cluster
Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test monitoring cluster / Services
02 Nov 2023 15:26:38 UTC
Test monitoring cluster
Overview
1
Services
Hosts
Mapping
Configuration
Import
Search service
2
Add service
Name
Version
State
Concerns
Actions
No Data
Show
10 per page

Переход к добавлению сервисов

3. В открывшемся окне выберите сервисы для добавления в кластер и нажмите ***Add***.


Add services
☑ All services
Search services
Select filtered
☑ Diamond
☑ Grafana
☑ Graphite
Cancel
Add

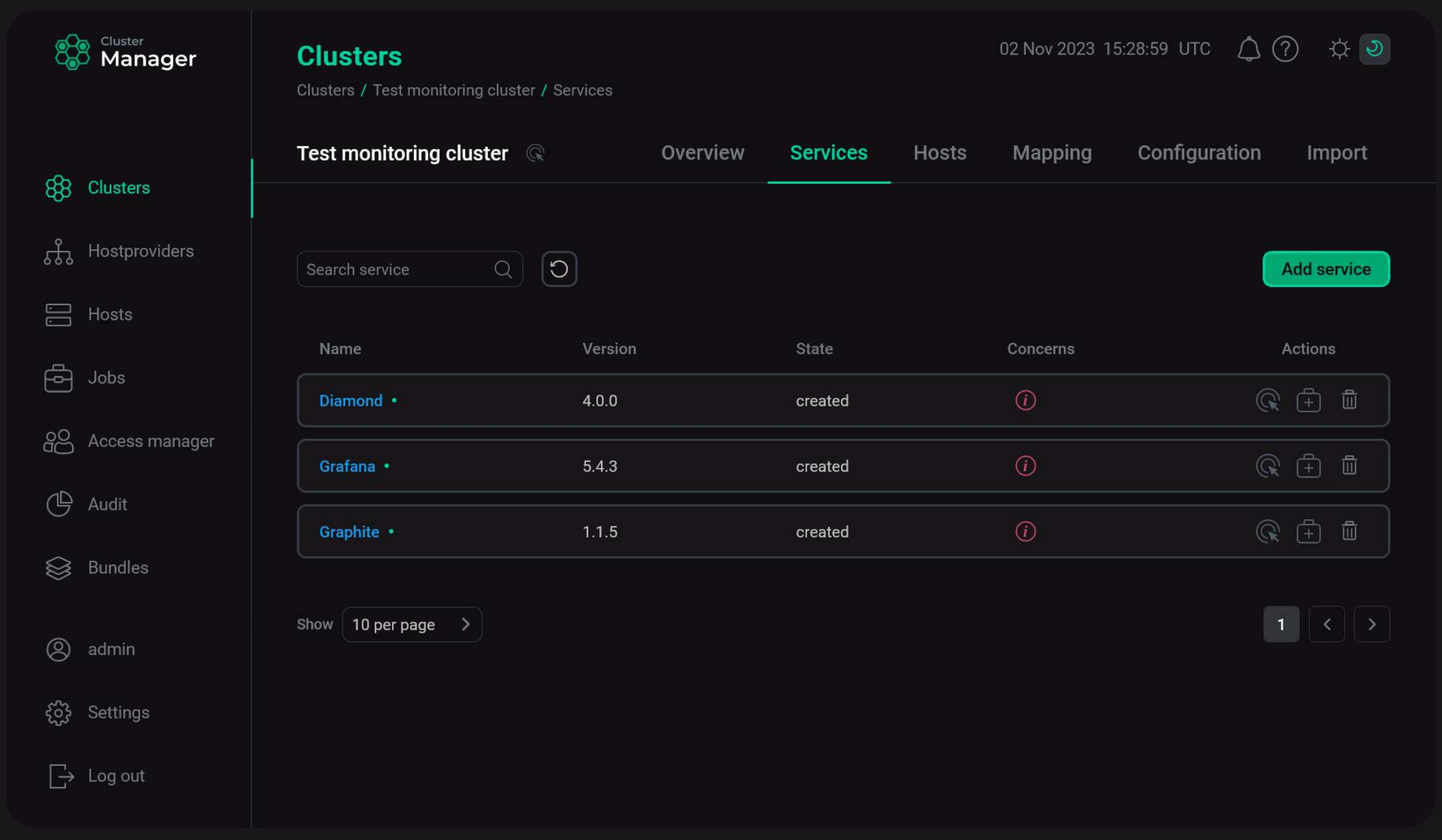
Выбор сервисов

Краткое описание доступных сервисов приведено ниже.

Сервисы, которые необходимо добавить в кластер мониторинга

Сервис	Назначение
Diamond	Python-сервис, используемый для сбора системных метрик с хостов, подключенных к системе мониторинга. Собранные метрики обрабатываются в Carbon (backend-сервисах Graphite)
Graphite	Состоит из нескольких компонентов: ▪ **Carbon** — набор сервисов для обработки метрик, поступающих из Diamond. ▪ **Whisper DB** — временная файловая база данных, где хранятся собранные метрики. ▪ **Web service** — web-сервис, позволяющий получать и визуализировать метрики из базы данных Whisper DB.
Grafana	Web-сервис, предназначенный для создания графиков и оповещений для визуализации метрик, полученных из Graphite. По умолчанию использует порт `3000`

4. В результате успешно добавленные сервисы отображаются на вкладке ***Services***.


Cluster
Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test monitoring cluster / Services
02 Nov 2023 15:28:59 UTC
Test monitoring cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add service
Show
10 per page
1

Name	Version	State	Concerns	Actions
Diamond	4.0.0	created		
Grafana	5.4.3	created		
Graphite	1.1.5	created		

Результат успешного добавления сервисов в кластер

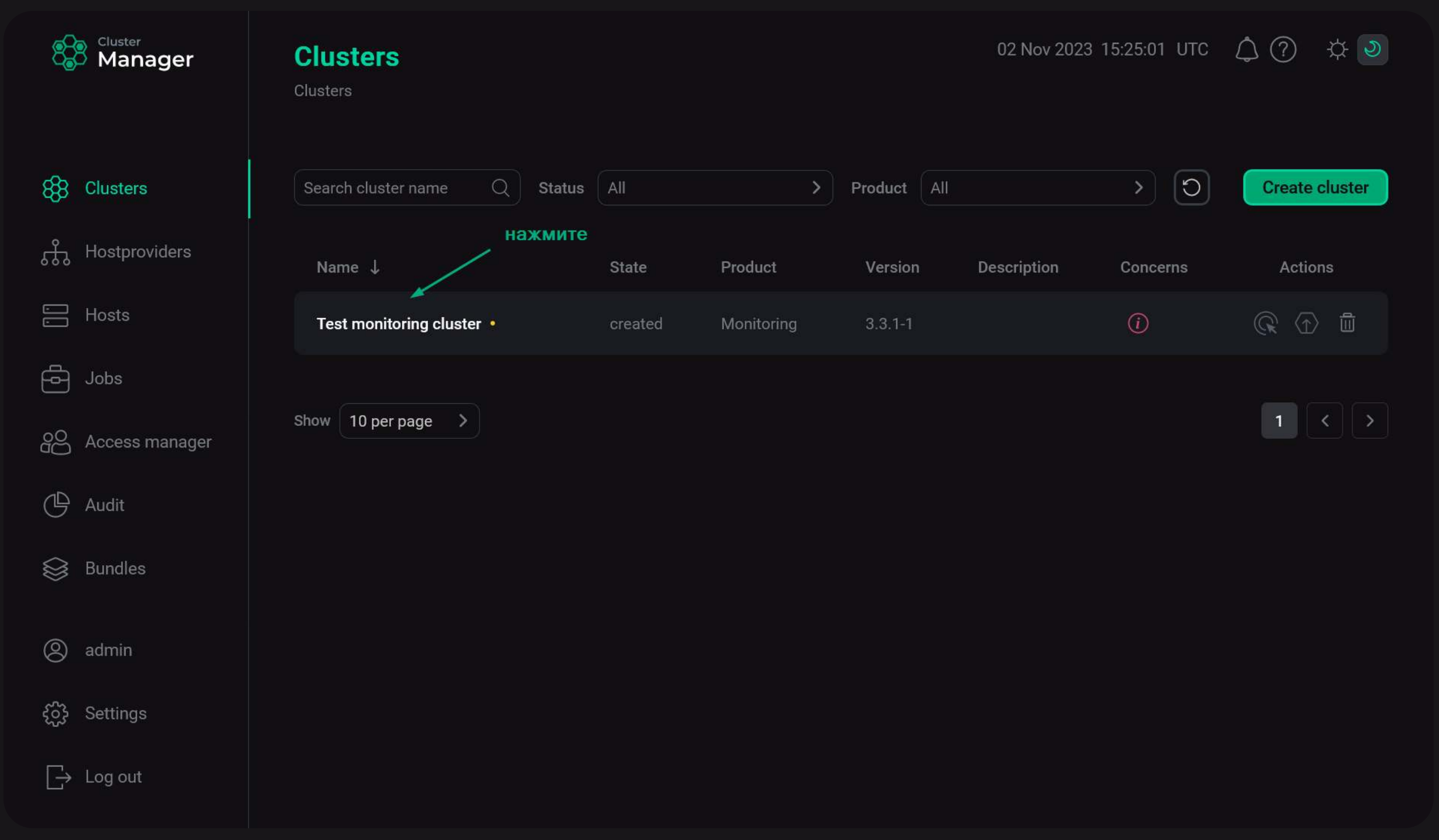
Добавление хостов в кластер

Дарья Барышева

Назад к содержанию

Для добавления хостов в кластер:

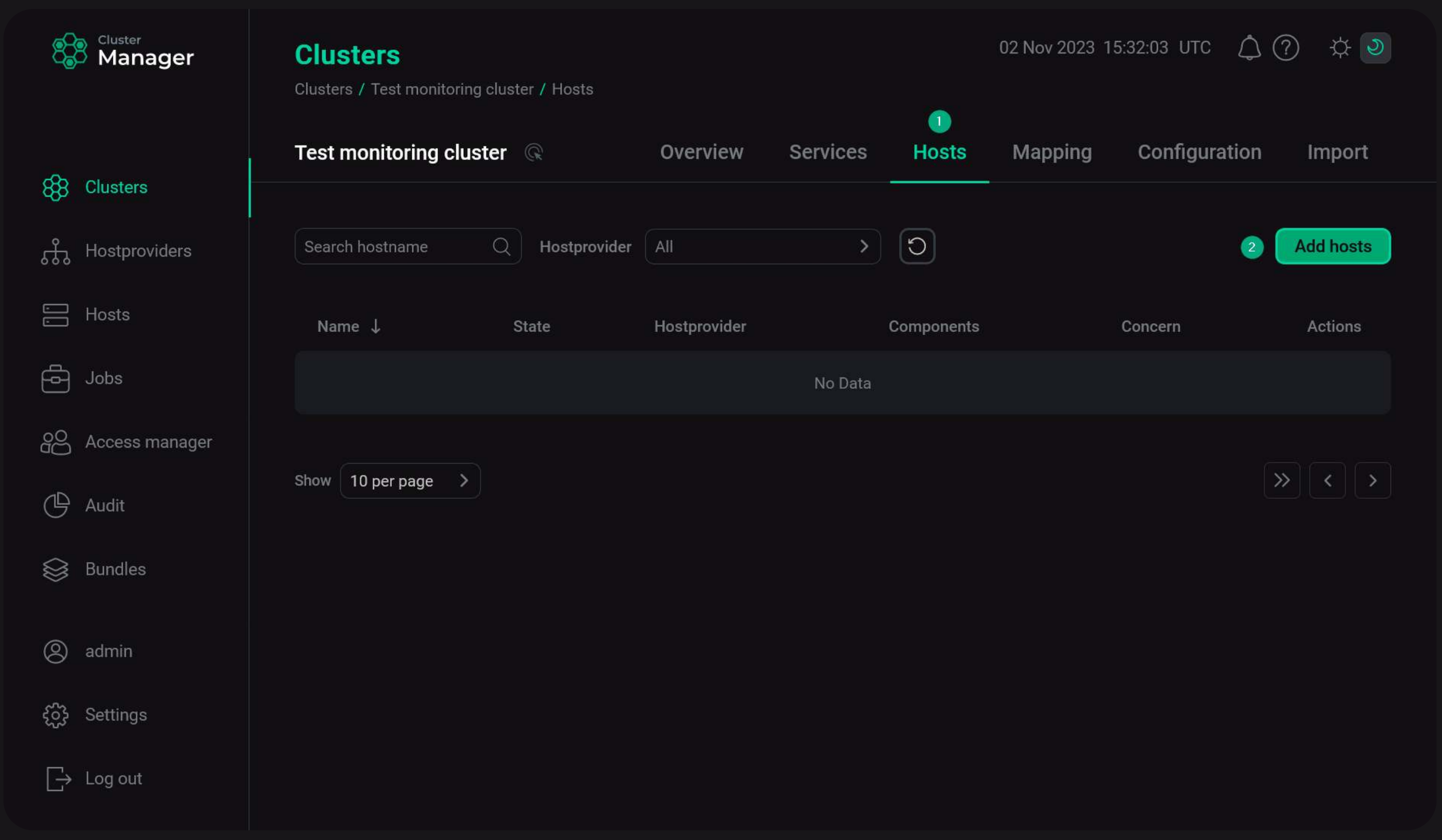
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
02 Nov 2023 15:25:01 UTC
Search cluster name
Status
All
Product
All
Create cluster
нажмите
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test monitoring cluster	created	Monitoring	3.3.1-1			

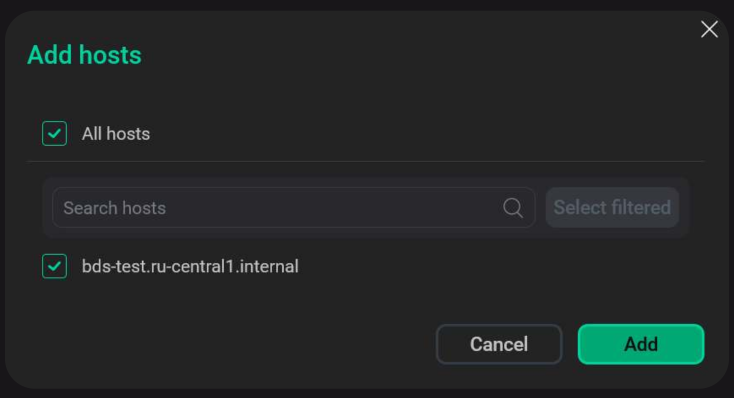
Выбор кластера

2. Откройте вкладку ***Hosts*** на странице кластера и нажмите ***Add hosts***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test monitoring cluster / Hosts
02 Nov 2023 15:32:03 UTC
Test monitoring cluster
Overview
Services
1
Hosts
Mapping
Configuration
Import
Search hostname
Hostprovider
All
2
Add hosts
Name
State
Hostprovider
Components
Concern
Actions
No Data
Show
10 per page

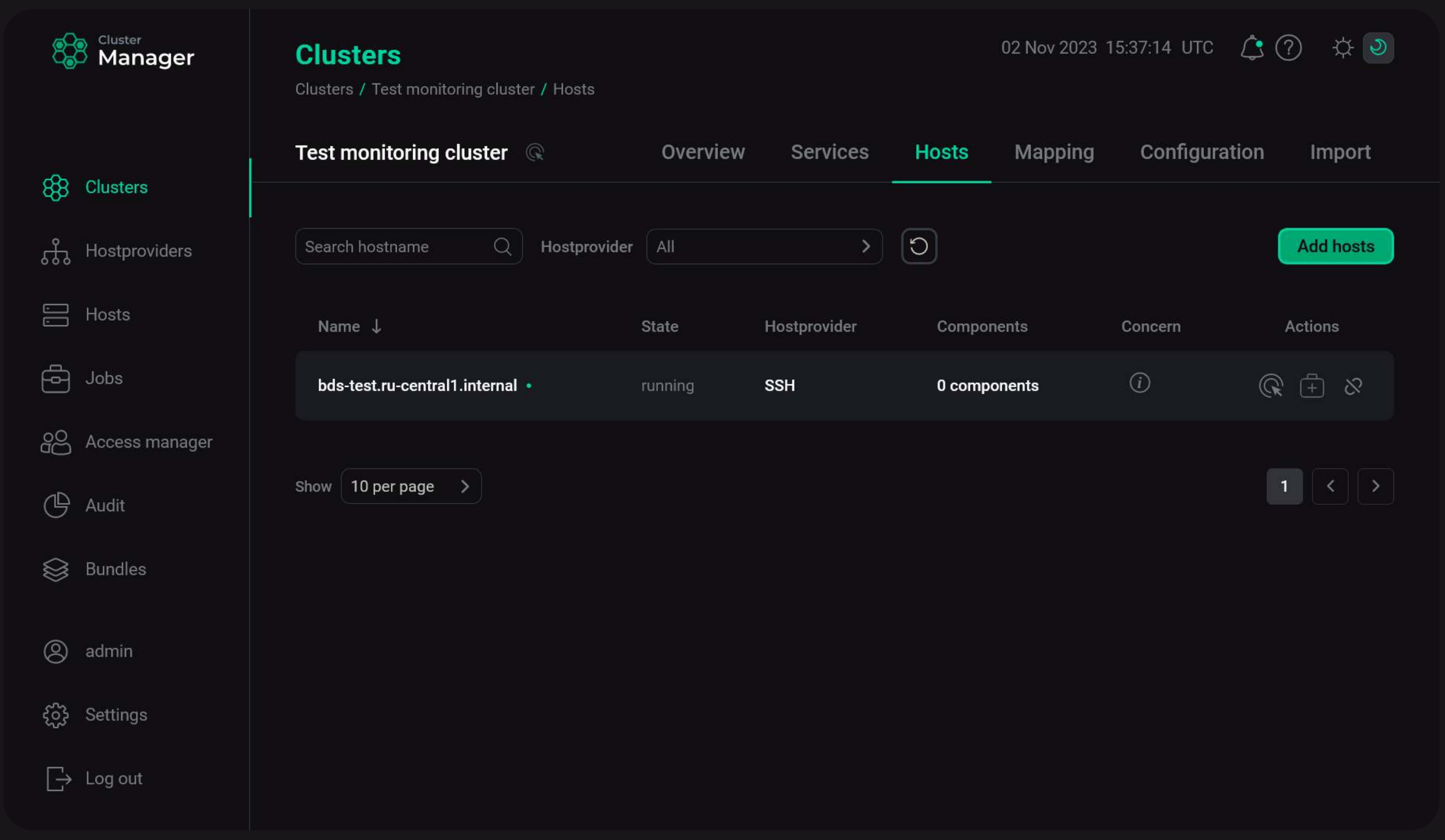
Переход к добавлению хостов

3. В открывшемся окне выберите хосты для добавления в кластер и нажмите ***Add***.


Add hosts
☑ All hosts
Search hosts
Select filtered
☑ bds-test.ru-central1.internal
Cancel
Add

Выбор хостов

4. В результате успешно добавленные хосты отображаются на вкладке ***Hosts***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test monitoring cluster / Hosts
02 Nov 2023 15:37:14 UTC
Test monitoring cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Search hostname
Hostprovider
All
Add hosts
Show
10 per page
1

Name	State	Hostprovider	Components	Concern	Actions
bds-test.ru-central1.internal	running	SSH	0 components		

Результат успешного добавления хостов в кластер

Добавление компонентов

Дарья Барышева

Назад к содержанию

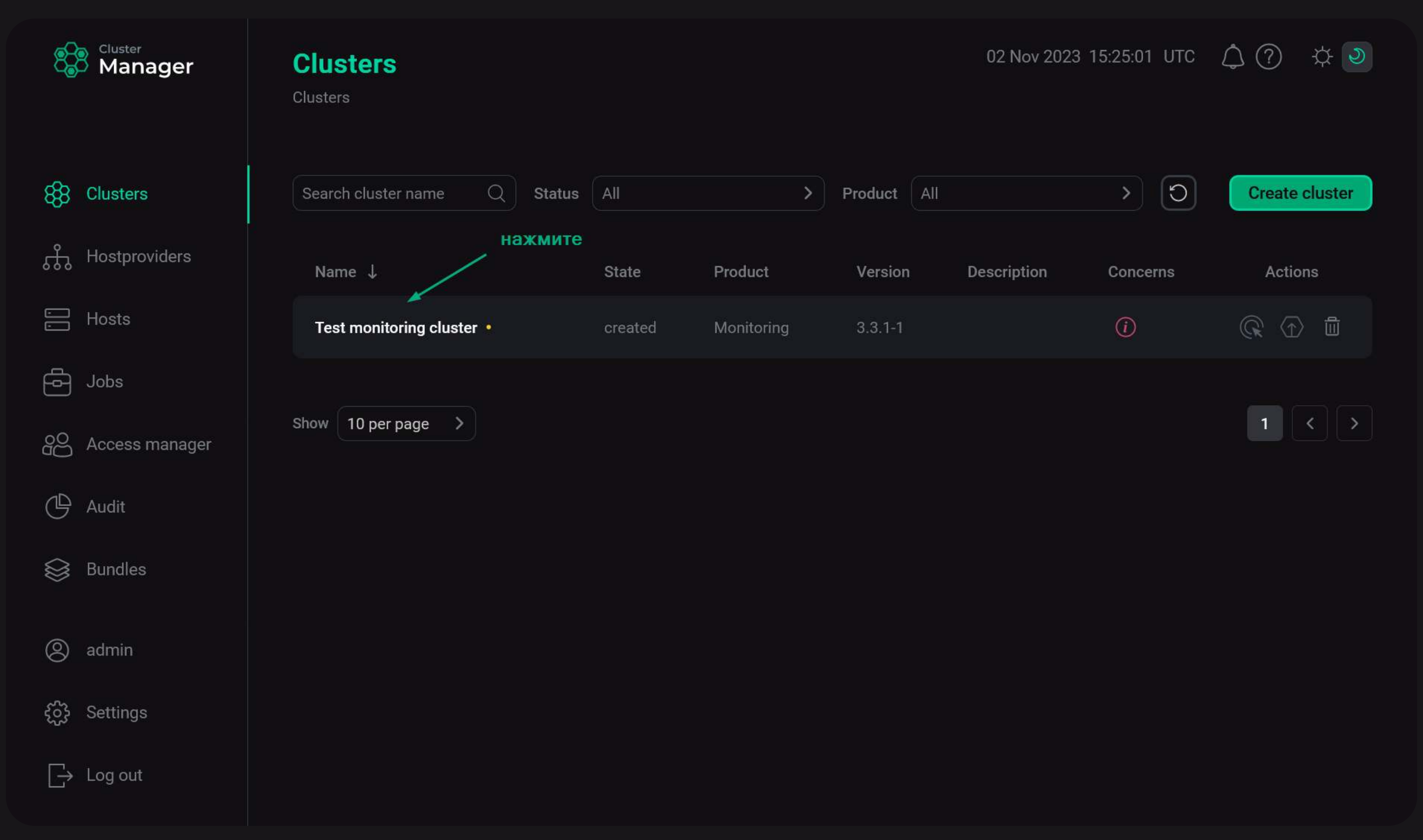
В ADCM **компонент** означает часть сервиса, которая должна быть развернута на одном или нескольких хостах кластера.

ВАЖНО

Изначально на хостах кластера нет ни одного компонента — распределение компонентов между хостами должно быть проведено вручную.

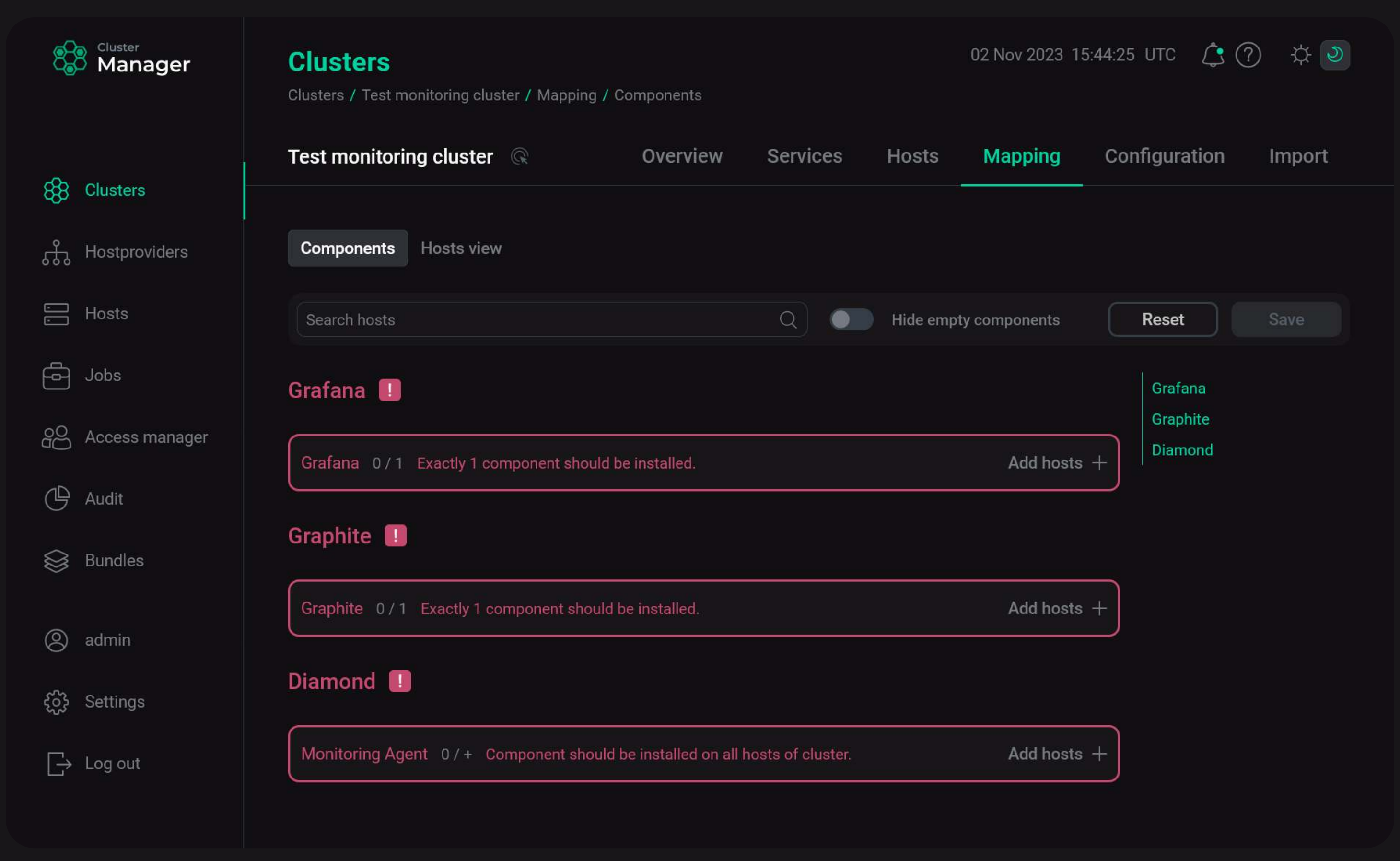
Для добавления компонентов на хосты кластера:

1. Выберите кластер на странице *Clusters*. Для этого нажмите на имя кластера в столбце *Name*.


Cluster Manager
Clusters
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status All
Product All
Create cluster
нажмите
Name
State
Product
Version
Description
Concerns
Actions
Test monitoring cluster
created
Monitoring
3.3.1-1
Show 10 per page

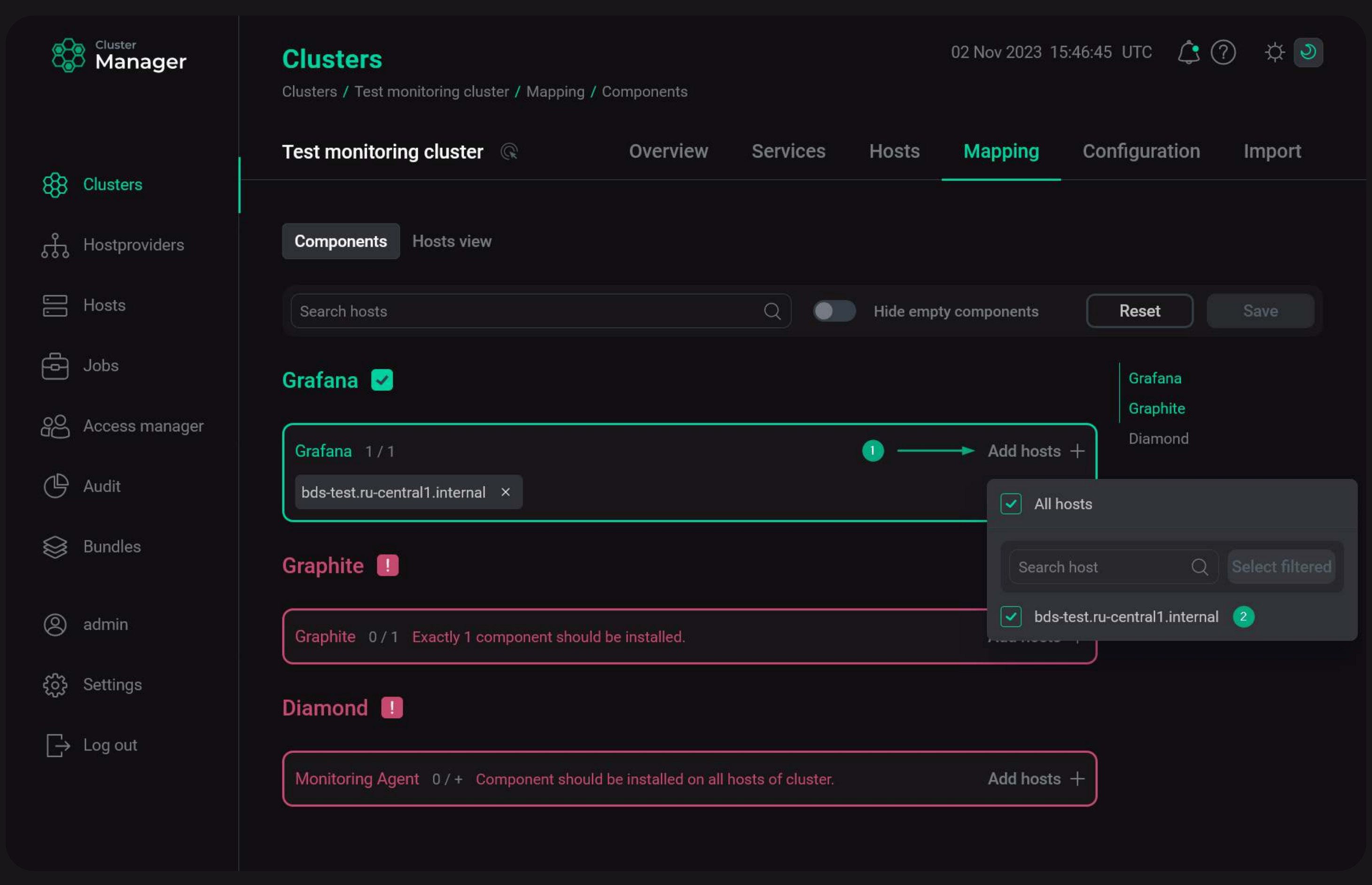
Выбор кластера

2. Откройте вкладку *Mapping* на странице кластера.


Clusters
Clusters / Test monitoring cluster / Mapping / Components
Test monitoring cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Components
Hosts view
Search hosts
Hide empty components
Reset
Save
Grafana
Grafana 0 / 1 Exactly 1 component should be installed.
Add hosts +
Graphite
Graphite 0 / 1 Exactly 1 component should be installed.
Add hosts +
Diamond
Monitoring Agent 0 / + Component should be installed on all hosts of cluster.
Add hosts +
Grafana
Graphite
Diamond

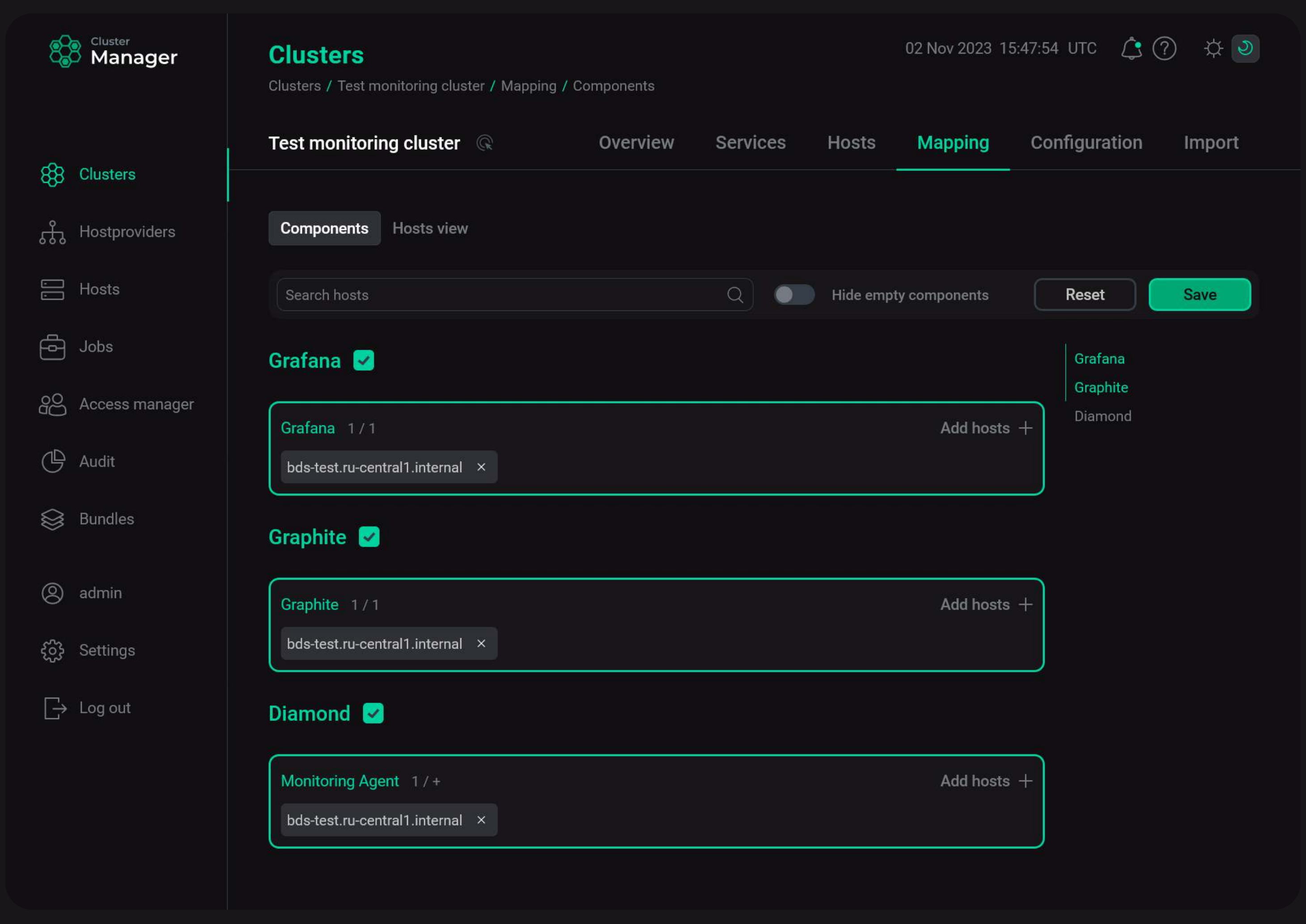
Переход к распределению сервисных компонентов

3. Распределите все компоненты между хостами кластера. Для назначения хостов компоненту нажмите ***Add hosts*** и выберите хосты в открывшемся окне.


Grafana
Grafana 1 / 1
bds-test.ru-central1.internal
1
Add hosts +
All hosts
Search host
Select filtered
bds-test.ru-central1.internal
2
Graphite
Graphite 0 / 1 Exactly 1 component should be installed.
Diamond
Monitoring Agent 0 / + Component should be installed on all hosts of cluster.
Add hosts +

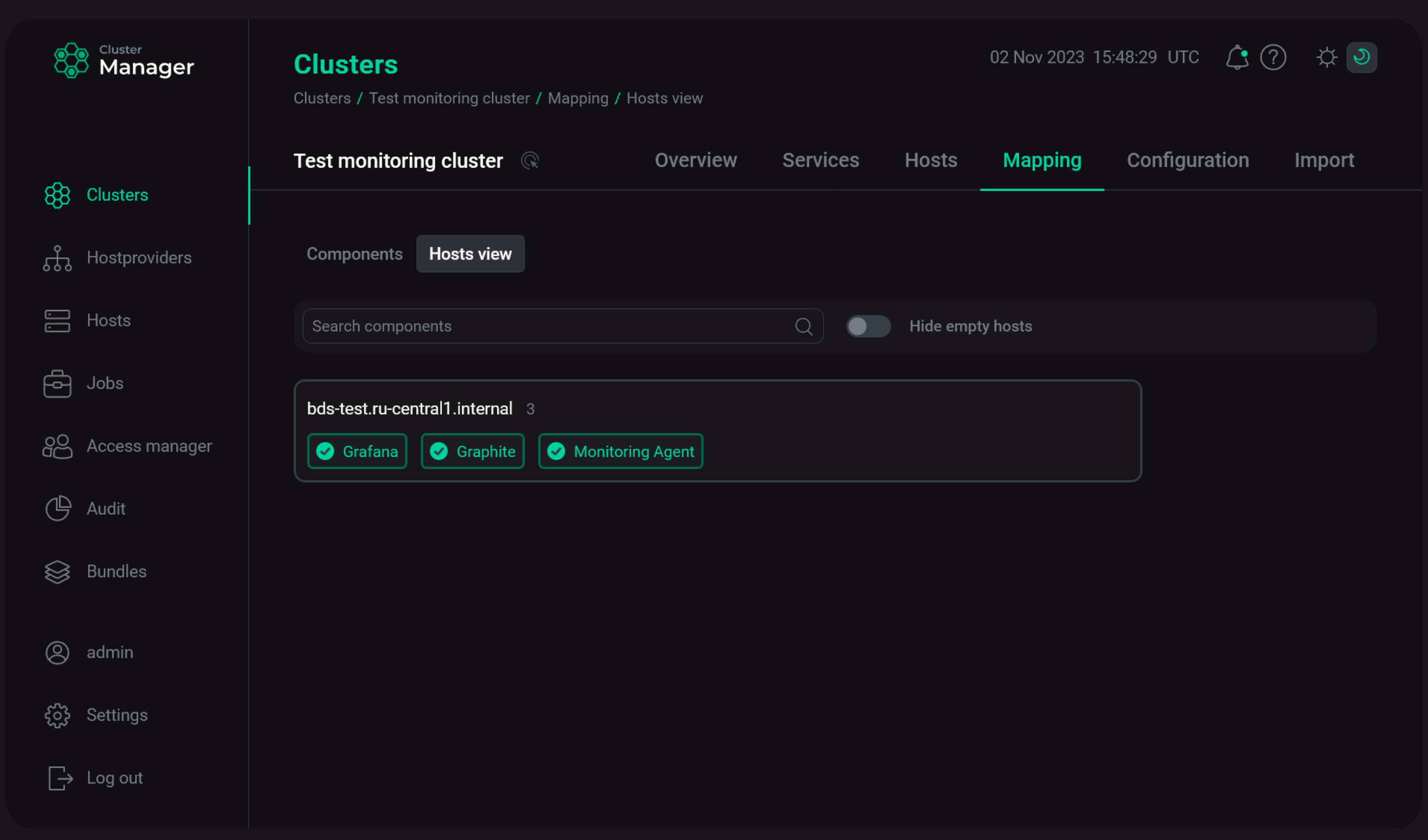
Выбор хостов для компонента

4. После завершения распределения компонентов нажмите *Save*.


Reset
Save
Grafana
Grafana 1 / 1
bds-test.ru-central1.internal
Graphite
Graphite 1 / 1
bds-test.ru-central1.internal
Diamond
Monitoring Agent 1 / +
bds-test.ru-central1.internal
Add hosts +

Сохранение распределения компонентов

5. Для перехода к альтернативному варианту просмотра — с разбиением по хостам — можно открыть вкладку *Hosts view*.


Clusters / Test monitoring cluster / Mapping / Hosts view
Components
Hosts view
Search components
Hide empty hosts
bds-test.ru-central1.internal 3
Grafana
Graphite
Monitoring Agent

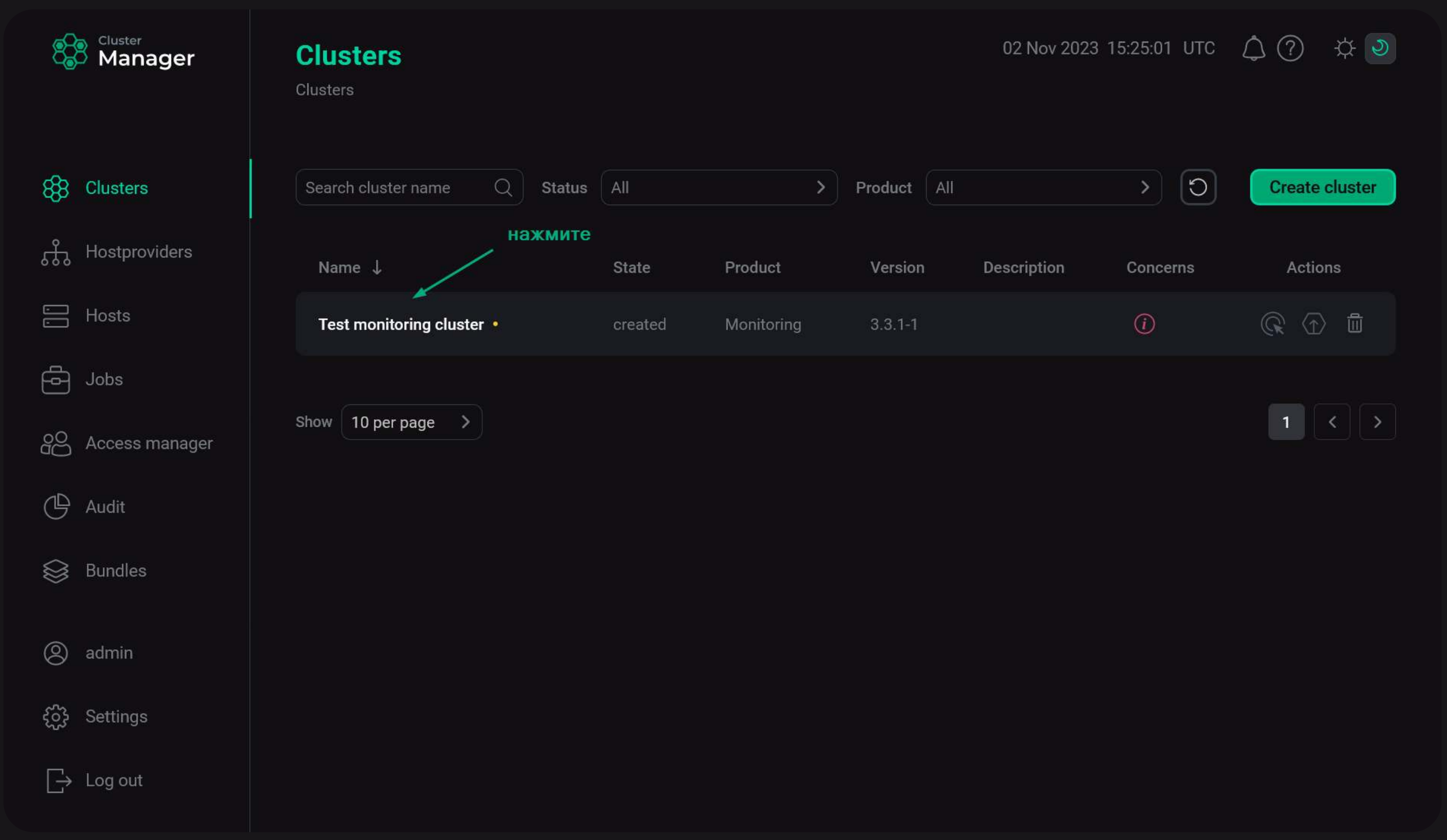
Просмотр распределения по хостам

Настройка сервисов

Дарья Барышева

Назад к содержанию

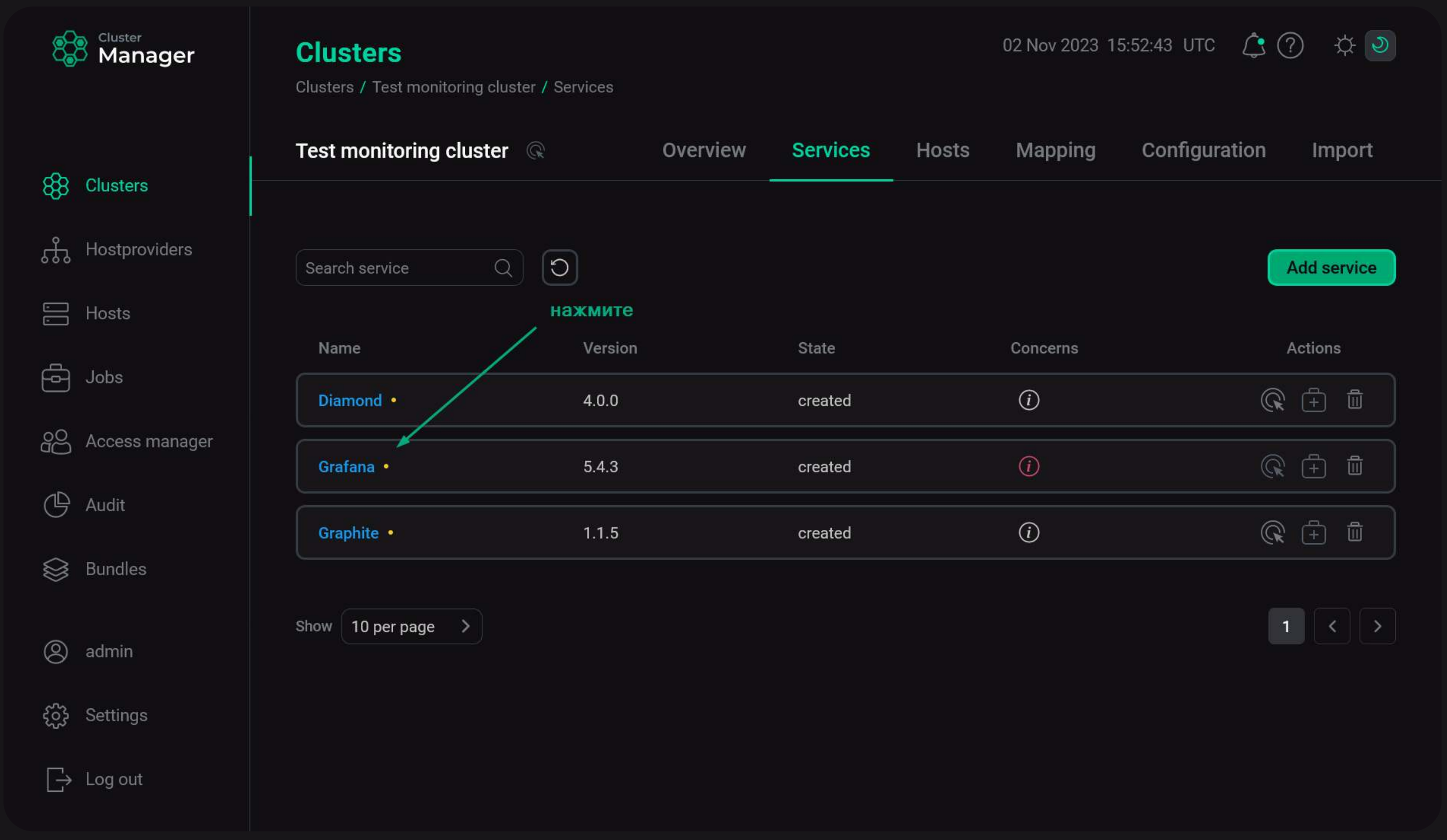
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
02 Nov 2023 15:25:01 UTC
Search cluster name
Status All
Product All
Create cluster
нажмите
Show 10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test monitoring cluster	created	Monitoring	3.3.1-1			

Выбор кластера

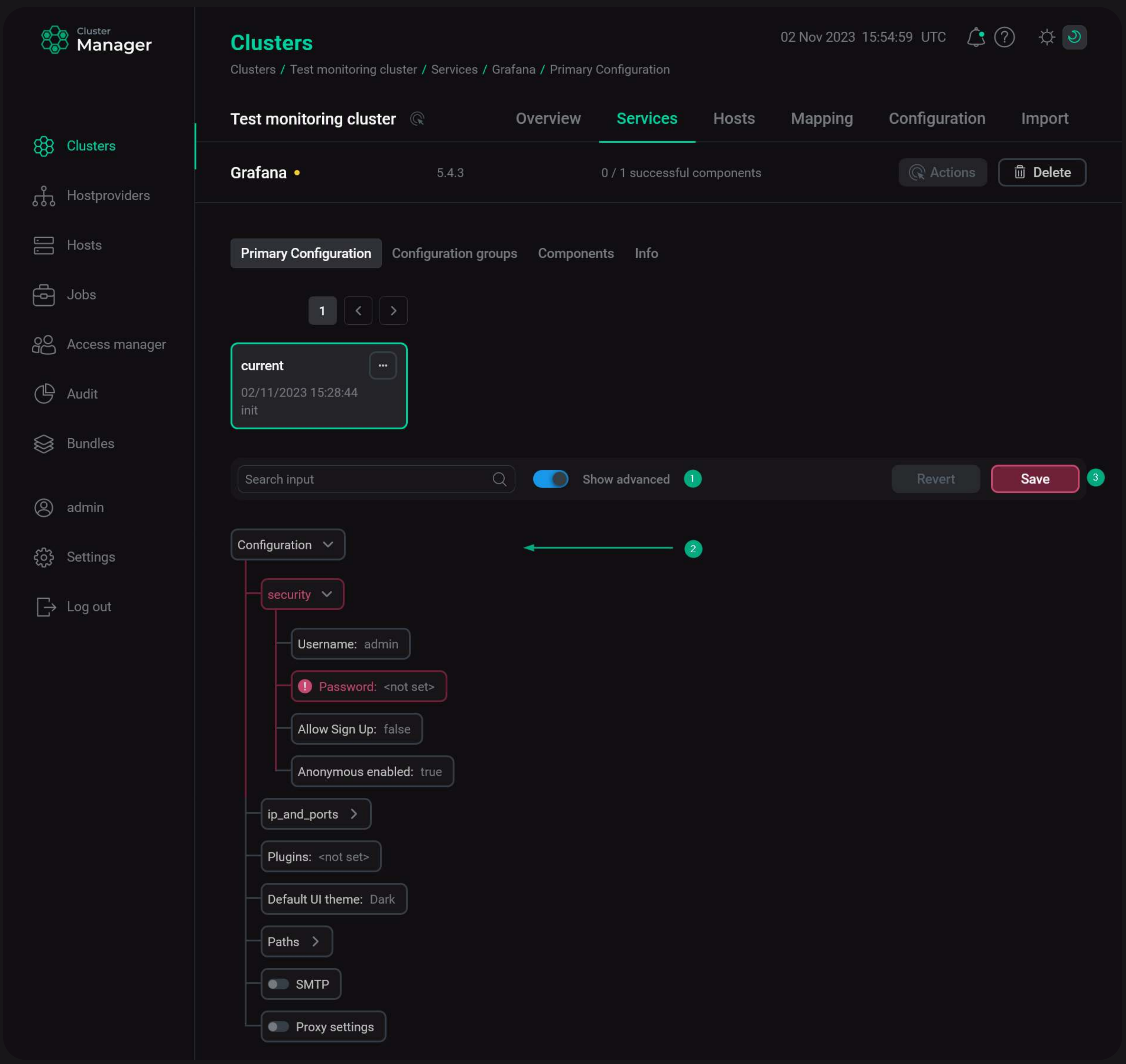
2. Откройте вкладку ***Services*** на странице кластера. Нажмите на имя сервиса в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test monitoring cluster / Services
02 Nov 2023 15:52:43 UTC
Test monitoring cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add service
нажмите
Show 10 per page
1

Name	Version	State	Concerns	Actions
Diamond	4.0.0	created		
Grafana	5.4.3	created		
Graphite	1.1.5	created		

Переход к настройке сервисов

3. В открывшемся окне:
   - Откройте вкладку ***Primary Configuration***.
   - Переведите в активное состояние переключатель ***Show advanced***.
   - Заполните конфигурационные параметры выбранного сервиса. Поля, подсвеченные красным, обязательны для заполнения.
   - Нажмите ***Save***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test monitoring cluster / Services / Grafana / Primary Configuration
02 Nov 2023 15:54:59 UTC
Test monitoring cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Actions
Delete
Primary Configuration
Configuration groups
Components
Info
1
Search input
Show advanced
1
Revert
Save
3
2
Configuration
security
ip_and_ports
Paths
SMTP
Proxy settings

Grafana 5.4.3 0 / 1 successful components

current
02/11/2023 15:28:44
init

Username: admin
Password: <not set>
Allow Sign Up: false
Anonymous enabled: true
Plugins: <not set>
Default UI theme: Dark

Настройка сервиса

Установка кластера

Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Запуск установки

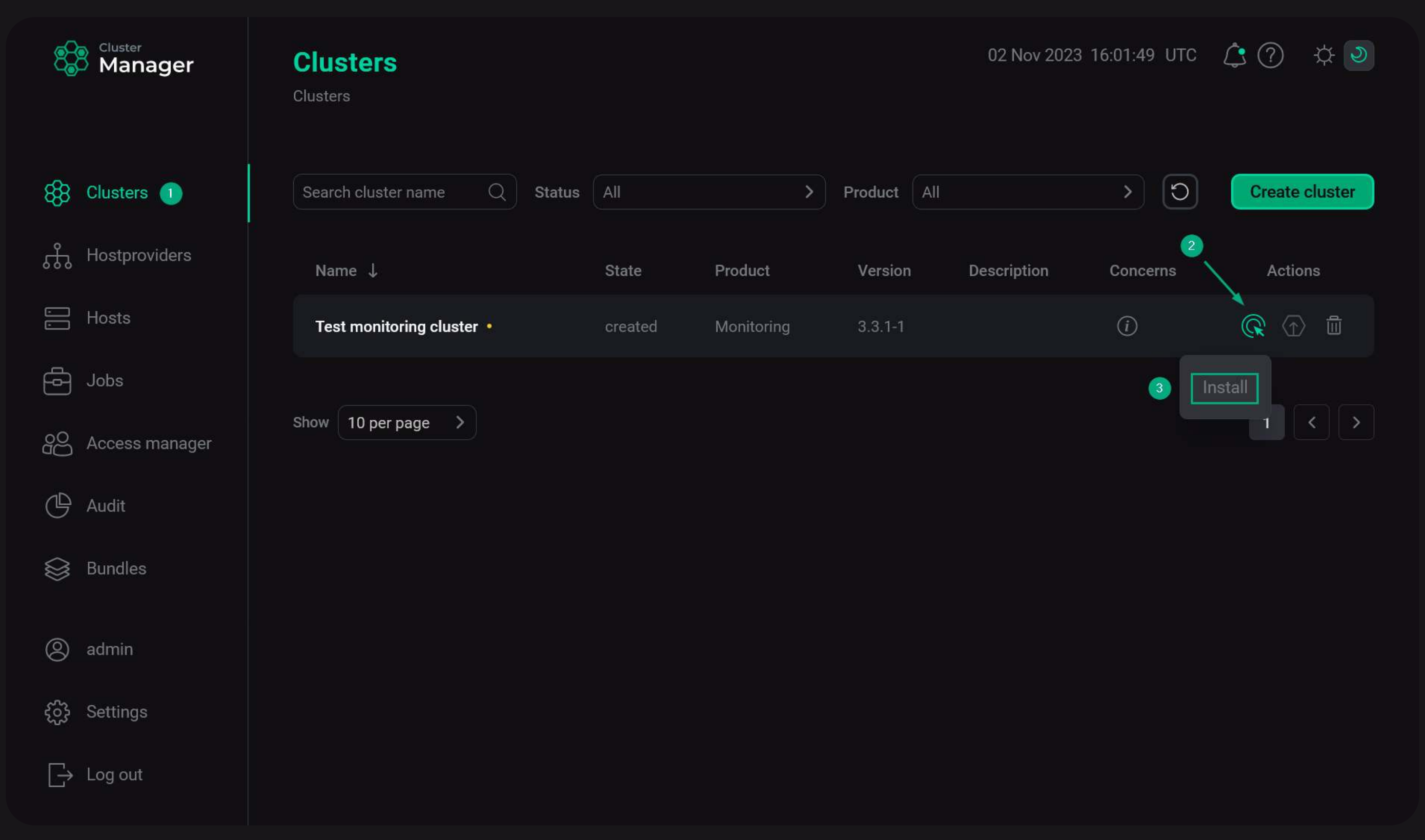
Шаг 2. Проверка результатов

- Graphite
- Grafana

Шаг 1. Запуск установки

Для установки кластера мониторинга необходимо выполнить шаги:

1. Запустите действие ***Install*** для кластера, выбранного на странице ***Clusters***, нажав на иконку в столбце ***Actions***. Подтвердите операцию в открывшемся окне. Все сервисы устанавливаются по очереди.



Cluster Manager
Clusters
Clusters
02 Nov 2023 16:01:49 UTC
Clusters 1
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status All
Product All
Create cluster
2
3
Install
Show 10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test monitoring cluster	created	Monitoring	3.3.1-1			

Переход к установке кластера мониторинга

2. Дождитесь завершения процесса установки.

ПРИМЕЧАНИЕ

В случае ошибок логи можно найти на странице ***Jobs***.

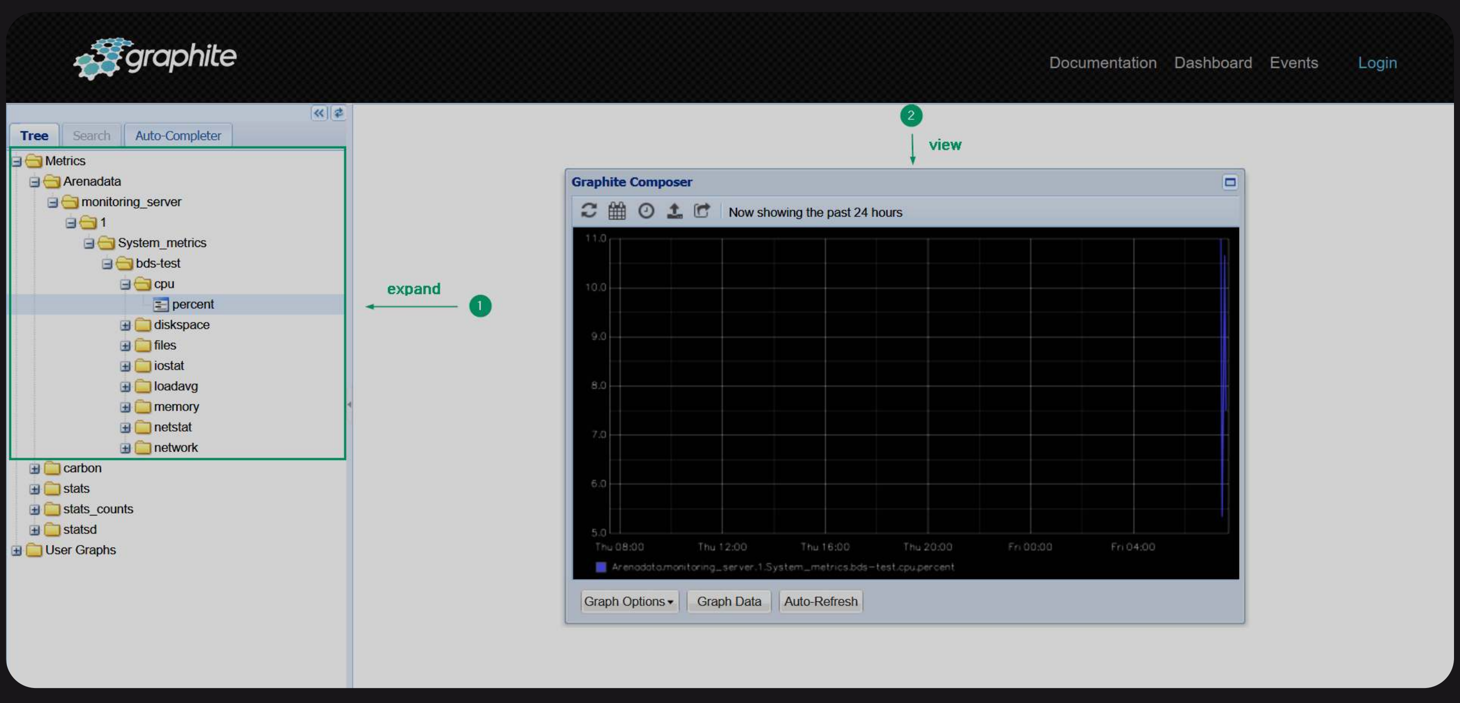
Шаг 2. Проверка результатов

Graphite

Чтобы проверить доступность web-интерфейса Graphite, необходимо выполнить шаги:

1. В адресной строке браузера введите адрес хоста, на котором развернут кластер мониторинга. В качестве номера порта укажите значение параметра ***Web-interface TCP port*** из раздела ***ip_and_ports***, определенное на этапе конфигурирования сервиса Graphite при установке кластера мониторинга. Значение `80` (используемое по умолчанию) можно не указывать.

2. Разверните дерево в левой части открывшейся формы. Здесь отображаются метрики System метрики с группировкой по хостам. Можно выбрать любую из метрик для просмотра динамики ее изменения в виде графика в правой части экрана.



graphite
Documentation
Dashboard
Events
Login
Tree
Search
Auto-Completer
Metrics
Arenadata
monitoring_server
1
System_metrics
bds-test
cpu
percent
diskspace
files
iostat
loadavg
memory
netstat
network
carbon
stats
stats_counts
statsd
User Graphs
expand
1
2
view
Graphite Composer
Now showing the past 24 hours
Thu 08:00
Thu 12:00
Thu 16:00
Thu 20:00
Fri 00:00
Fri 04:00
Arenadata.monitoring_server.1.System_metrics.bds-test.cpu.percent
Graph Options
Graph Data
Auto-Refresh

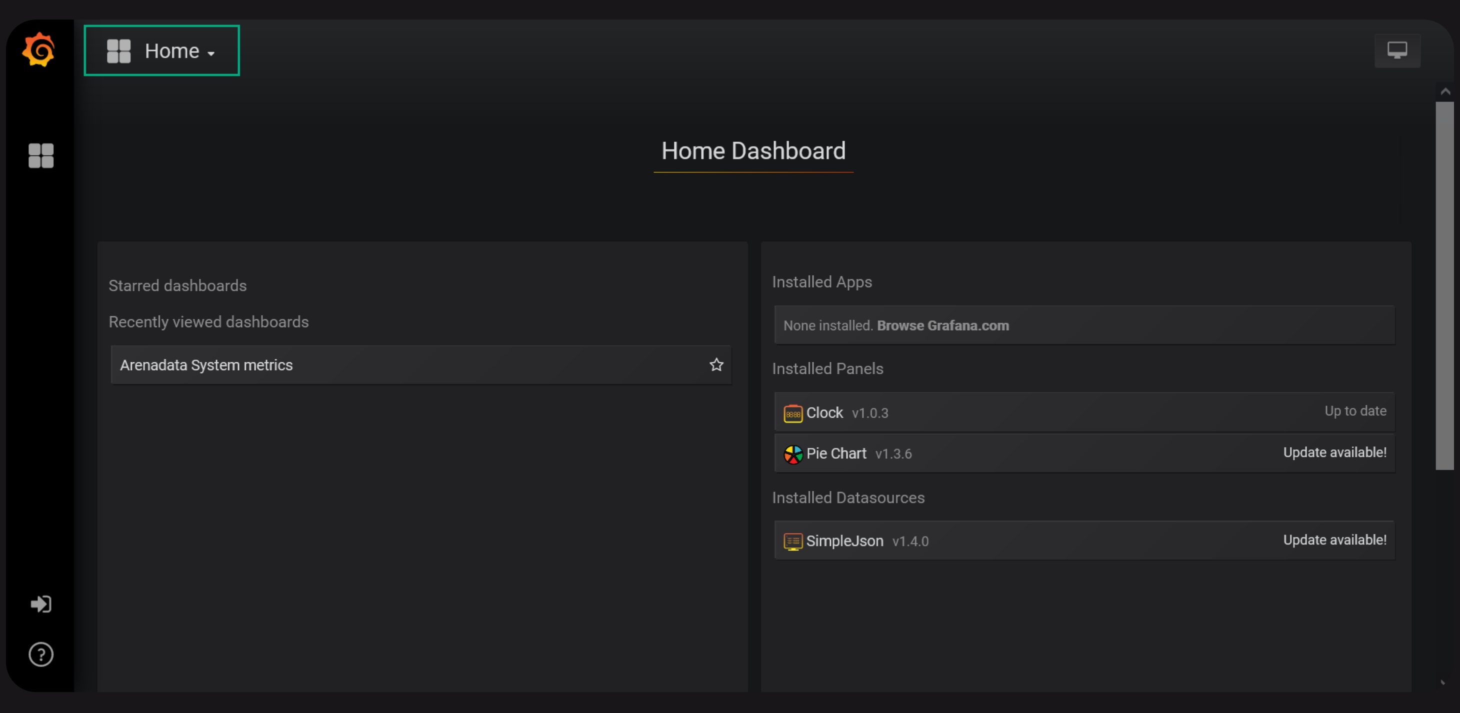
Web-интерфейс Graphite

Grafana

Чтобы проверить доступность web-интерфейса Grafana, необходимо выполнить шаги:

1. В адресной строке браузера введите адрес хоста, на котором развернут кластер мониторинга. В качестве номера порта укажите значение параметра ***Port*** из раздела ***ip_and_ports***, определенное на этапе конфигурирования сервиса Grafana при установке кластера мониторинга (по умолчанию `3000`).

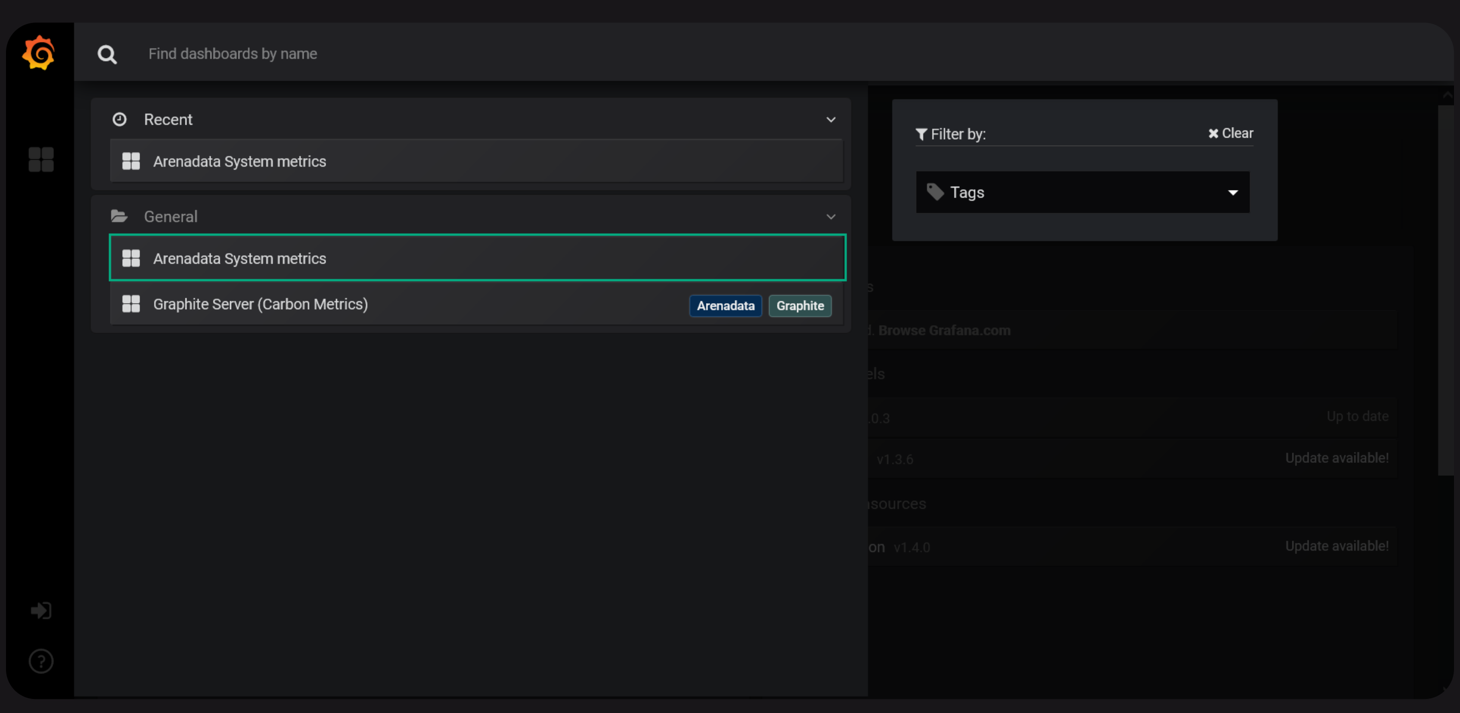
2. Выберите пункт меню ***Home*** в открывшемся окне.



Home
Home Dashboard
Starred dashboards
Recently viewed dashboards
Arenadata System metrics
Installed Apps
None installed. Browse Grafana.com
Installed Panels
Clock v1.0.3
Up to date
Pie Chart v1.3.6
Update available!
Installed Datasources
SimpleJson v1.4.0
Update available!

Web-интерфейс Grafana

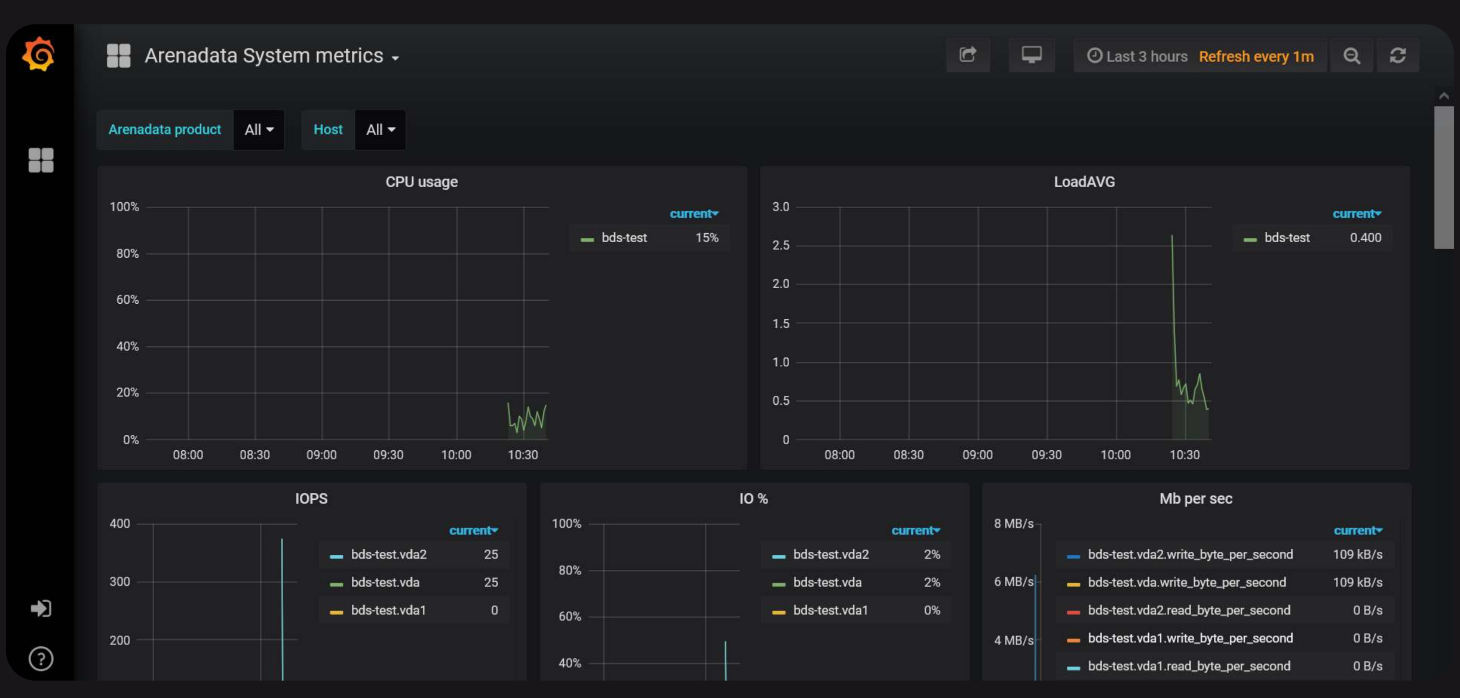
3. Откройте любой из доступных дашбордов (dashboards). Например, ***Arenadata System metrics***.



Find dashboards by name
Recent
Arenadata System metrics
General
Arenadata System metrics
Graphite Server (Carbon Metrics)
Arenadata
Graphite
Filter by:
Clear
Tags

Переход к метрикам Arenadata System

4. Проанализируйте доступные графики.



Arenadata System metrics
Last 3 hours
Refresh every 1m
Arenadata product All
Host All
CPU usage
current
bds-test 15%
LoadAVG
current
bds-test 0.400
IOPS
current
bds-test.vda2 25
bds-test.vda 25
bds-test.vda1 0
IO %
current
bds-test.vda2 2%
bds-test.vda 2%
bds-test.vda1 0%
Mb per sec
current
bds-test.vda2.write_byte_per_second 109 kB/s
bds-test.vda.write_byte_per_second 109 kB/s
bds-test.vda2.read_byte_per_second 0 B/s
bds-test.vda1.write_byte_per_second 0 B/s
bds-test.vda1.read_byte_per_second 0 B/s

Просмотр метрик Arenadata System

Интеграция с кластером ADH

Евгения Кузина, Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Импорт настроек мониторинга

Шаг 2. Добавление сервиса мониторинга

Шаг 3. Распределение компонентов мониторинга

Шаг 4. Установка сервиса мониторинга

Шаг 5. Проверка результатов

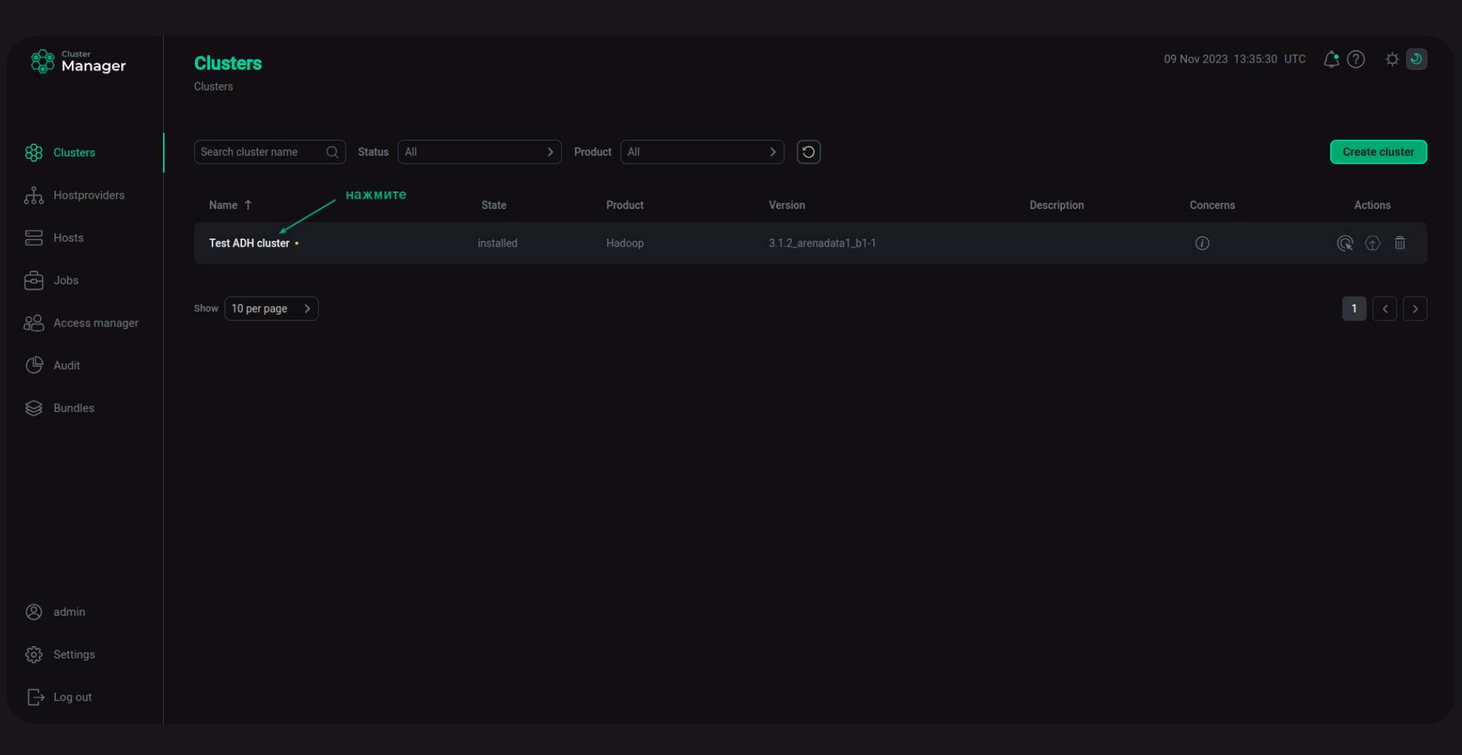
- Graphite
- Grafana

Для настройки мониторинга нового кластера необходимо интегрировать его с предварительно созданным кластером мониторинга и добавить отдельный сервис Monitoring .

Шаги для интеграции кластера мониторинга и кластера ADH описаны ниже. Все действия выполняются на базе кластера ADH.

Шаг 1. Импорт настроек мониторинга

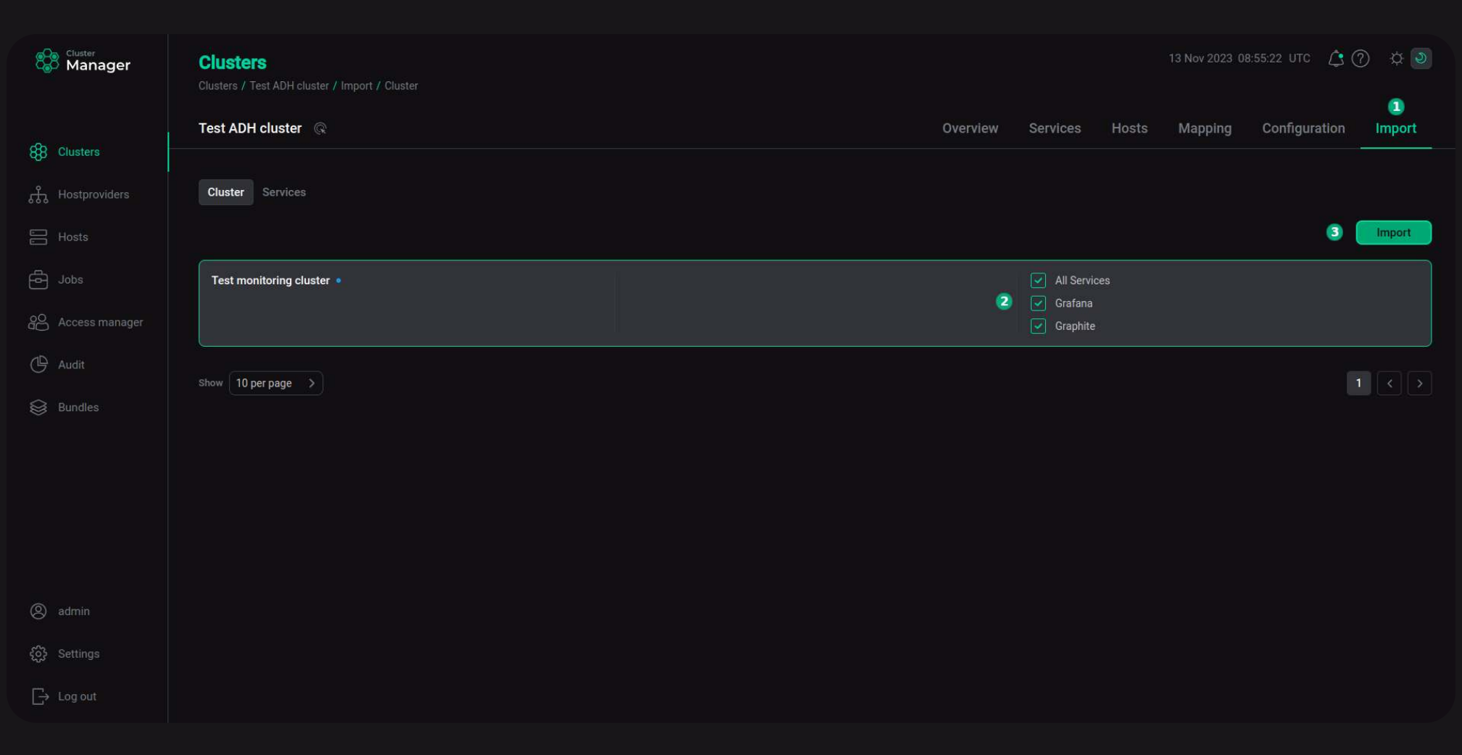
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Create cluster
нажмите

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1			

Выбор кластера

2. Откройте вкладку ***Import*** на странице кластера.
3. Выберите сервисы ***Graphite*** и ***Grafana*** ранее добавленного кластера мониторинга.
4. Нажмите ***Import***.


Clusters
Clusters / Test ADH cluster / Import / Cluster
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Cluster
Services
Import
1
2
3

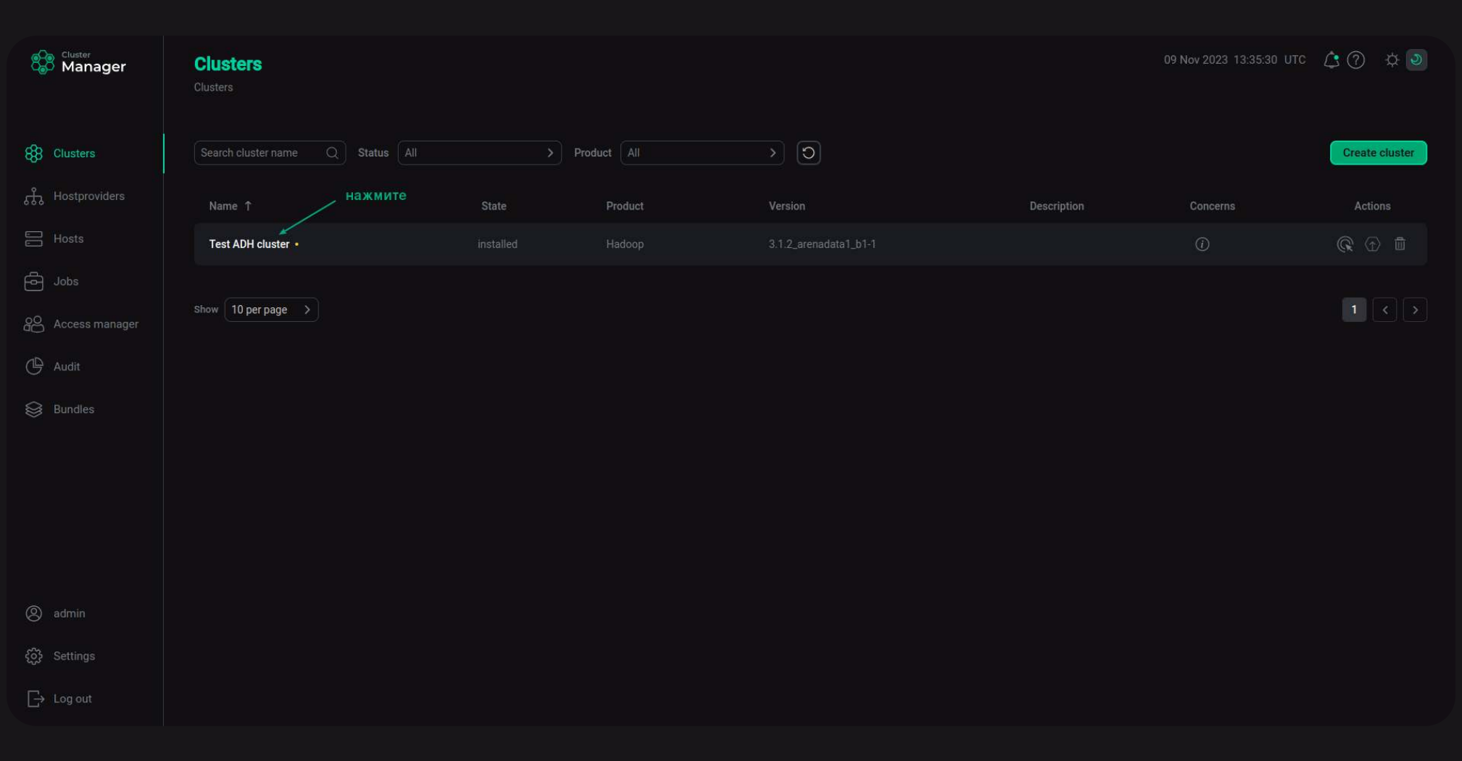
Test monitoring cluster
☑ All Services
☑ Grafana
☑ Graphite

Интеграция с кластером мониторинга

Шаг 2. Добавление сервиса мониторинга

Для добавления сервисов в кластер:

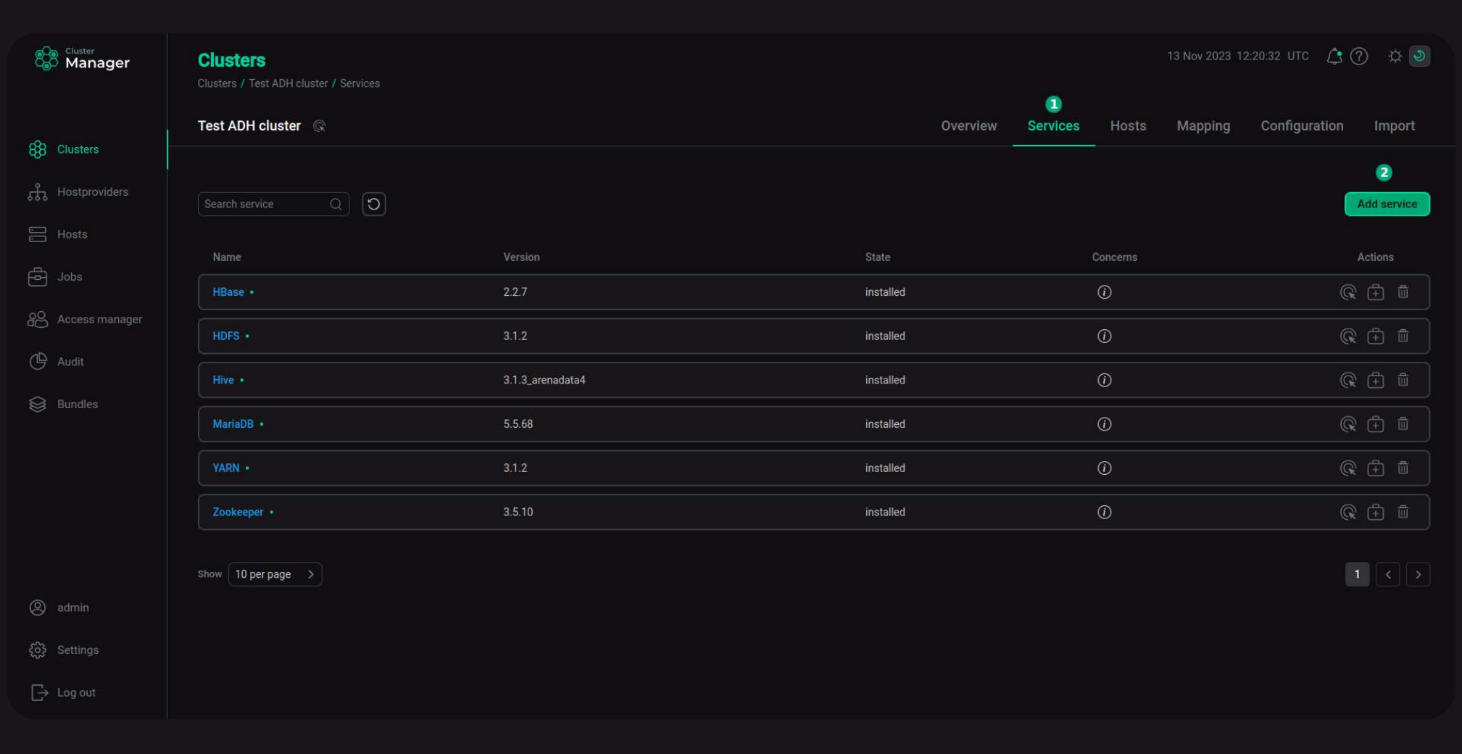
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Create cluster
нажмите

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1			

Выбор кластера

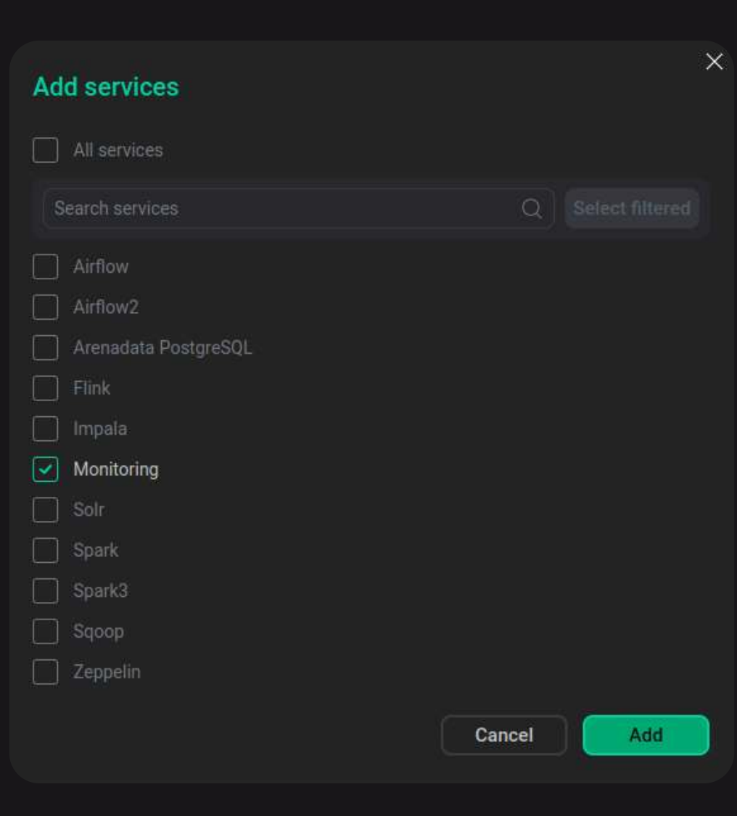
2. Откройте вкладку ***Services*** на странице кластера и нажмите ***Add service***.


Clusters
Clusters / Test ADH cluster / Services
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Add service
1
2

Name	Version	State	Concerns	Actions
HBase	2.2.7	installed		
HDFS	3.1.2	installed		
Hive	3.1.3_arenadata4	installed		
MariaDB	5.5.68	installed		
YARN	3.1.2	installed		
Zookeeper	3.5.10	installed		

Переход к добавлению сервисов

3. Выберите сервис Monitoring в открывшемся окне и нажмите ***Add***.



Add services

☐ All services

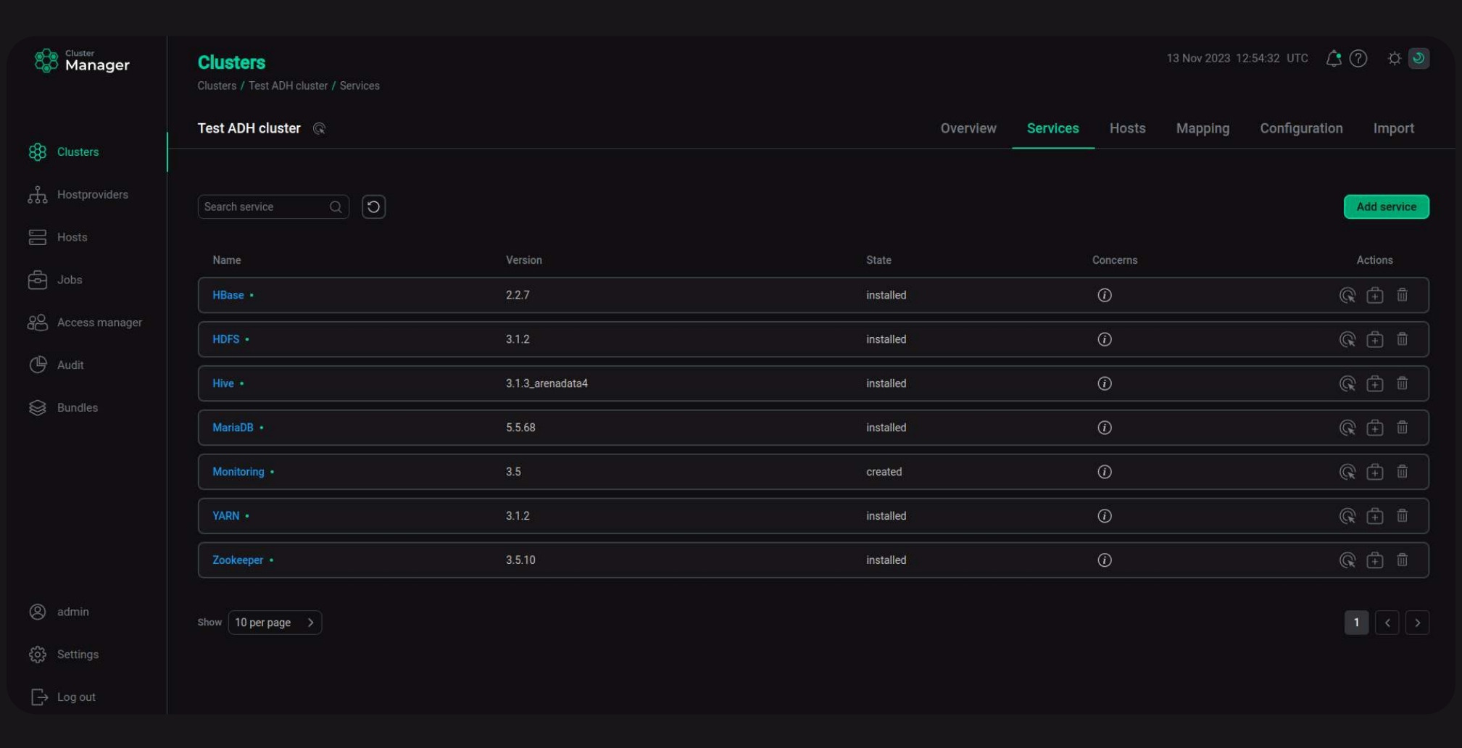
Search services | Select filtered

☐ Airflow
☐ Airflow2
☐ Arenadata PostgreSQL
☐ Flink
☐ Impala
☑ Monitoring
☐ Solr
☐ Spark
☐ Spark3
☐ Sqoop
☐ Zeppelin

Cancel | Add

Выбор сервисов

4. В результате успешно добавленные сервисы отображаются на вкладке ***Services***.


Clusters
Clusters / Test ADH cluster / Services
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Add service

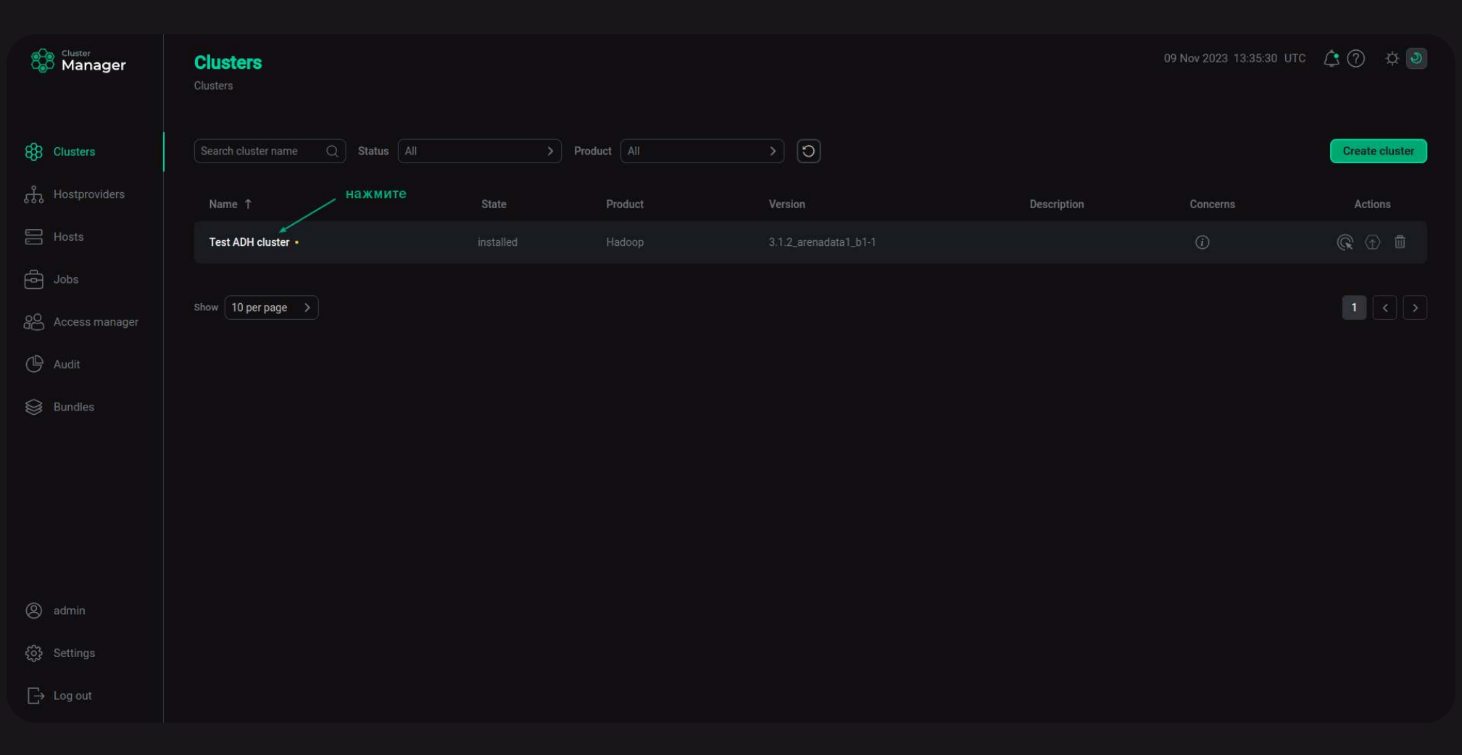
Name	Version	State	Concerns	Actions
HBase	2.2.7	installed		
HDFS	3.1.2	installed		
Hive	3.1.3_arenadata4	installed		
MariaDB	5.5.68	installed		
Monitoring	3.5	created		
YARN	3.1.2	installed		
Zookeeper	3.5.10	installed		

Результат успешного добавления сервисов в кластер

Шаг 3. Распределение компонентов мониторинга

Для добавления компонентов на хосты кластера:

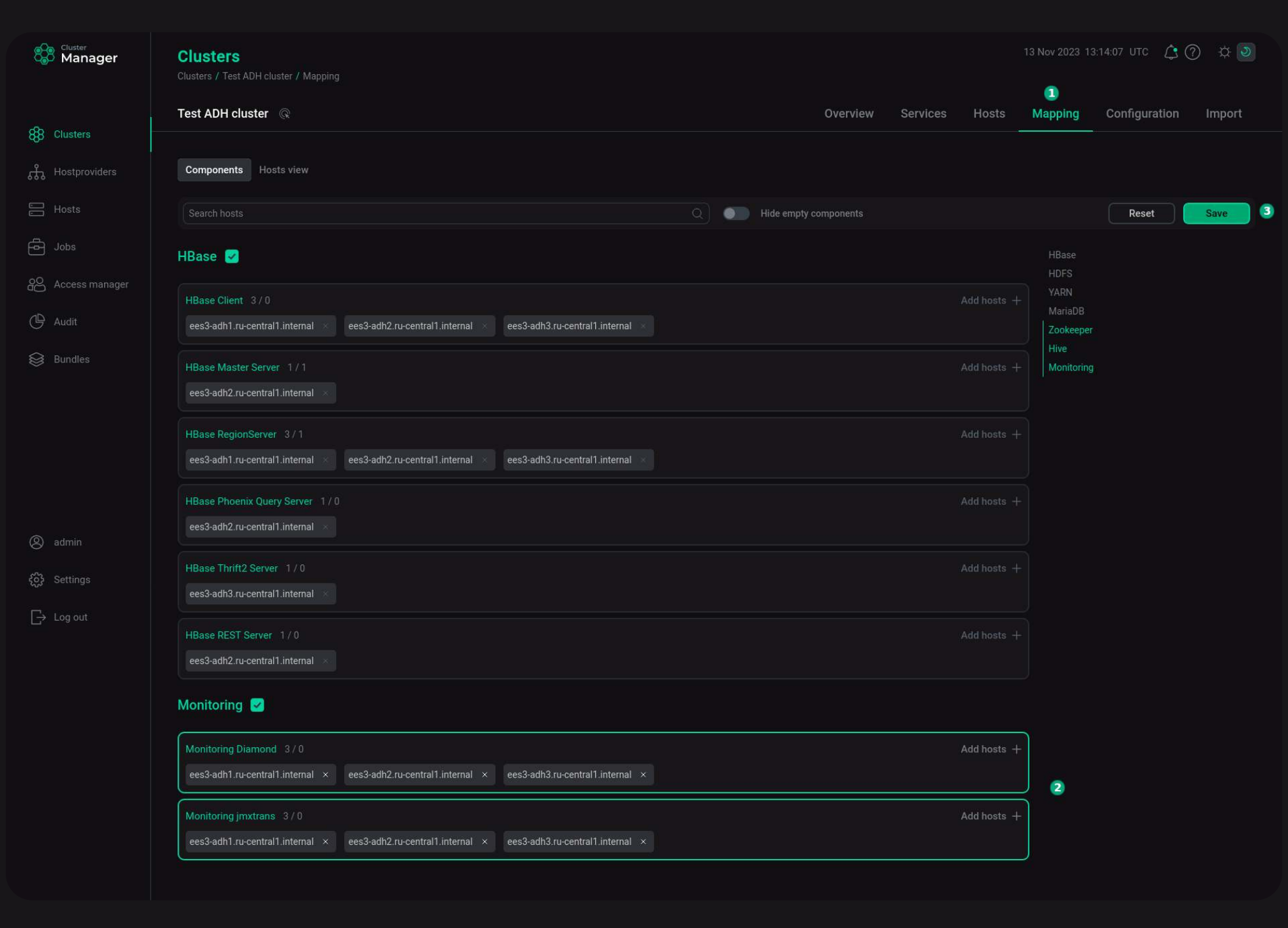
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Create cluster
нажмите

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1			

Выбор кластера

2. Откройте вкладку ***Mapping*** на странице кластера, добавьте компоненты мониторинга на все хосты кластера и нажмите ***Save***.


Clusters
Clusters / Test ADH cluster / Mapping
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Components
Hosts view
Hide empty components
Reset
Save
HBase
HDFS
YARN
MariaDB
Zookeeper
Hive
Monitoring
Add hosts
1
2
3

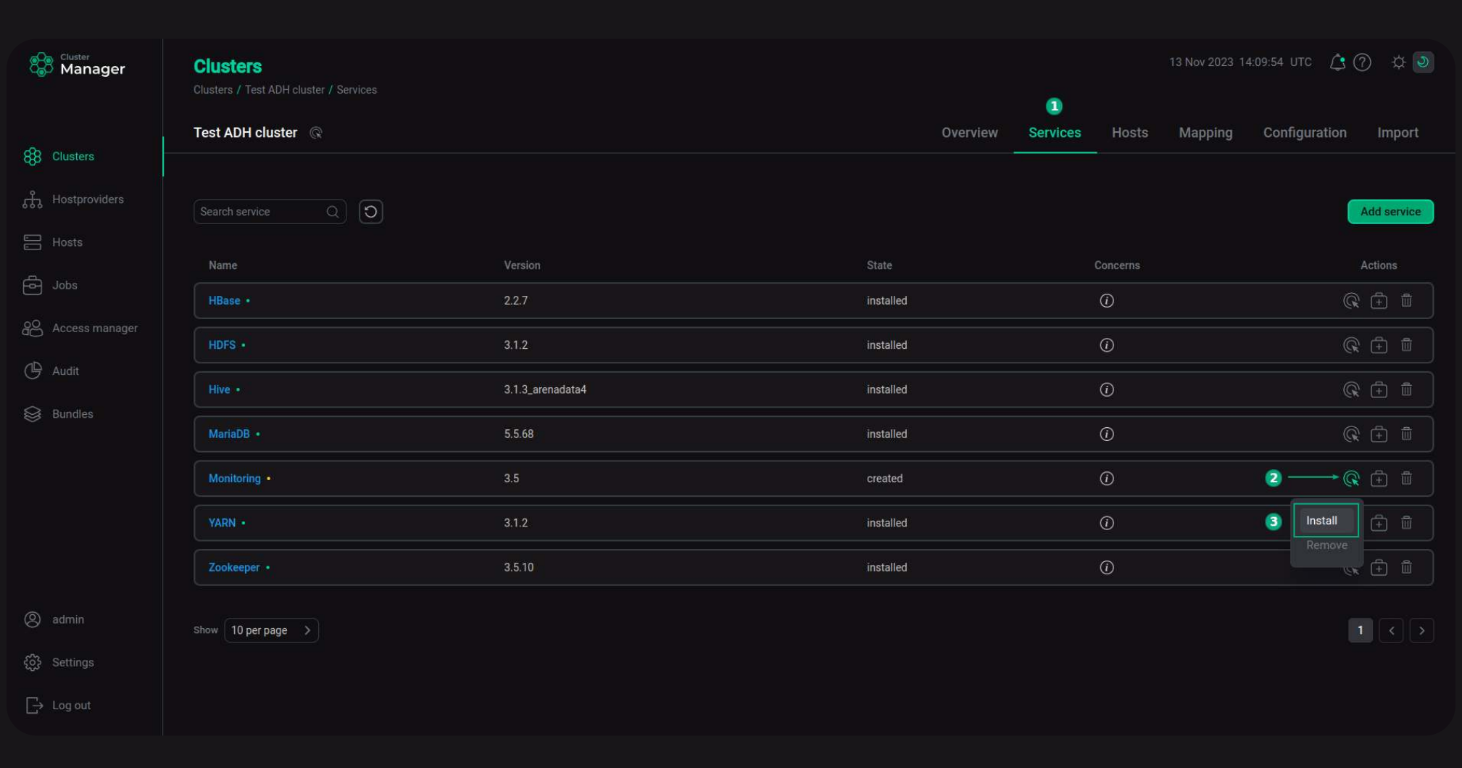
HBase
HBase Client 3 / 0: ees3-adh1.ru-central1.internal, ees3-adh2.ru-central1.internal, ees3-adh3.ru-central1.internal
HBase Master Server 1 / 1: ees3-adh2.ru-central1.internal
HBase RegionServer 3 / 1: ees3-adh1.ru-central1.internal, ees3-adh2.ru-central1.internal, ees3-adh3.ru-central1.internal
HBase Phoenix Query Server 1 / 0: ees3-adh2.ru-central1.internal
HBase Thrift2 Server 1 / 0: ees3-adh3.ru-central1.internal
HBase REST Server 1 / 0: ees3-adh2.ru-central1.internal

Monitoring
Monitoring Diamond 3 / 0: ees3-adh1.ru-central1.internal, ees3-adh2.ru-central1.internal, ees3-adh3.ru-central1.internal
Monitoring jmxtrans 3 / 0: ees3-adh1.ru-central1.internal, ees3-adh2.ru-central1.internal, ees3-adh3.ru-central1.internal

Распределение компонентов между хостами

Шаг 4. Установка сервиса мониторинга

1. Откройте вкладку ***Services*** на странице кластера. Нажмите на иконку ◎ в столбце ***Actions*** и выберите действие ***Install*** в строке, содержащей сервис Monitoring ; подтвердите действие в открывшемся окне.

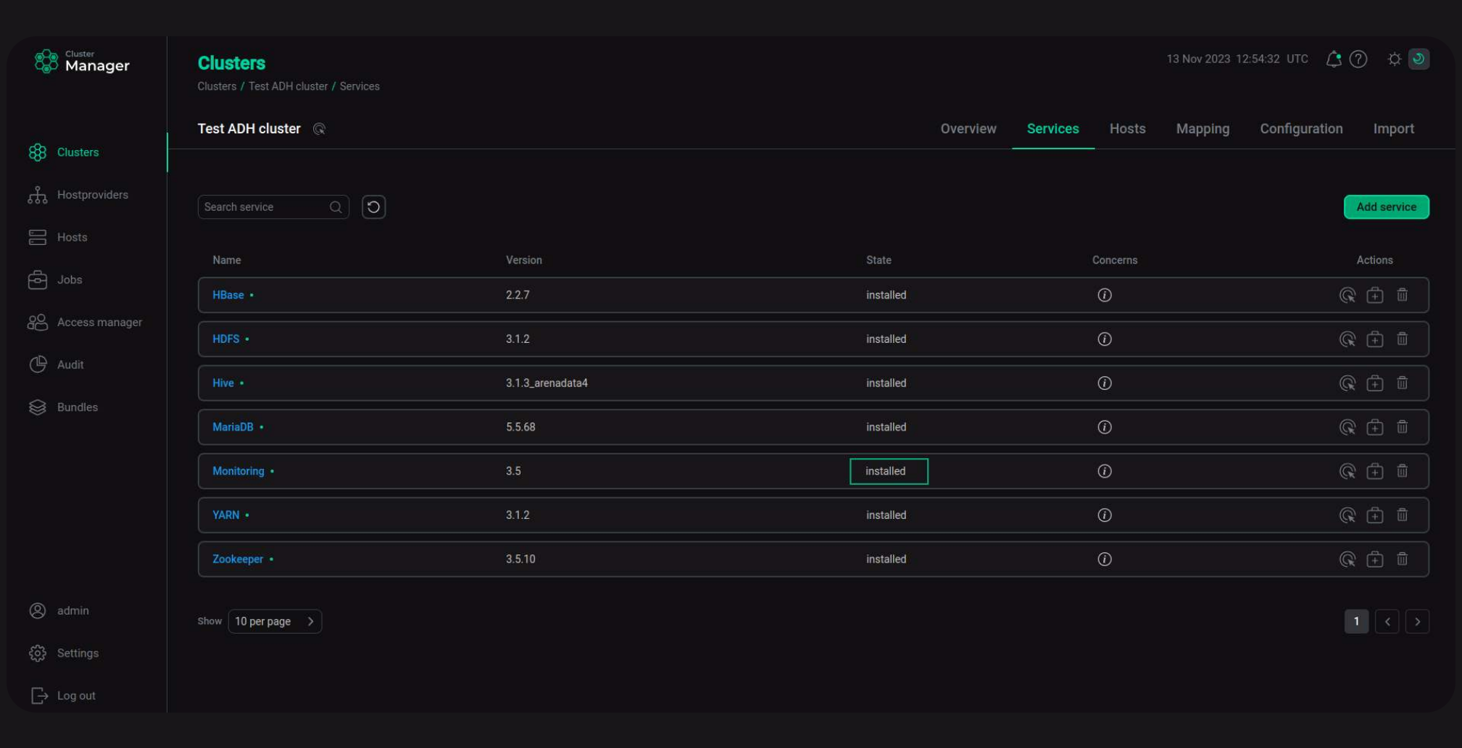

Clusters
Clusters / Test ADH cluster / Services
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Add service
Install
Remove
1
2
3

Name	Version	State	Concerns	Actions
HBase	2.2.7	installed		
HDFS	3.1.2	installed		
Hive	3.1.3_arenadata4	installed		
MariaDB	5.5.68	installed		
Monitoring	3.5	created		
YARN	3.1.2	installed		
Zookeeper	3.5.10	installed		

Переход к установке сервиса мониторинга

2. Дождитесь завершения процесса установки. В результате статус сервиса мониторинга изменяет свое значение с `created` на `installed` .

> **ПРИМЕЧАНИЕ**
>
> В случае ошибок логи можно найти на странице ***Jobs***.


Clusters
Clusters / Test ADH cluster / Services
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Add service

Name	Version	State	Concerns	Actions
HBase	2.2.7	installed		
HDFS	3.1.2	installed		
Hive	3.1.3_arenadata4	installed		
MariaDB	5.5.68	installed		
Monitoring	3.5	installed		
YARN	3.1.2	installed		
Zookeeper	3.5.10	installed		

Процесс установки завершен

> **ВНИМАНИЕ**
>
> После установки кластера мониторинга необходимо перезапустить кластер ADH, чтобы просматривать метрики.

Шаг 5. Проверка результатов

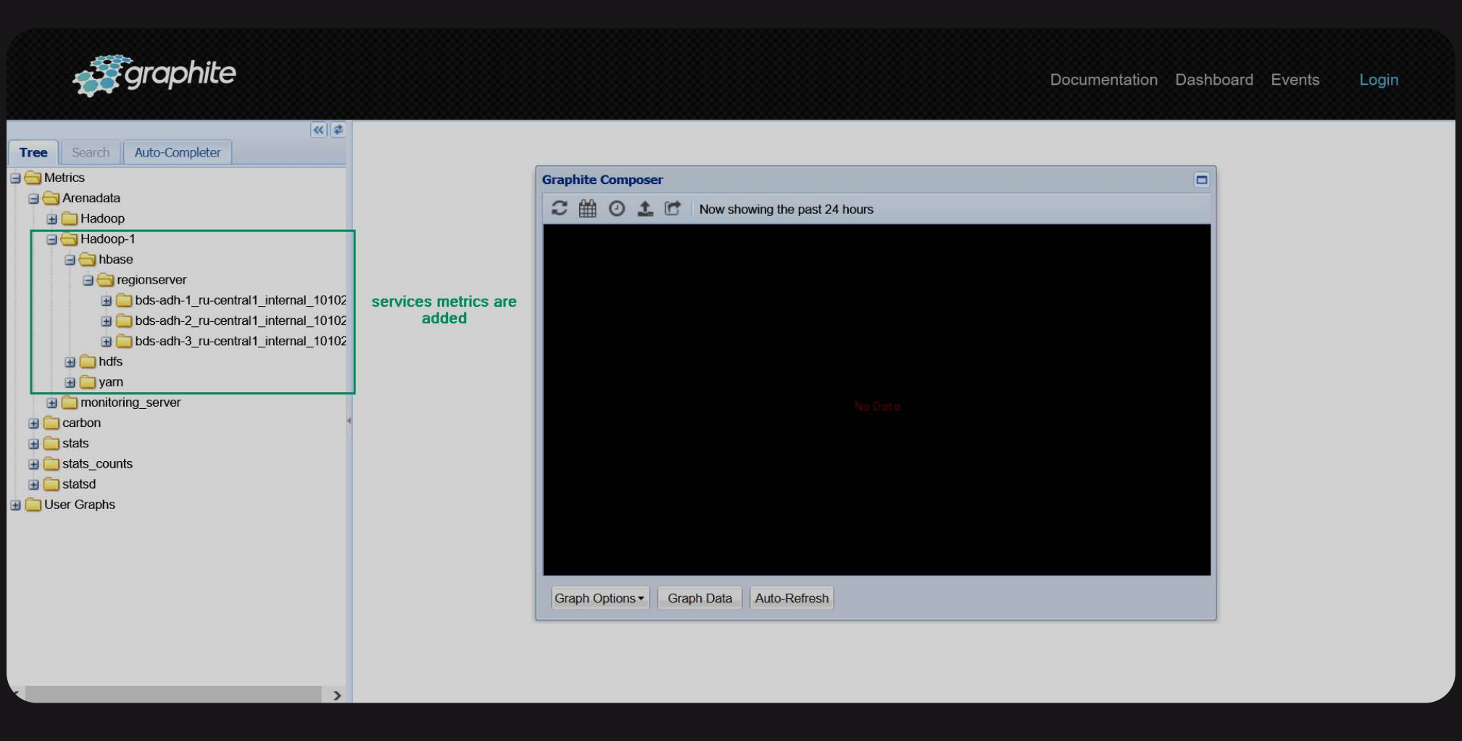
По завершении установки необходимо убедиться в следующем:

- Системные метрики собираются с хостов вашего кластера, а не только с хоста, на котором ранее был развернут кластер мониторинга.
- Наряду с системными метриками, с хостов кластера поступают сервисные метрики. Например, в кластерах ADH поддерживаются метрики для таких сервисов, как HDFS, YARN, HBase, Spark3.

Оба пункта можно проверить в web-интерфейсе Graphite и Grafana.

Graphite

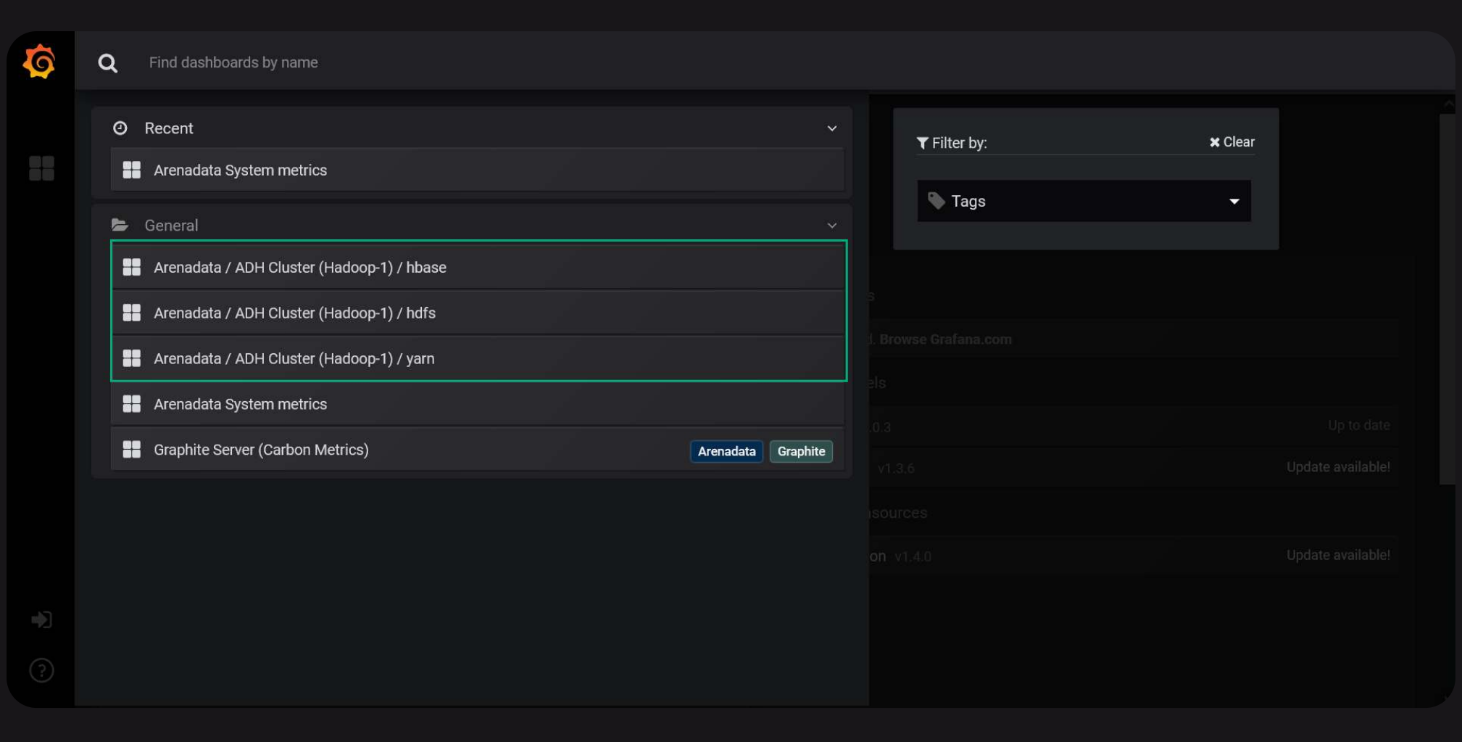
1. В адресной строке браузера введите адрес хоста, на котором развернут кластер мониторинга. В качестве номера порта укажите значение параметра ***Web-interface TCP port*** из раздела ***ip_and_ports***, определенное на этапе конфигурирования сервиса Graphite при установке кластера мониторинга. Значение `80` (используемое по умолчанию) можно не указывать.
2. Разверните дерево в левой части открывшейся формы. Здесь отображаются метрики System и сервисные метрики с группировкой по хостам. Можно выбрать любую из метрик для просмотра динамики ее изменения в виде графика в правой части экрана.


graphite
Documentation
Dashboard
Events
Login
Tree
Search
Auto-Completer
Metrics
Arenadata
Hadoop
Hadoop-1
hbase
regionserver
bds-adh-1_ru-central1_internal_10102
bds-adh-2_ru-central1_internal_10102
bds-adh-3_ru-central1_internal_10102
hdfs
yarn
monitoring_server
carbon
stats
stats_counts
statsd
User Graphs
services metrics are added
Graphite Composer
Now showing the past 24 hours
Graph Options
Graph Data
Auto-Refresh

Новые хосты и сервисные метрики доступны в Graphite

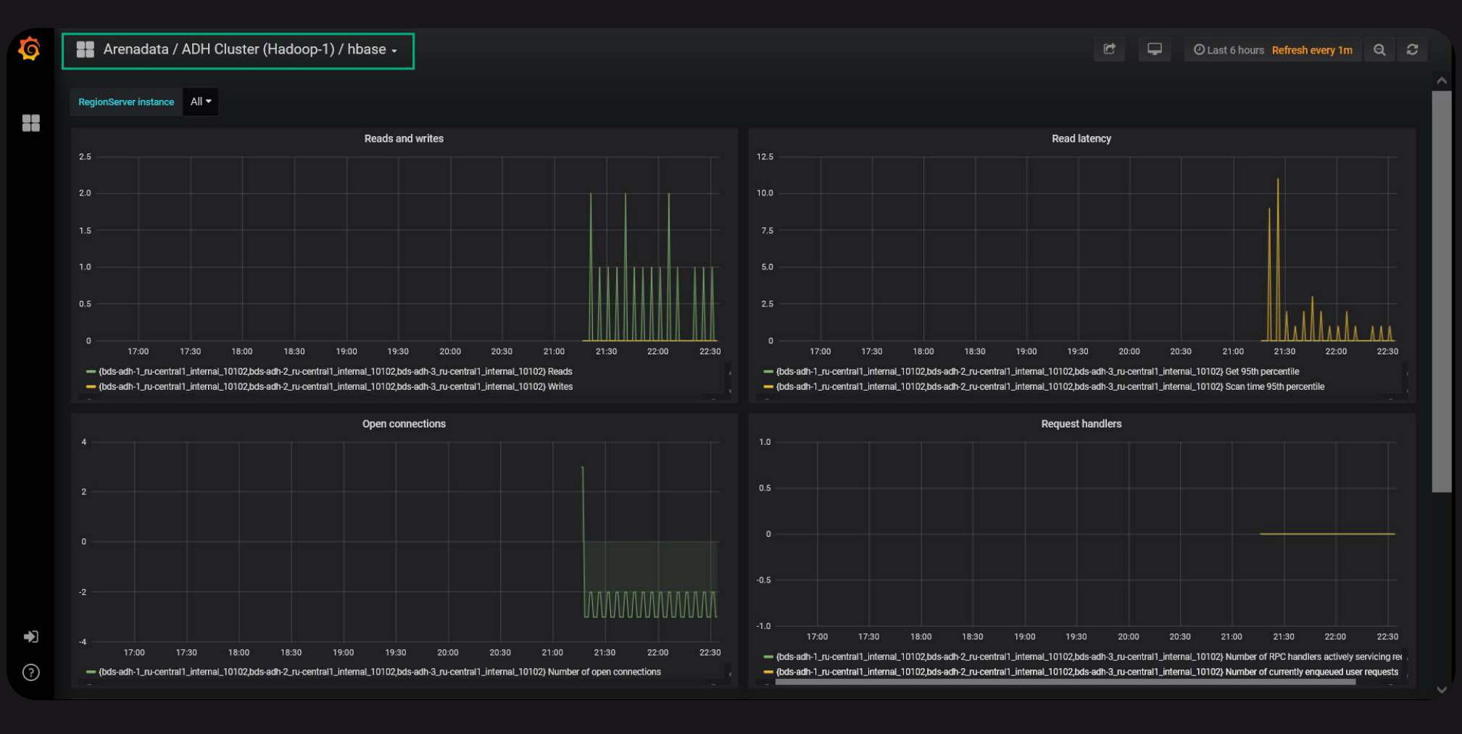
Grafana

1. В адресной строке браузера введите адрес хоста, на котором развернут кластер мониторинга. В качестве номера порта укажите значение параметра ***Port*** из раздела ***ip_and_ports***, определенное на этапе конфигурирования сервиса Grafana при установке кластера мониторинга (по умолчанию `3000`).
2. Выберите пункт меню ***Home*** в открывшемся окне.
3. Обратите внимание, что новые дашборды доступны. Можно выбрать один из них для просмотра сервисных метрик, поступающих из вашего кластера.


Find dashboards by name
Recent
Arenadata System metrics
General
Arenadata / ADH Cluster (Hadoop-1) / hbase
Arenadata / ADH Cluster (Hadoop-1) / hdfs
Arenadata / ADH Cluster (Hadoop-1) / yarn
Arenadata System metrics
Graphite Server (Carbon Metrics)
Arenadata
Graphite
Filter by:
Clear
Tags

Переход к сервисным метрикам

4. Проанализируйте доступные графики.


Arenadata / ADH Cluster (Hadoop-1) / hbase
Last 6 hours
Refresh every 1m
RegionServer instance
All
Reads and writes
Read latency
Open connections
Request handlers

Просмотр сервисных метрик

Offline-установка

Назад к содержанию

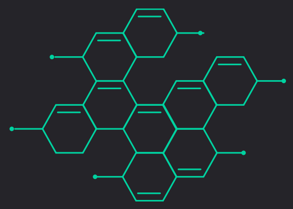
Этот раздел посвящен **offline**-установке ADH. Данный способ установки подходит для хостов без доступа к Интернету.

Для offline-установки необходимо установить и настроить Arenadata Enterprise Tools (ET). ET — это решение, предоставляющее инфраструктуру для развертывания продуктов Аренадата в среде с ограниченным доступом в интернет. Решение является частью платформы данных Аренадата и предоставляет репозитории, необходимые для установки и обновления компонентов платформы.

Процесс offline-установки через ADCM включает следующие шаги.

ПРИМЕЧАНИЕ

Эта функция доступна только в Enterprise-версии ADH.



Шаг 1.

Установка ADCM

Шаг 2.

Подготовка хостов

Шаг 3.

Установка кластера ET

ARENADATA
HADOOP

Шаг 4.

Установка кластера ADH

Шаг 5.

Установка мониторинга

Установка ADCM

Дарья Барышева

Содержание

Назад к содержанию

- Шаг 1. Предварительные требования
  - Требования к software и hardware
  - Установка Docker
  - Отключение SELinux
  - Остановка firewalld
  - Настройка доступа к Docker CLI без прав root
- Шаг 2. Установка ADCM
- Шаг 3. Запуск ADCM
- Шаг 4. Проверка установки
  - Проверка статуса Docker-контейнера
  - Проверка доступности порта 8000
  - Проверка web-интерфейса ADCM
  - Установка URL ADCM
- Шаг 5. Обновление ADCM

Для развертывания ADCM требуется отдельный хост. Перед началом установки рекомендуем ознакомиться с документацией ADCM.

Этапы установки описаны ниже.

Шаг 1. Предварительные требования

Требования к software и hardware

ADCM поставляется в виде Docker-образа. Для его установки необходимы:

- Одна из приведенных ниже операционных систем:
  - Ubuntu 22.04
  - CentOS 7
  - RHEL 7
  - Alt Linux SP 8
  - Alt Linux SP 10.2
  - Astra Linux SE 1.7 Орел
  - Astra Linux 1.7.6 Воронеж
  - РЕД ОС 7.3 Сертифицированная редакция
- Пользователь с правами `sudo`.
- Доступ к официальным репозиториям CentOS Extras и CentOS Base. По умолчанию, эти репозитории доступны в операционной системе. Однако, если они были отключены, потребуется их предварительное включение (например, путем редактирования конфигурационного файла в директории */etc/yum.repos.d/*).
- Менеджер программных пакетов YUM/APT.
- Хост, отвечающий следующим требованиям:

> **ВАЖНО**
>
> Системные требования, приведенные ниже, являются **минимальными**. Целевой сайзинг необходимо рассчитывать исходя из требований вашей организации.

  - **CPU**: 4 ядра CPU;
  - **RAM**: 16 ГБ;
  - **HDD**: 50 ГБ.

> **ВАЖНО**
>
> Начиная с версии ADCM 2.7.0 для установки ADPG без подключения к Интернету и использования его в качестве базы данных ADCM требуется любой Container Registry Client с загруженными образами ADPG, ADCM и ADCM Installer. Установка выполняется с помощью ADCM Installer с указанием URL локального хранилища образов ADCM и ADPG.

Чтобы пользователь обладал необходимыми правами для установки ADCM, выполните следующее:

1. Предоставьте пользователю привилегии `root`:

CentOS 7/RHEL 7/РЕД ОС 7.3/Alt Linux SP 8

```
$ sudo usermod -aG wheel <user>
```

где `<user>` — имя пользователя.

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
$ sudo usermod -aG sudo <user>
```

где `<user>` — имя пользователя.

2. Убедитесь, что в файле */etc/sudoers* есть запись, указанная ниже в зависимости от используемой операционной системы, с помощью следующей команды:

```
$ sudo vi /etc/sudoers
```

Чтобы вернуться в терминал, нажмите `Esc`, введите `:q!` и нажмите `Enter`.

CentOS 7/RHEL 7/РЕД ОС 7.3

```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
```

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Alt Linux SP 8

```
## Uncomment to allow members of group wheel to execute any command
WHEEL_USERS ALL=(ALL:ALL) ALL
```

Установка Docker

> **ВНИМАНИЕ**
>
> Установка Docker с официального сайта не рекомендуется. Вместо этого используйте официальные репозитории дистрибутива Linux. Репозитории, предоставляемые разработчиками операционных систем, как правило, более стабильны и безопасны.

Ниже приведены шаги, необходимые для установки Docker, на примере нескольких поддерживаемых в ADCM операционных систем. Для других версий операционных систем, отличных от указанных во вкладках, шаги могут отличаться.

CentOS 7/RHEL 7

1. Обновите все программные пакеты, установленные в системе:

```
$ sudo yum update -y
```

2. Установите пакеты, необходимые для Docker:

```
$ sudo yum install -y yum-utils docker device-mapper-persistent-data lvm2
```

где:

- `yum-utils` — пакет, который предоставляет утилиту `yum-config-manager`;
- `device-mapper-persistent-data` и `lvm2` — пакеты, необходимые драйверу хранилища devicemapper.

3. Запустите Docker:

```
$ sudo systemctl start docker
```

4. Включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable docker
```

Ubuntu 22.04

1. Обновите все программные пакеты, установленные в системе:

```
$ sudo apt-get update
```

2. Установите пакеты, необходимые для Docker:

```
$ sudo apt-get install ca-certificates curl
```

где:

- `ca-certificates` — пакет корневых сертификатов;
- `curl` — пакет для работы с веб-ресурсами.

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

3. Добавьте GPG-ключ для Docker-репозитория:

```
$ sudo install -m 0755 -d /etc/apt/keyrings
$ sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
$ sudo chmod a+r /etc/apt/keyrings/docker.asc
```

4. Добавьте Docker-репозиторий:

```
$ echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \
$(. /etc/os-release && echo "jammy") stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

5. Обновите все программные пакеты, в том числе пакеты для добавленного Docker-репозитория:

```
$ sudo apt-get update
```

6. Установите Docker Engine:

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

7. Запустите Docker:

```
$ sudo systemctl start docker
```

8. Включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable docker
```

Alt Linux SP 8

1. Обновите все программные пакеты, установленные в системе:

```
$ sudo apt-get update
```

2. Установите Docker Engine:

```
$ sudo apt-get install docker-engine
```

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

3. Запустите и включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable --now docker
```

Astra Linux SE 1.7 Орел

1. Обновите все программные пакеты, установленные в системе:

```
$ sudo apt-get update
```

2. Установите пакеты, необходимые для Docker:

```
$ sudo apt-get install ca-certificates curl net-tools
```

где:

- `ca-certificates` — пакет корневых сертификатов;
- `curl` — пакет для работы с веб-ресурсами;
- `net-tools` — пакет для управления сетевой подсистемой ядра Linux, в частности для запуска команды `netstat`.

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

3. Добавьте GPG-ключ для Docker-репозитория:

```
$ curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
```

4. Добавьте Docker-репозиторий:

```
$ echo "deb [arch=amd64] https://download.docker.com/linux/debian stretch stable" | sudo tee -a /etc/apt/sources.list
```

5. Обновите все программные пакеты, в том числе пакеты для добавленного Docker-репозитория:

```
$ sudo apt-get update
```

6. Установите Docker Engine:

```
$ sudo apt install docker-ce docker-ce-cli containerd.io
```

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

7. Запустите Docker:

```
$ sudo systemctl start docker
```

8. Включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable docker
```

РЕД ОС 7.3

1. Установите Docker Engine:

```
$ sudo dnf install docker-ce docker-ce-cli
```

При выполнении команды подтвердите увеличение объема занятого дискового пространства.

2. Запустите и включите Docker в качестве системного сервиса:

```
$ sudo systemctl enable docker --now
```

Отключение SELinux

> **ПРИМЕЧАНИЕ**
>
> Отключение системы безопасности SELinux необходимо только в случае использования операционной системы CentOS 7 или RHEL 7.

Для долговременного отключения SELinux следует прописать `SELINUX=disabled` в конфигурационном файле */etc/selinux/config* и перезагрузить систему.

Файл может быть изменен с помощью редактора `vi` (или `vim`):

```
$ sudo vi /etc/selinux/config
```

Содержимое обновленного файла:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - SELinux is fully disabled.
SELINUX=disabled
# SELINUXTYPE= type of policy in use. Possible values are:
#       targeted - Only targeted network daemons are protected.
#       strict - Full SELinux protection.
SELINUXTYPE=targeted
```

Кроме этого, доступно временное отключение SELinux с помощью следующей команды:

```
$ sudo setenforce 0
```

Тем не менее рекомендуется отключить SELinux на постоянной основе, чтобы предотвратить ее перезапуск при каждой перезагрузке системы.

> **ВНИМАНИЕ**
>
> Не забудьте перезагрузить хост после отключения SELinux.

Остановка firewalld

Если на хосте используется сервис firewalld, требуется его остановка.

Перед остановкой firewalld убедитесь, что он установлен, запущен и включен, выполнив команду:

```
$ sudo systemctl status firewalld
```

Пример результата выполнения команды:

```
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2024-12-02 10:02:57 UTC; 3min 25s ago
     Docs: man:firewalld(1)
 Main PID: 3023 (firewalld)
    Tasks: 2
   Memory: 22.8M
   CGroup: /system.slice/firewalld.service
           └─3023 /usr/bin/python2 -Es /usr/sbin/firewalld --nofork --nopid
```

Остановите firewalld с помощью команды:

```
$ sudo systemctl stop firewalld
```

Альтернативный вариант — отключить применение правил к сетевым пакетам при помощи команды `disable`:

```
$ sudo systemctl disable firewalld
```

Настройка доступа к Docker CLI без прав root

Для настройки доступа к Docker CLI без использования прав `root` необходимо создать группу пользователей `docker` (если она еще не существует) и добавить текущего пользователя в эту группу с определенными правами:

```
$ sudo groupadd docker
$ sudo usermod -a -G docker $USER
```

После запуска команд требуется перезайти в систему под текущим пользователем.

Шаг 2. Установка ADCM

Offline-установка ADCM включает следующие шаги:

1. Запросите у команды поддержки Arenadata архив с расширением *sh.xz*, который содержит специальный установочный shell-скрипт.
2. Создайте директорию на хосте, где должен быть развернут ADCM, и разместите в ней полученный архив.
3. Распакуйте архив в созданной директории любым из доступных способов.

> **ПРИМЕЧАНИЕ**
>
> Выполнение команды может занять некоторое время. Пожалуйста, дождитесь ее завершения.

4. Загрузите Docker-образ в локальное хранилище с помощью полученного скрипта:

```
$ sudo bash <script_name> unpack_adcm
```

где `<script_name>` — полное имя скрипта с расширением (например, *adcm_2.5.0.sh*).

Пример результата успешной обработки команды:

```
Loaded image: hub.arenadata.io/adcm/adcm:2.5.0
```

5. Создайте новый Docker-контейнер на базе загруженного Docker-образа:

```
$ sudo docker create --name adcm -p 8000:8000 -v /opt/adcm:/adcm/data -e DB_HOST="<DATABASE_HOSTNAME_OR_IP_ADDRESS>" -e DB_PORT="
<DATABASE_TCP_PORT>" -e DB_USER="<DATABASE_USERNAME>" -e DB_NAME="<DATABASE_NAME>" -e DB_PASS="<DATABASE_USER_PASSWORD>" -e
DB_OPTIONS="<DATABASE_OPTIONS>" hub.arenadata.io/adcm/adcm:<version>
```

Шаг 3. Запуск ADCM

Поскольку ADCM представляет собой контейнер под управлением Docker, за его запуск и установку отвечают стандартные Docker-команды:

- **Запуск ADCM**.

```
$ sudo docker start adcm
```

- **Остановка ADCM**.

```
$ sudo docker stop adcm
```

- **Автозапуск ADCM**. Чтобы настроить автоматический запуск Docker-контейнера в случае непредвиденных ошибок, используйте следующую команду.

```
$ sudo docker update --restart=on-failure adcm
```

> **ПРИМЕЧАНИЕ**
>
> - Дополнительная информация по политикам перезапуска Docker-контейнеров (restart policies) может быть получена из документации Docker.
> - Если для текущего пользователя настроен доступ к Docker CLI без использования прав `root`, слово `sudo` в предыдущих командах необязательно.

Шаг 4. Проверка установки

Проверка статуса Docker-контейнера

Чтобы убедиться в успешной установке Docker-контейнера `adcm`, выполните следующую команду:

```
$ sudo docker container ls
```

Пример результата выполнения команды приведен ниже. Убедитесь, что в столбце `STATUS` выводится значение `Up`:

```
CONTAINER ID        IMAGE                               COMMAND             CREATED             STATUS              PORTS
NAMES
74b5628146e5        hub.arenadata.io/adcm/adcm:2.0   "/etc/startup.sh"   4 days ago          Up 21 minutes       0.0.0.0:8000-
>8000/tcp   adcm
```

Проверка доступности порта 8000

После установки и запуска ADCM его web-интерфейс должен стать доступным на `8000` порте развернутого контейнера.

Для операционной системы Ubuntu 22.04 предварительно установите пакет для запуска команды `netstat`:

```
$ sudo apt-get install net-tools
```

Для проверки доступности порта выполните команды:

- `netstat` — для вывода информации о состоянии порта.

```
$ sudo netstat -ntpl | grep 8000
```

Пример результата выполнения команды:

```
tcp6       0      0 :::8000                 :::*                    LISTEN
1514/docker-proxy-c
```

- `curl` — для проверки URL-соединения.

```
$ curl http://localhost:8000
```

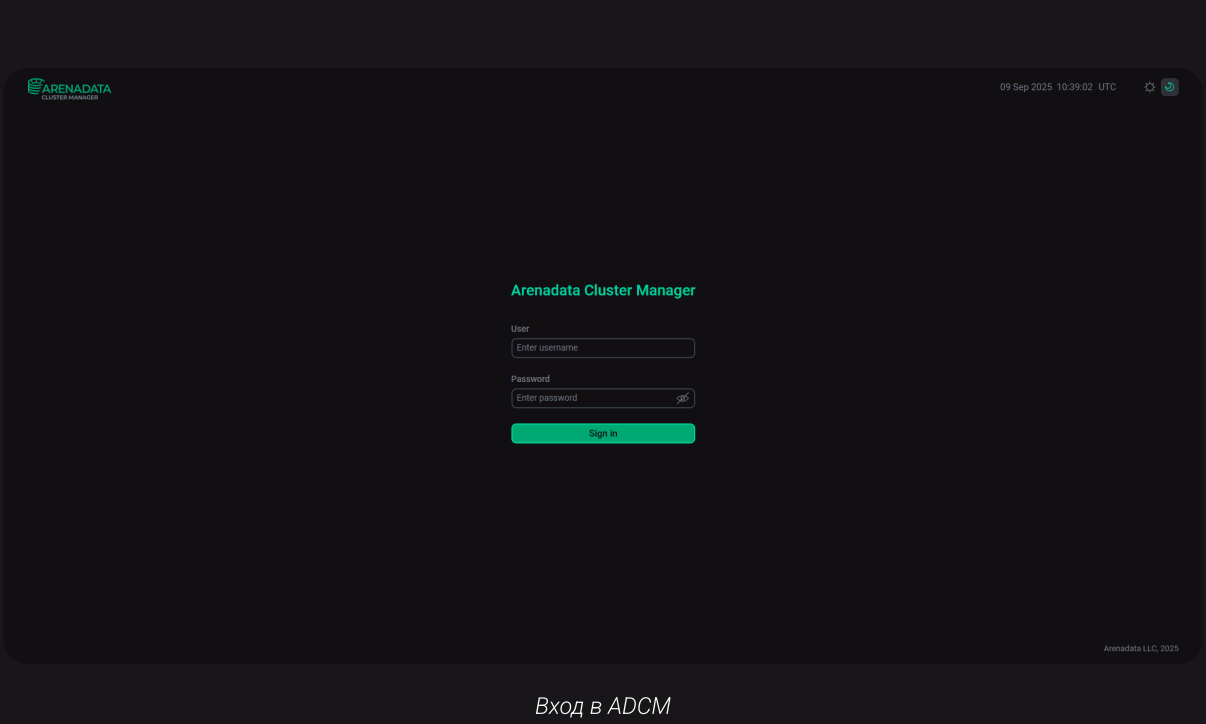
Пример результата выполнения команды:

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Arenadata Cluster Manager</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="assets/favicon.ico">
<link rel="stylesheet" href="styles.10db6328264e0907c52f.css"></head>
<body>
  <app-root></app-root>
<script src="runtime-es2015.7eb406ed18bf0258cd35.js" type="module"></script><script src="runtime-es5.7eb406ed18bf0258cd35.js"
nomodule defer></script><script src="polyfills-es5.2e224d70daec4412d3c2.js" nomodule defer></script><script src="polyfills-es2015.
37e0553ac06970d6a5b5.js" type="module"></script><script src="main-es2015.39851da0ebf9ed6fec45.js" type="module"></script><script
src="main-es5.39851da0ebf9ed6fec45.js" nomodule defer></script></body>
</html>
```

Проверка web-интерфейса ADCM

При установке ADCM автоматически создается пользователь с административными правами. Чтобы зайти в ADCM, используйте следующие учетные данные:

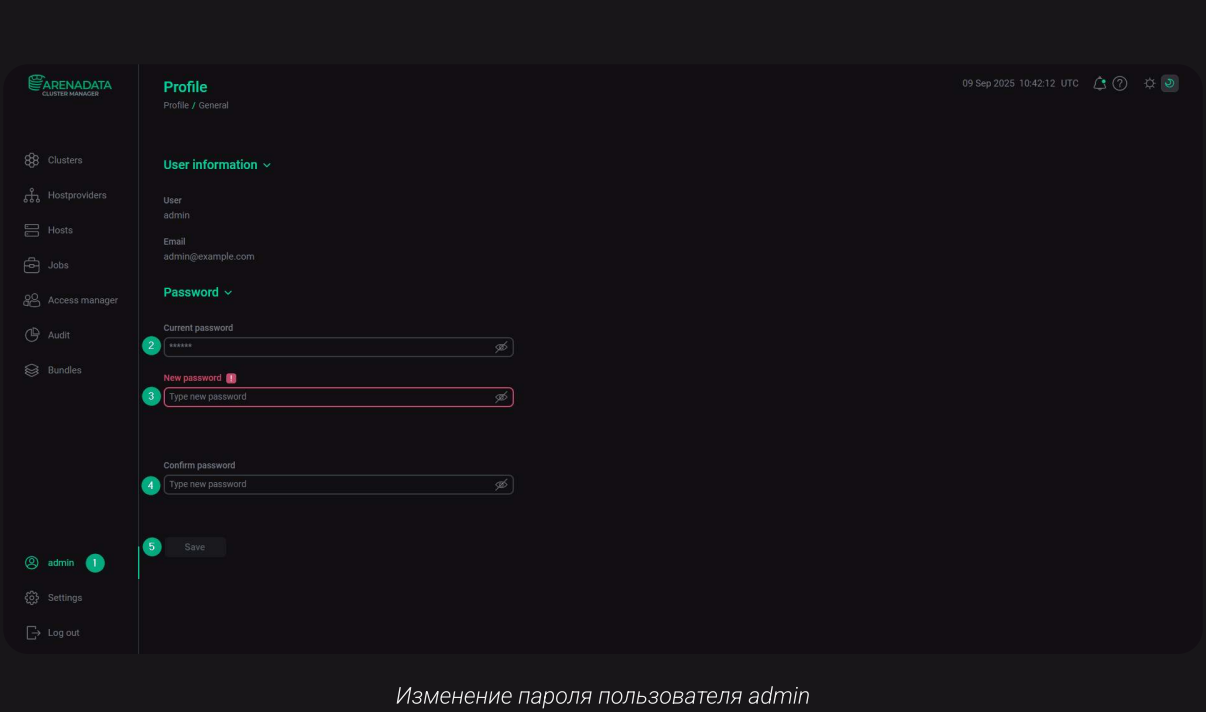
- ***User***: admin
- ***Password***: admin



Вход в ADCM

Для изменения пароля требуется выполнить следующие действия:

1. Откройте профиль текущего пользователя в левом навигационном меню.
2. Введите текущий пароль в поле ***Current password***.
3. Укажите новый пароль в поле ***New password***.
4. Подтвердите новый пароль в поле ***Confirm password***.
5. Нажмите ***Save***.

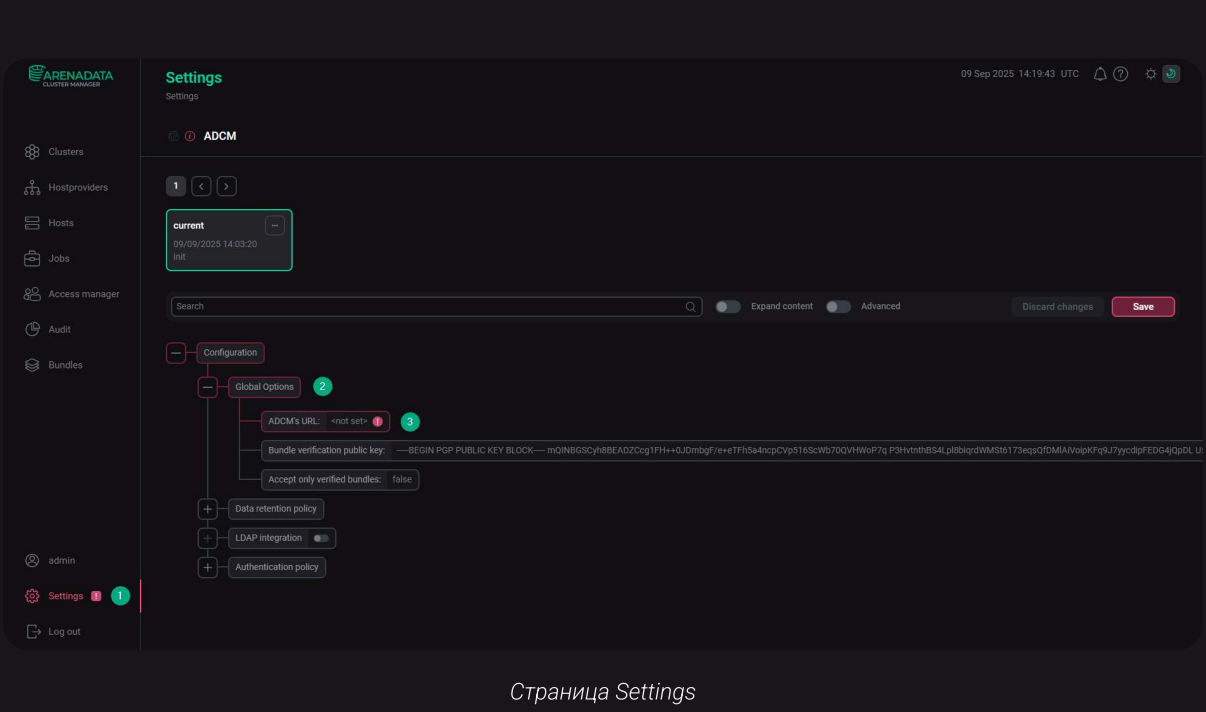


Изменение пароля пользователя admin

Установка URL ADCM

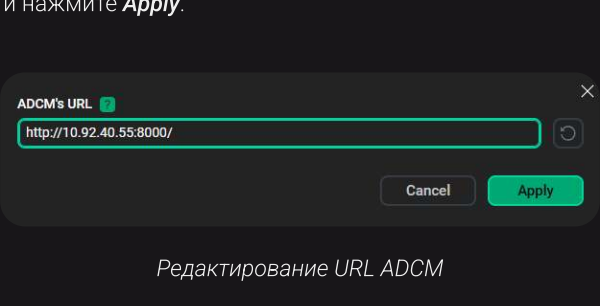
Для успешного обмена данными между ADCM и развернутыми с его помощью кластерами важно корректное определение внешнего адреса ADCM. В дальнейшем этот адрес будет использоваться для отправки в ADCM статусов кластерных компонентов. Для указания URL ADCM используется переменная окружения `DEFAULT_ADCM_URL` при создании контейнера ADCM. Если переменная не была указана, необходимо установить URL вручную в web-интерфейсе ADCM. Для этого выполните шаги:

1. Выберите пункт ***Settings*** в левом навигационном меню.
2. Раскройте узел ***Global Options*** в дереве конфигурационных настроек.
3. Нажмите на значение поля ***ADCM's URL***.



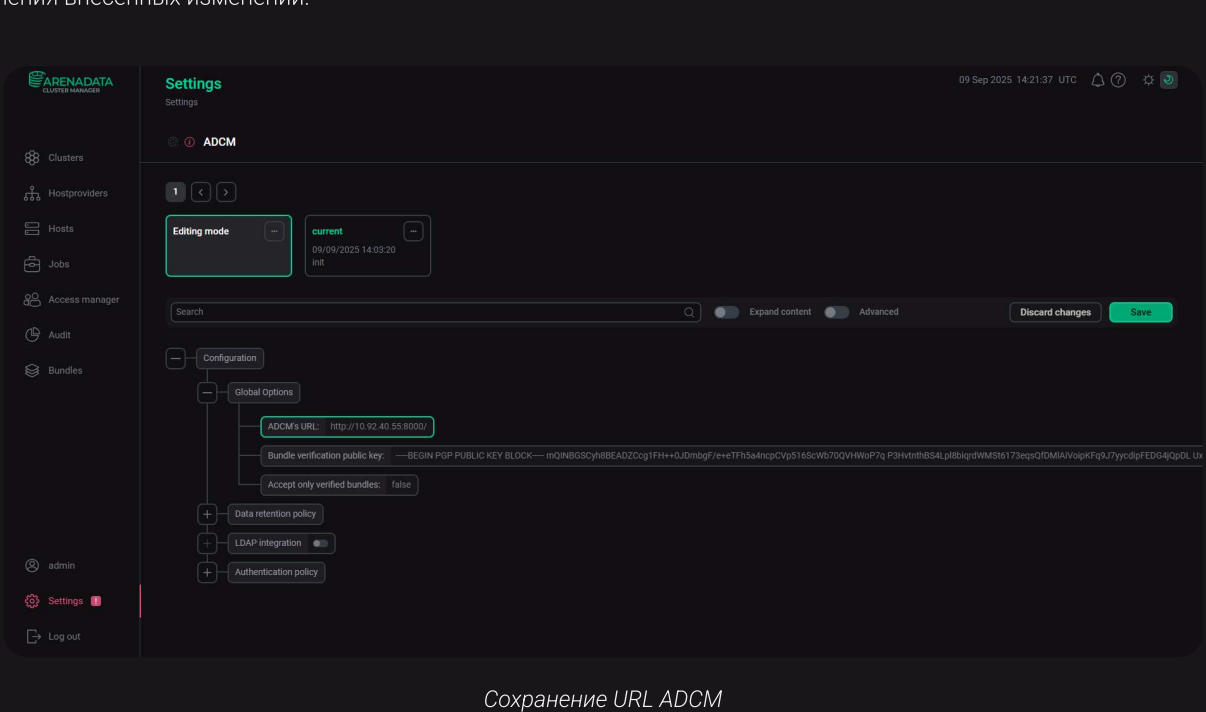
Страница Settings

4. В открывшемся окне измените значение ***ADCM's URL*** и нажмите ***Apply***.



Редактирование URL ADCM

5. Нажмите ***Save*** для сохранения внесенных изменений.



Сохранение URL ADCM

Шаг 5. Обновление ADCM

> **ВНИМАНИЕ**
>
> - Перед выполнением обновления рекомендуется сделать резервную копию данных ADCM. Дополнительную информацию можно получить в статье Резервное копирование и восстановление.
> - Понижение версии ADCM не поддерживается.
> - Не рекомендуется обновлять ADCM более, чем на 5 минорных версий (см. `<minor>` в описании формата версионирования ниже).
> - Перед обновлением ADCM до версии 2.7.0 выполните миграцию данных из SQLite в PostgreSQL. Миграцию можно осуществить в ADCM любой версии (до 2.6.0 включительно).

Чтобы выполнить offline-обновление ADCM до нужной версии, сделайте следующее:

1. Остановите Docker-контейнер `adcm`:

```
$ sudo docker stop adcm
```

2. Удалите Docker-контейнер `adcm`:

```
$ sudo docker container rm adcm
```

> **ПРИМЕЧАНИЕ**
>
> Клиенты, использующие Enterprise-версии продуктов, могут обратиться в команду поддержки Arenadata для получения архива с расширением *sh.xz*, который содержит специальный скрипт для ADCM. После остановки Docker-контейнера `adcm` дальнейшее обновление ADCM включает установку согласно инструкции.

3. Загрузите Docker-образ из Arenadata Docker Registry на хост, который имеет доступ в Интернет.

```
$ sudo docker pull hub.arenadata.io/adcm/adcm:<version>
```

где `<version>` — это версия Docker-образа в одном из следующих форматов:

- `<major>.<minor>.<patch>` — если необходим конкретный патч ADCM. Пример: `2.0.0`.
- `<major>.<minor>` — если необходим последний патч в рамках выбранной версии ADCM. Пример: `2.0`.

> **ВАЖНО**
>
> - `latest` использовался для ранних версий ADCM и начиная с версии `2.0.0` больше не поддерживается.
> - До выпуска ADCM 2.0.0 для версионирования использовался следующий формат: `YYYY.MM.DD.HH`.

4. Сохраните загруженный Docker-образ в файл с расширением *.tar*.

```
$ sudo docker save -o <file_name> hub.arenadata.io/adcm/adcm:<version>
```

где `<file_name>` — имя создаваемого файла с расширением (например, *adcm_2.5.0_arenadata.tar*).

> **ПРИМЕЧАНИЕ**
>
> Выполнение команды может занять некоторое время. Пожалуйста, дождитесь ее завершения.

5. Скопируйте сохраненный файл с текущего хоста на хост без доступа к Интернету любым из доступных способов.

> **ВАЖНО**
>
> Убедитесь в наличии настроенного доступа к хостам и необходимых прав доступа к файлу.

6. Загрузите Docker-образ из сохраненного файла.

```
$ sudo docker load -i <file_path>
```

где `<file_path>` — абсолютный путь к сохраненному файлу с Docker-образом.

Пример результата успешного выполнения команды:

```
Loaded image: hub.arenadata.io/adcm/adcm:2.5.0
```

7. Создайте новый Docker-контейнер на базе загруженного образа:

```
$ sudo docker create --name adcm -p 8000:8000 -v /opt/adcm:/adcm/data -e DB_HOST="<DATABASE_HOSTNAME_OR_IP_ADDRESS>" -e DB_PORT="
<DATABASE_TCP_PORT>" -e DB_USER="<DATABASE_USERNAME>" -e DB_NAME="<DATABASE_NAME>" -e DB_PASS="<DATABASE_USER_PASSWORD>" -e
DB_OPTIONS="<DATABASE_OPTIONS>" hub.arenadata.io/adcm/adcm:<version>
```

> **ПРИМЕЧАНИЕ**
>
> Подробная информация о переменных окружения `DB_HOST`, `DB_PORT`, `DB_USER`, `DB_NAME`, `DB_PASS` и `DB_OPTIONS` приведена в статье Установка.

8. Запустите новый Docker-контейнер `adcm`:

```
$ sudo docker start adcm
```

Подготовка хостов

Содержание

Назад к содержанию

Права пользователей для установки ADH

Перед установкой нового кластера необходимо добавить информацию о его будущих хостах в ADCM. Это задача **хостпровайдеров** — специальных плагинов, отвечающих за взаимодействие между хостами и ADCM.

ADCM поддерживает несколько типов хостпровайдеров. Для получения подробной информации о том, как их настроить и использовать для создания хостов, обратитесь к документации:

- SSH
- Yandex Cloud
- VCloud (VMware)
- Cloud.ru Advanced
- CROC Cloud

Для offline-установки рекомендуется использование SSH-хостпровайдера.

ВНИМАНИЕ

При создании хостов ADH в ADCM требуется заполнять их полные доменные имена (FQDN). Максимальная длина FQDN — 38 символов.

Права пользователей для установки ADH

Если в настройках хостпровайдера указан кастомный UNIX-пользователь, от имени которого будут выполняться команды установки компонентов ADH, убедитесь, что данный пользователь может выполнять команды с правами суперпользователя.

Чтобы пользователь обладал необходимыми правами для установки ADH, выполните следующее:

1. Предоставьте пользователю привилегии `root`:

CentOS 7/RHEL 7/РЕД ОС 7.3/Alt Linux SP 8

```
$ sudo usermod -aG wheel <user>
```

где `<user>` — имя пользователя.

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
$ sudo usermod -aG sudo <user>
```

где `<user>` — имя пользователя.

2. Убедитесь, что в файле */etc/sudoers* есть запись, указанная ниже в зависимости от используемой операционной системы, с помощью следующей команды:

```
$ sudo vi /etc/sudoers
```

Чтобы вернуться в терминал, нажмите `Esc`, введите `:q!` и нажмите `Enter`.

CentOS 7/RHEL 7/РЕД ОС 7.3

```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
```

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Alt Linux SP 8

```
## Uncomment to allow members of group wheel to execute any command
WHEEL_USERS ALL=(ALL:ALL) ALL
```

Создание кластера

Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Получение Enterprise-версии бандла

Шаг 2. Загрузка бандла в ADCM

Шаг 3. Создание кластера на базе загруженного бандла

В ADCM **кластер** означает набор сервисов, запущенных на группе хостов.

Дистрибутивы кластеров для ADCM поставляются в виде бандлов. В общем случае бандл представляет обычный архив, описывающий логику развертывания кластера.

Шаги для добавления нового кластера приведены ниже.

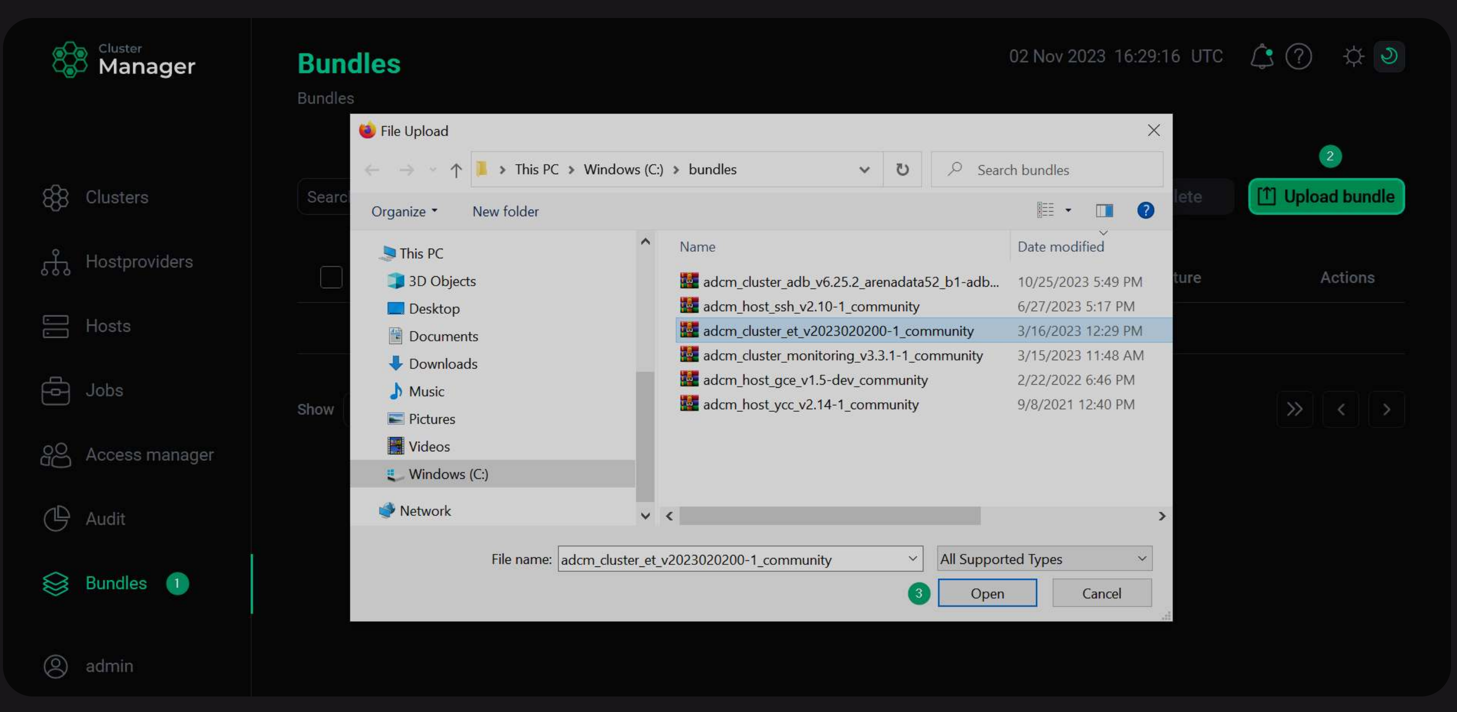
Шаг 1. Получение Enterprise-версии бандла

Пожалуйста, запросите бандл у команды поддержки Arenadata.

Шаг 2. Загрузка бандла в ADCM

Чтобы загрузить бандл в ADCM:

1. Выберите пункт ***Bundles*** в левом навигационном меню и нажмите ***Upload bundle***.
2. Выберите бандл в открывшемся диалоговом окне.

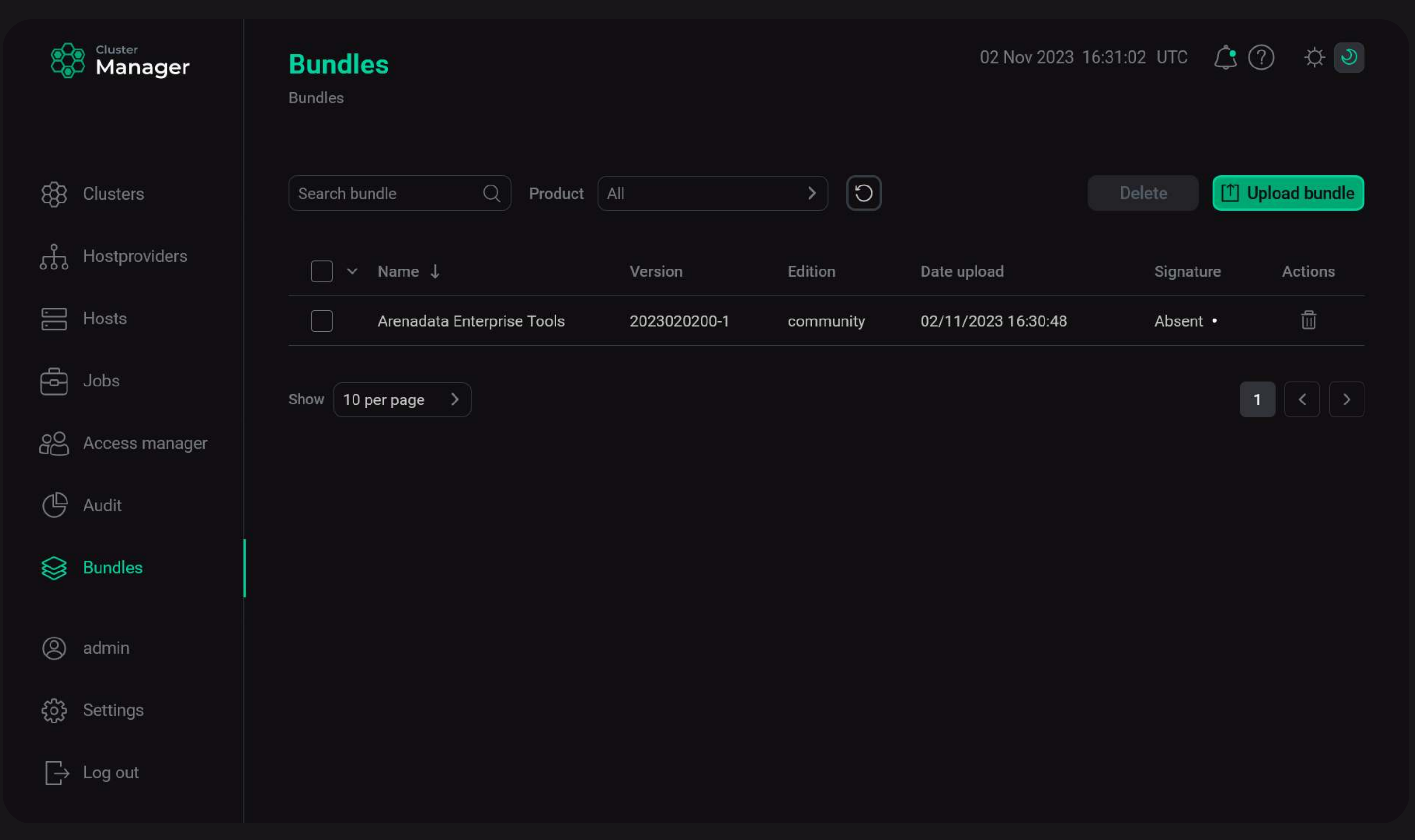

Cluster
Manager
Bundles
Bundles
02 Nov 2023 16:29:16 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
1
admin
2
Upload bundle
Actions
Show
File Upload
This PC > Windows (C:) > bundles
Search bundles
Organize
New folder
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
Network
All Supported Types
3
Open
Cancel

Name	Date modified
adcm_cluster_adb_v6.25.2_arenadata52_b1-adb...	10/25/2023 5:49 PM
adcm_host_ssh_v2.10-1_community	6/27/2023 5:17 PM
adcm_cluster_et_v2023020200-1_community	3/16/2023 12:29 PM
adcm_cluster_monitoring_v3.3.1-1_community	3/15/2023 11:48 AM
adcm_host_gce_v1.5-dev_community	2/22/2022 6:46 PM
adcm_host_ycc_v2.14-1_community	9/8/2021 12:40 PM

File name: adcm_cluster_et_v2023020200-1_community

Переход к загрузке бандла

3. После успешной загрузки бандл отображается на странице ***Bundles***.


Cluster
Manager
Bundles
Bundles
02 Nov 2023 16:31:02 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search bundle
Product
All
Delete
Upload bundle
Show
10 per page
1

Name	Version	Edition	Date upload	Signature	Actions
Arenadata Enterprise Tools	2023020200-1	community	02/11/2023 16:30:48	Absent •	

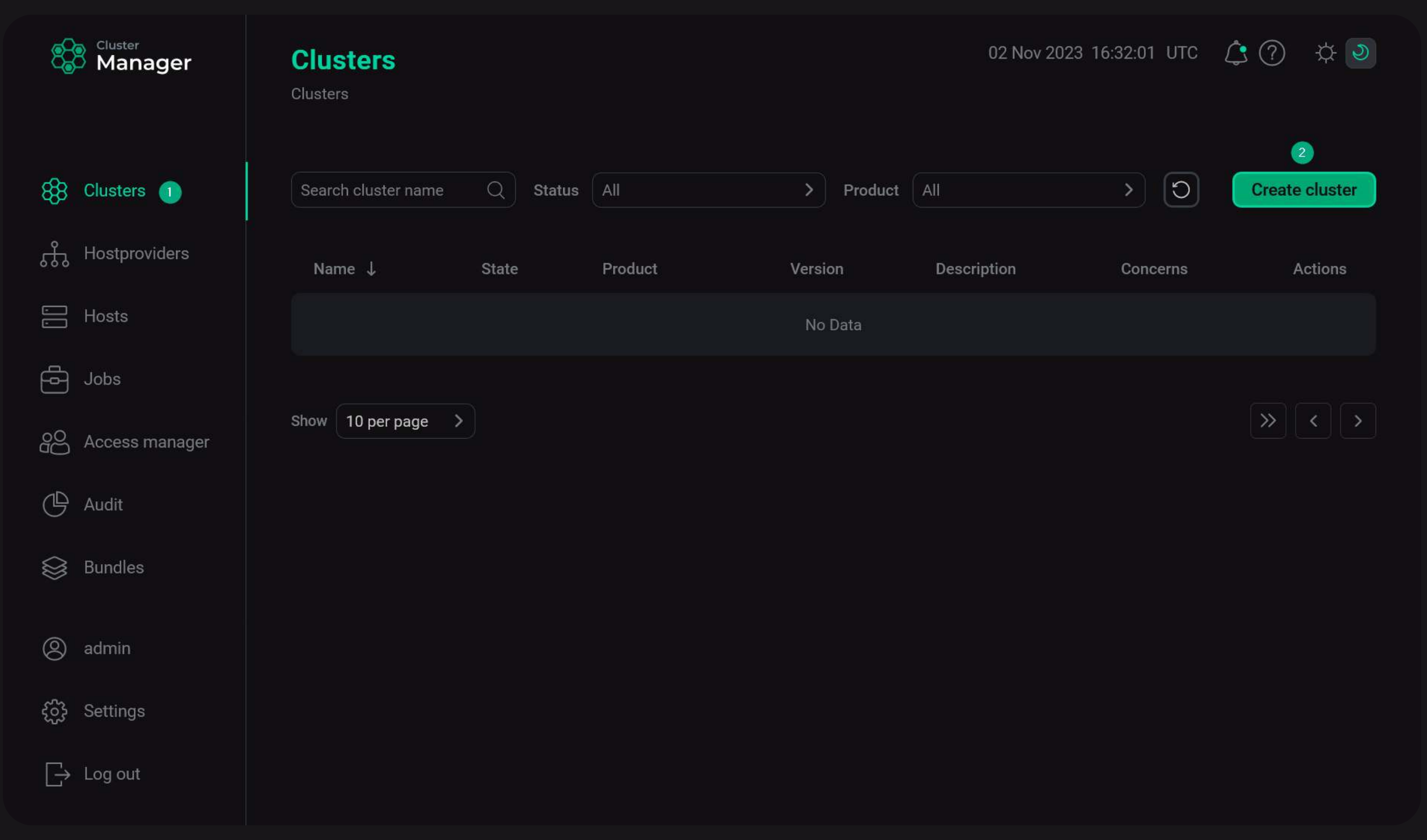
Результат успешной загрузки бандла

Шаг 3. Создание кластера на базе загруженного бандла

> **ВАЖНО**
>
> На этапе создания кластера через web-интерфейс ADCM генерируется только сам кластер. Это означает добавление записи о кластере в ADCM — сервисы на этом шаге еще не устанавливаются.

Для создания кластера необходимо выполнить шаги:

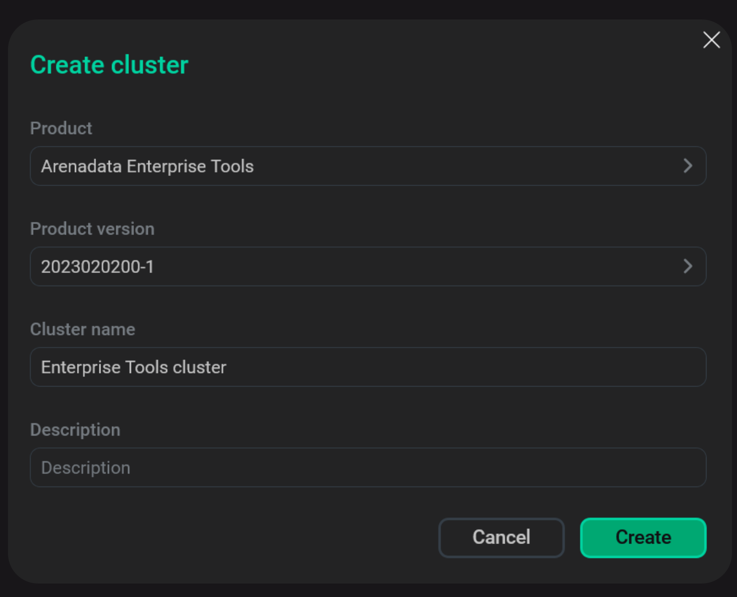
1. Выберите пункт ***Clusters*** в левом навигационном меню и нажмите ***Create cluster***.


Cluster
Manager
Clusters
Clusters
02 Nov 2023 16:32:01 UTC
Clusters
1
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
2
Create cluster
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
No Data						

Переход к созданию кластера

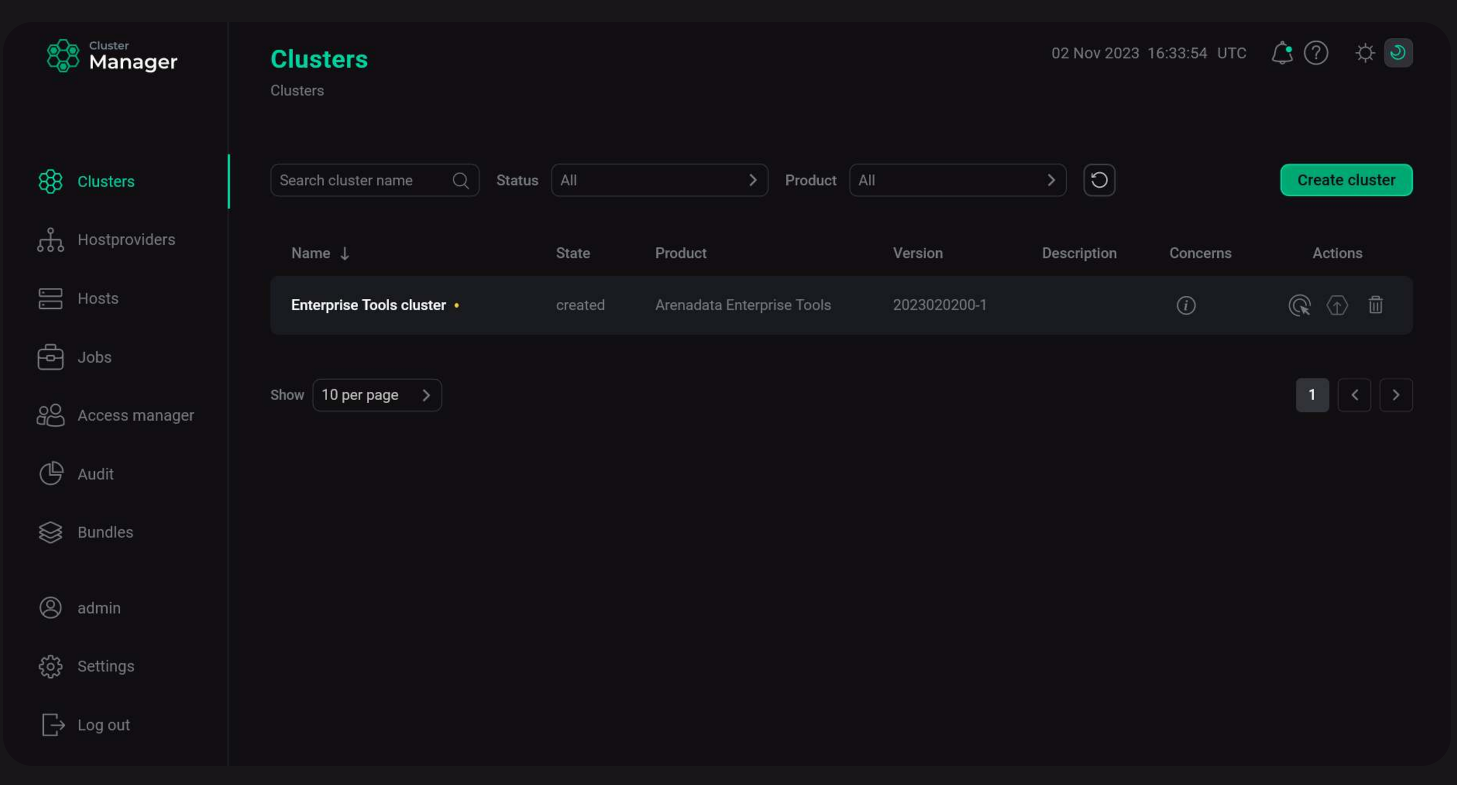
2. В открывшемся окне требуется описать новый кластер:
   - Выберите загруженный бандл в поле ***Product***.
   - Выберите версию бандла в поле ***Product version***. Несколько версий становятся доступны после загрузки нескольких бандлов для одного типа продуктов.
   - Введите имя кластера в поле ***Cluster name***.
   - При необходимости добавьте описание кластера в поле ***Description***.
   - Установите флаг ***I accept Terms of Agreement*** для подписания лицензионного соглашения. Флаг отображается только для бандлов версии Enterprise. Чтобы ознакомиться с текстом соглашения, перейдите по ссылке ***Terms of Agreement***. Подписать соглашение также возможно на странице ***Bundles***, нажав на соответствующий продуктовый бандл.
   - Нажмите ***Create***.


Create cluster
Cancel
Create

Product: Arenadata Enterprise Tools
Product version: 2023020200-1
Cluster name: Enterprise Tools cluster
Description: Description

Создание кластера

3. После успешного добавления кластер отображается на странице ***Clusters***.


Cluster
Manager
Clusters
Clusters
02 Nov 2023 16:33:54 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Enterprise Tools cluster •	created	Arenadata Enterprise Tools	2023020200-1			

Результат успешного добавления кластера

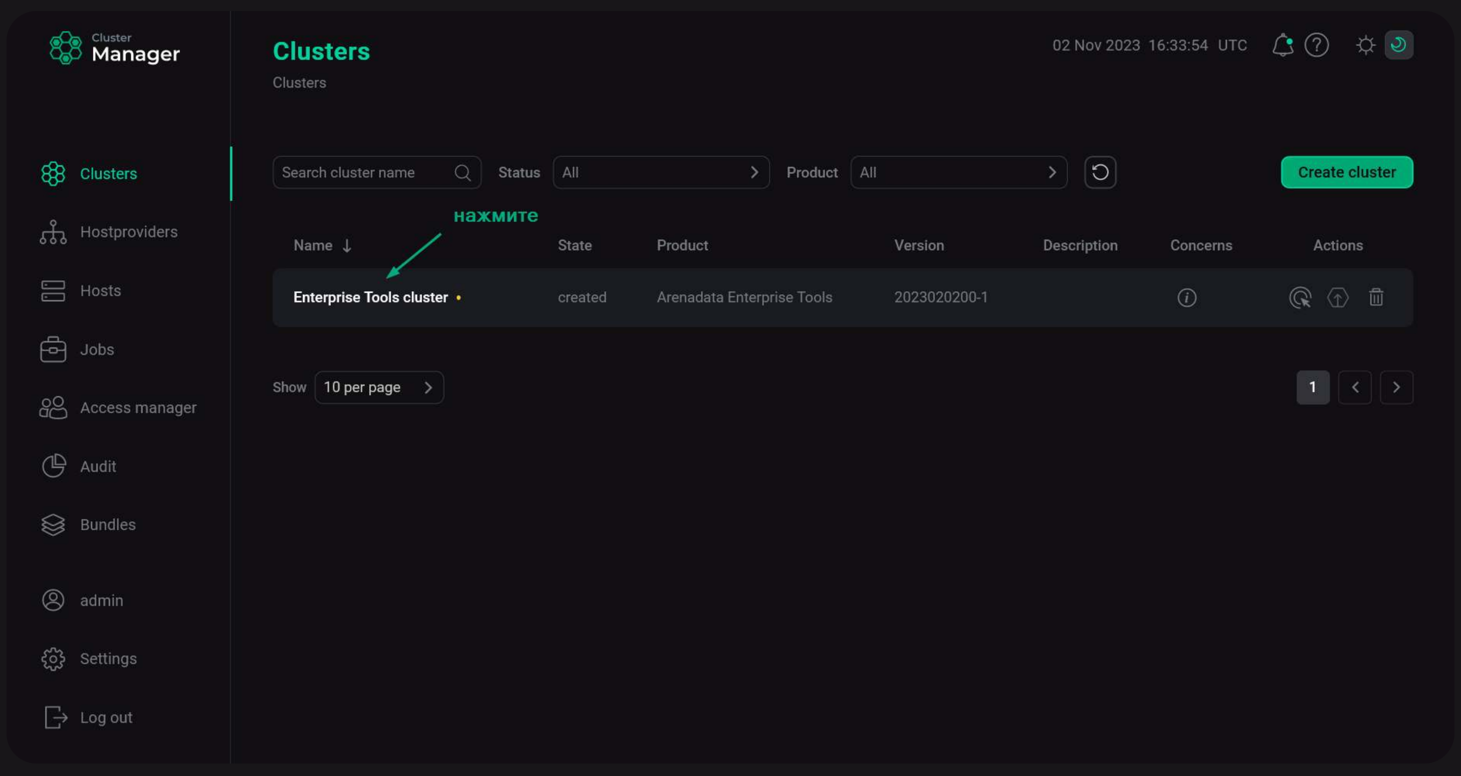
Добавление сервисов

Дарья Барышева

Назад к содержанию

В ADCM **сервис** означает программное обеспечение, выполняющее некоторую функцию. Для добавления сервисов в кластер:

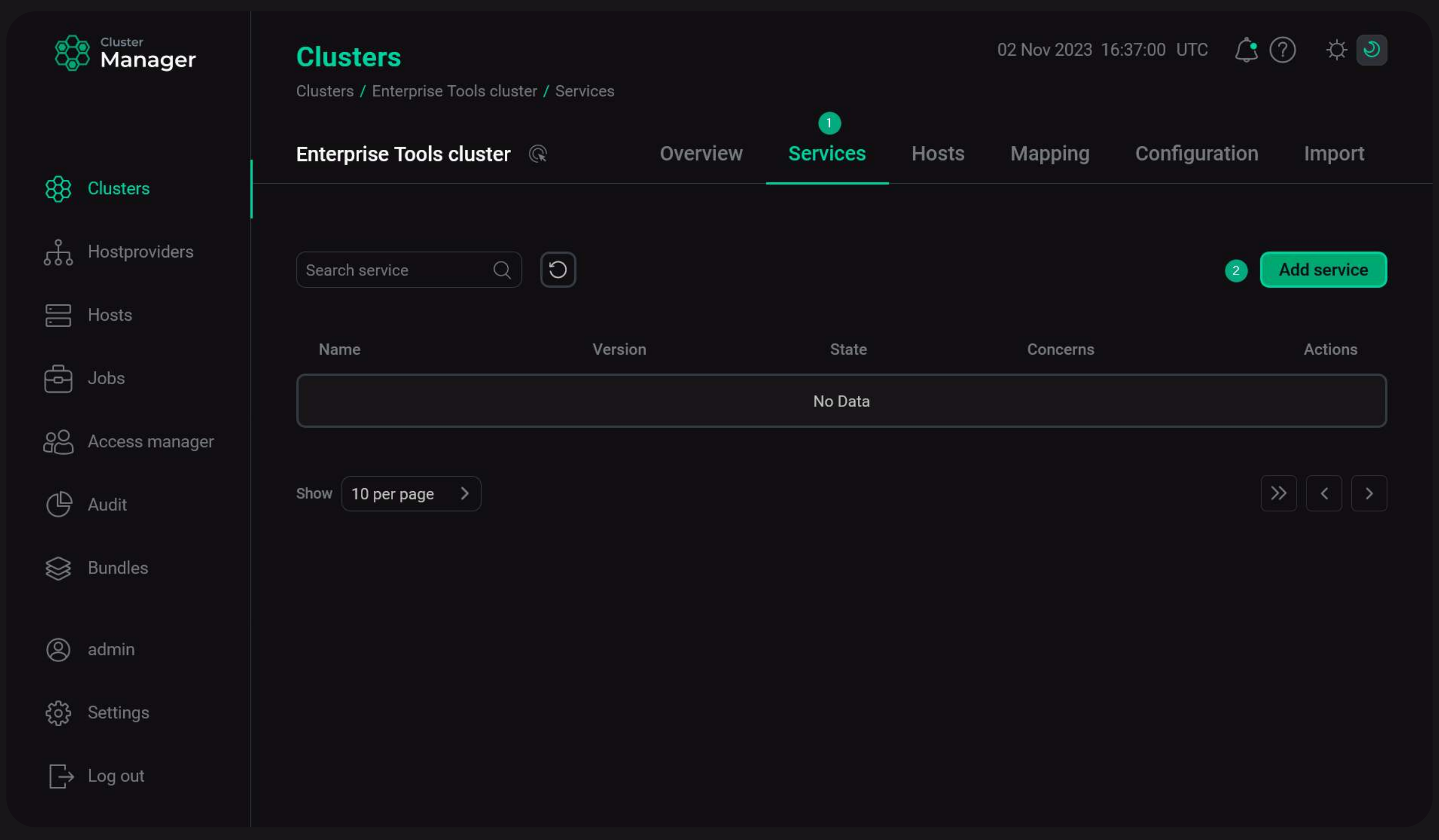
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
02 Nov 2023 16:33:54 UTC
Search cluster name
Status
All
Product
All
Create cluster
нажмите
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Enterprise Tools cluster	created	Arenadata Enterprise Tools	2023020200-1			

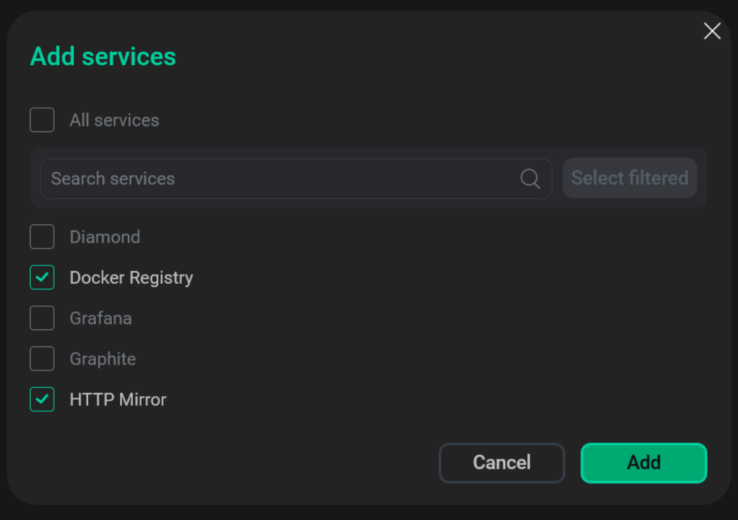
Выбор кластера

2. Откройте вкладку ***Services*** на странице кластера и нажмите ***Add service***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Enterprise Tools cluster / Services
02 Nov 2023 16:37:00 UTC
Enterprise Tools cluster
Overview
1
Services
Hosts
Mapping
Configuration
Import
Search service
2
Add service
Name
Version
State
Concerns
Actions
No Data
Show
10 per page

Переход к добавлению сервисов

3. В открывшемся окне выберите сервисы для добавления в кластер и нажмите ***Add***.


Add services
Search services
Select filtered
Cancel
Add

☐ All services
☐ Diamond
☑ Docker Registry
☐ Grafana
☐ Graphite
☑ HTTP Mirror

Выбор сервисов

Краткое описание доступных сервисов приведено ниже.

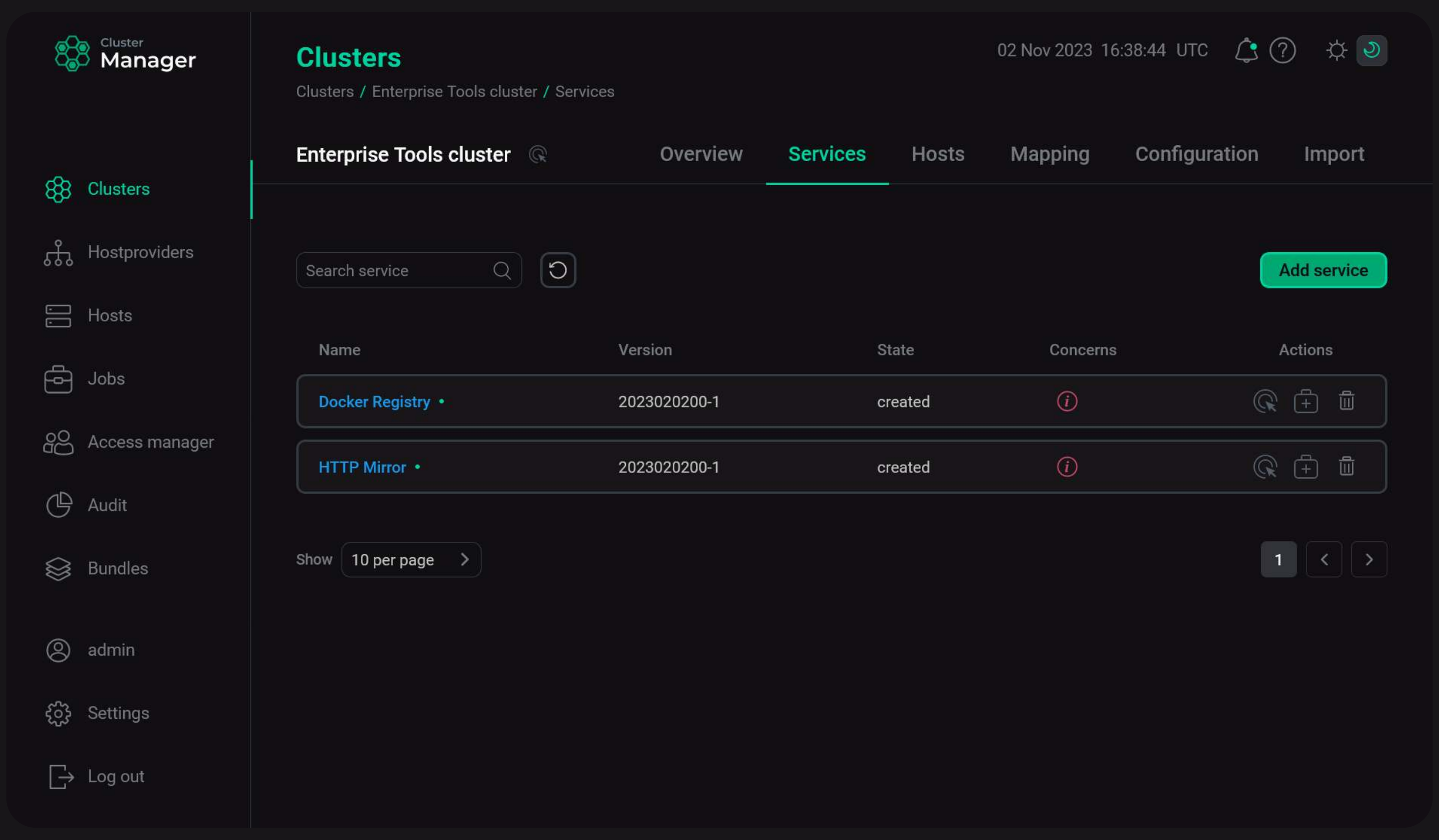
Сервисы, которые необходимо добавить в кластер Enterprise Tools

Сервис	Назначение
HTTP Mirror	Сервис, используемый для размещения пакетов репозиториев, необходимых для offline-установки продуктов
Docker Registry	Сервис, используемый для хранения образов Docker, необходимых для offline-установки продуктов

ПРИМЕЧАНИЕ

Сервис CoreDNS не используется (deprecated). Сервисы Diamond, Graphite и Grafana необходимы для настройки мониторинга — они будут описаны отдельно.

4. В результате успешно добавленные сервисы отображаются на вкладке ***Services***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Enterprise Tools cluster / Services
02 Nov 2023 16:38:44 UTC
Enterprise Tools cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add service
Show
10 per page
1

Name	Version	State	Concerns	Actions
Docker Registry	2023020200-1	created		
HTTP Mirror	2023020200-1	created		

Результат успешного добавления сервисов в кластер

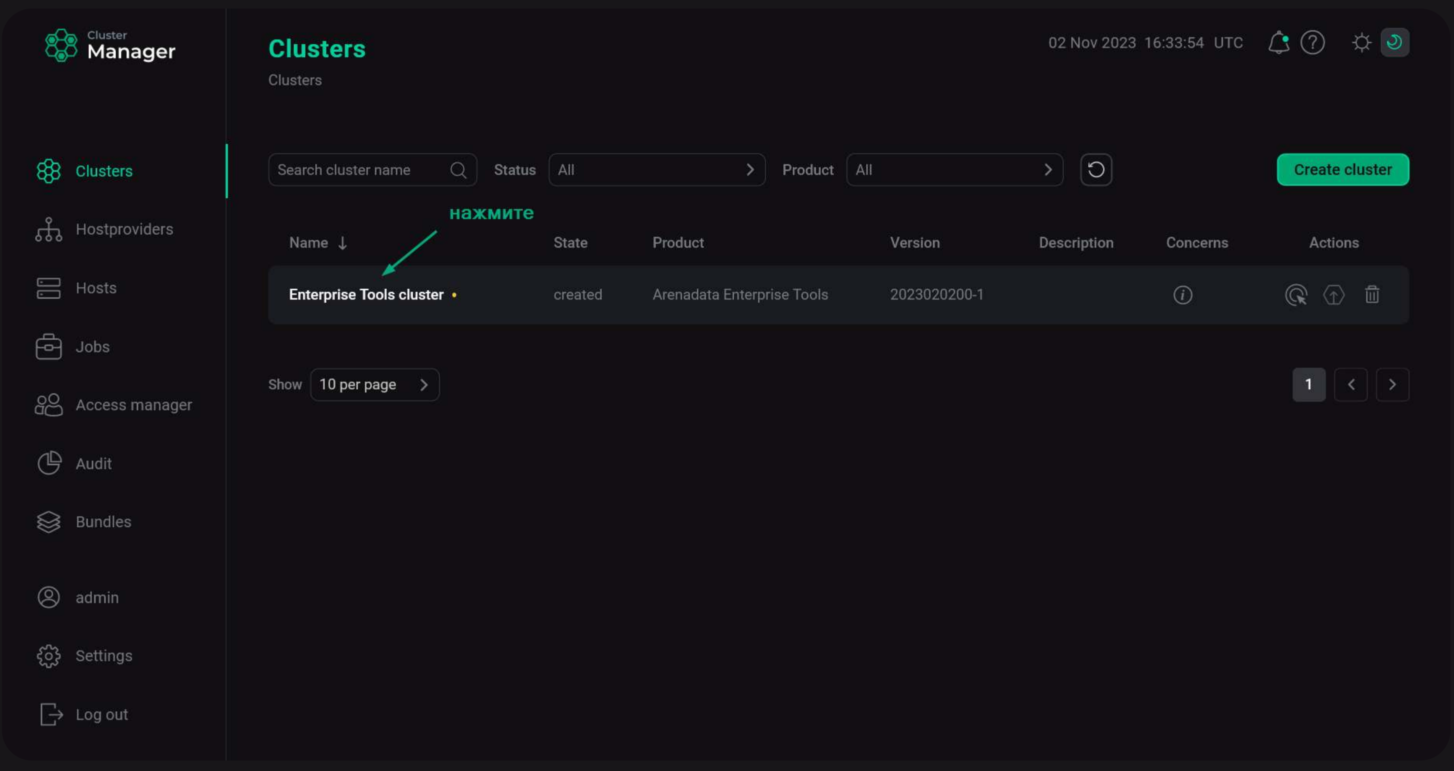
Добавление хостов в кластер

Дарья Барышева

Назад к содержанию

Для добавления хостов в кластер:

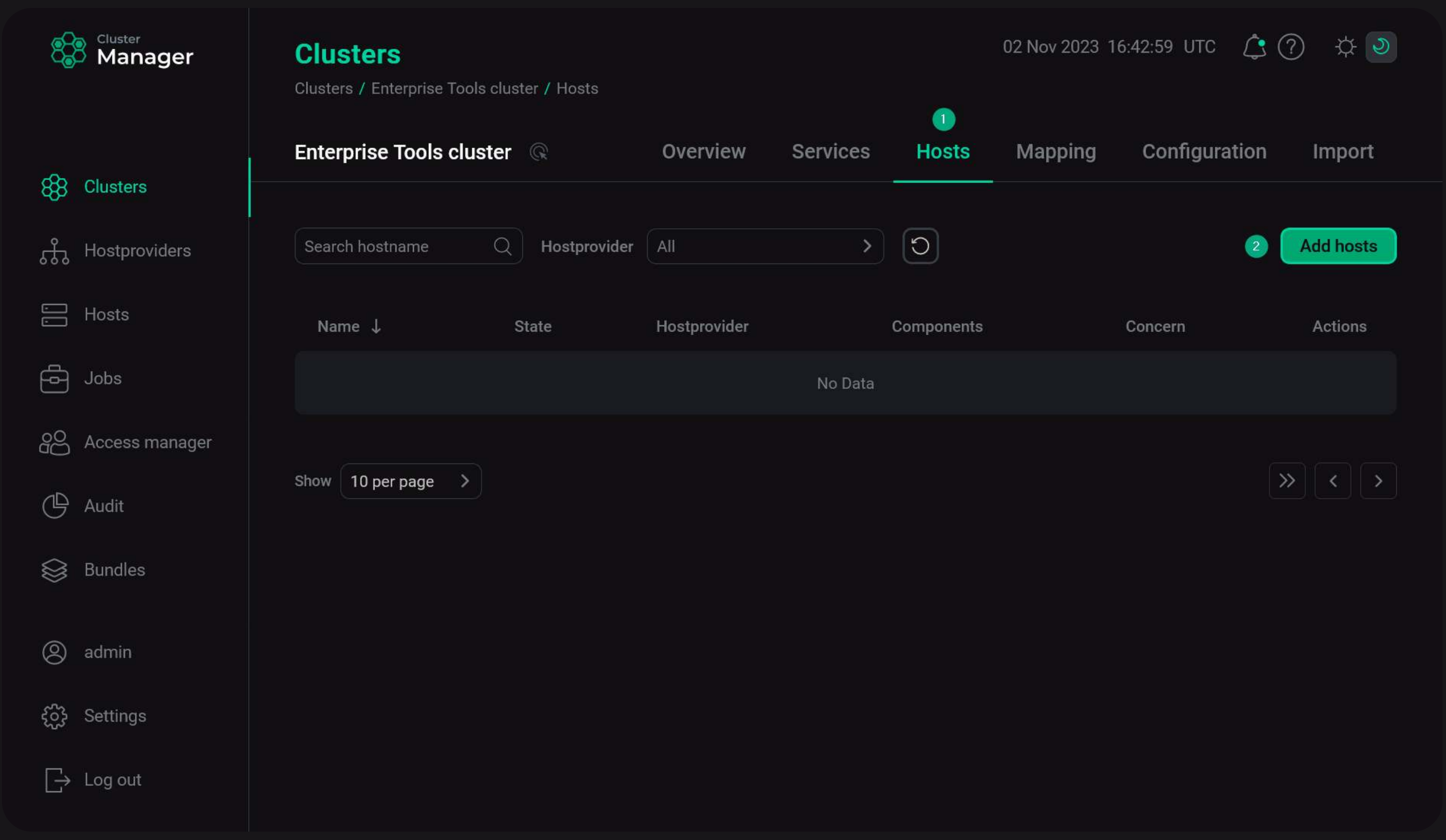
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
02 Nov 2023 16:33:54 UTC
Search cluster name
Status
All
Product
All
Create cluster
нажмите
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Enterprise Tools cluster	created	Arenadata Enterprise Tools	2023020200-1			

Выбор кластера

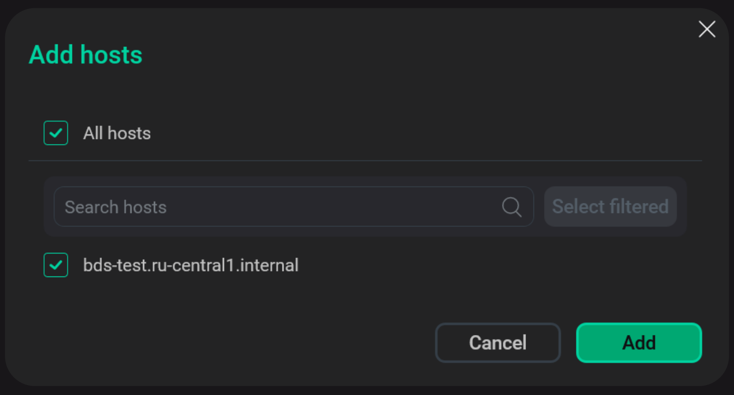
2. Откройте вкладку ***Hosts*** на странице кластера и нажмите ***Add hosts***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Enterprise Tools cluster / Hosts
02 Nov 2023 16:42:59 UTC
Enterprise Tools cluster
Overview
Services
1
Hosts
Mapping
Configuration
Import
Search hostname
Hostprovider
All
2
Add hosts
Show
10 per page

Name	State	Hostprovider	Components	Concern	Actions
No Data					

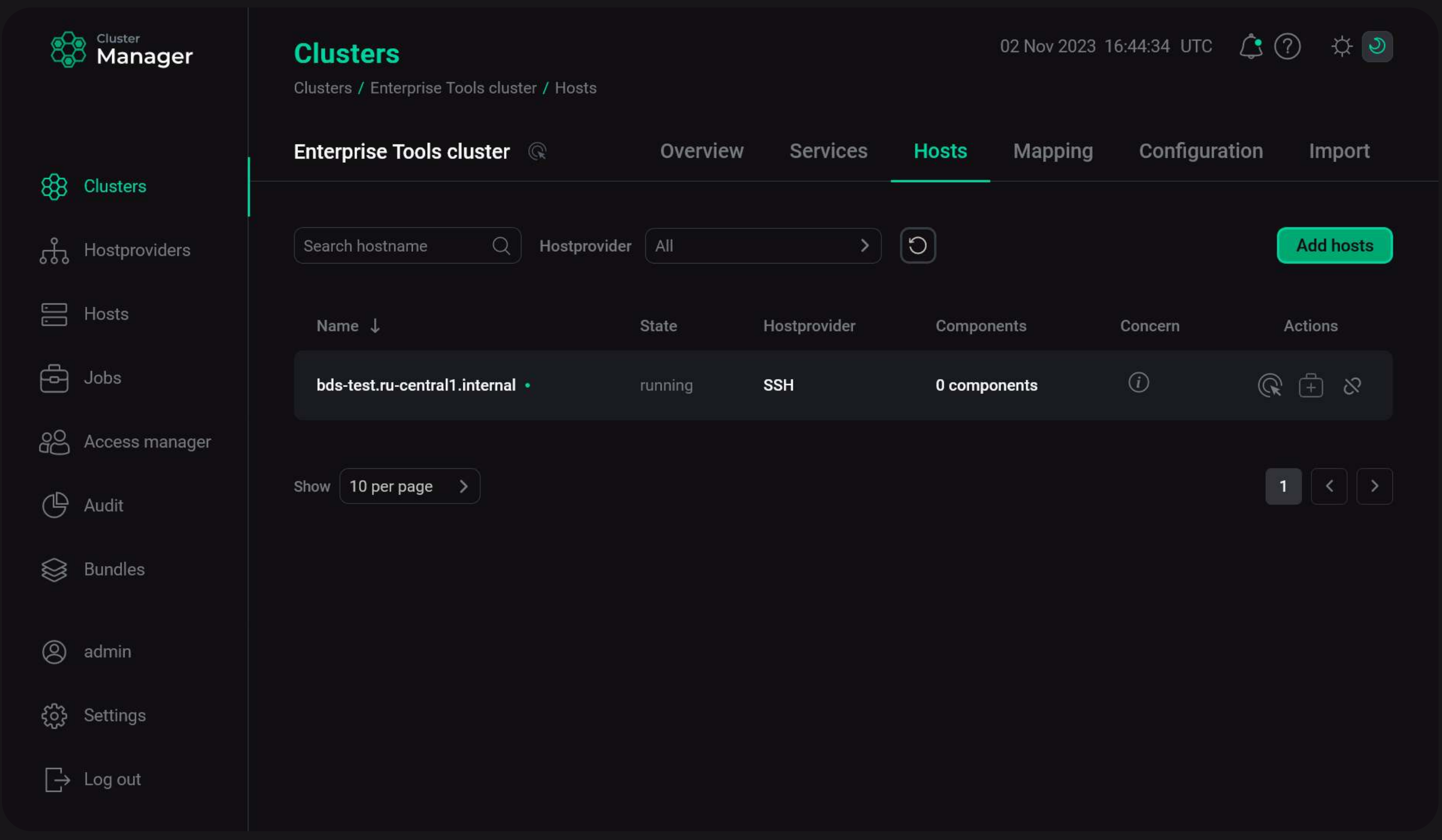
Переход к добавлению хостов

3. В открывшемся окне выберите хосты для добавления в кластер и нажмите ***Add***.


Add hosts
☑ All hosts
Search hosts
Select filtered
☑ bds-test.ru-central1.internal
Cancel
Add

Выбор хостов

4. В результате успешно добавленные хосты отображаются на вкладке ***Hosts***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Enterprise Tools cluster / Hosts
02 Nov 2023 16:44:34 UTC
Enterprise Tools cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Search hostname
Hostprovider
All
Add hosts
Show
10 per page
1

Name	State	Hostprovider	Components	Concern	Actions
bds-test.ru-central1.internal	running	SSH	0 components		

Результат успешного добавления хостов в кластер

Добавление компонентов

Дарья Барышева

Назад к содержанию

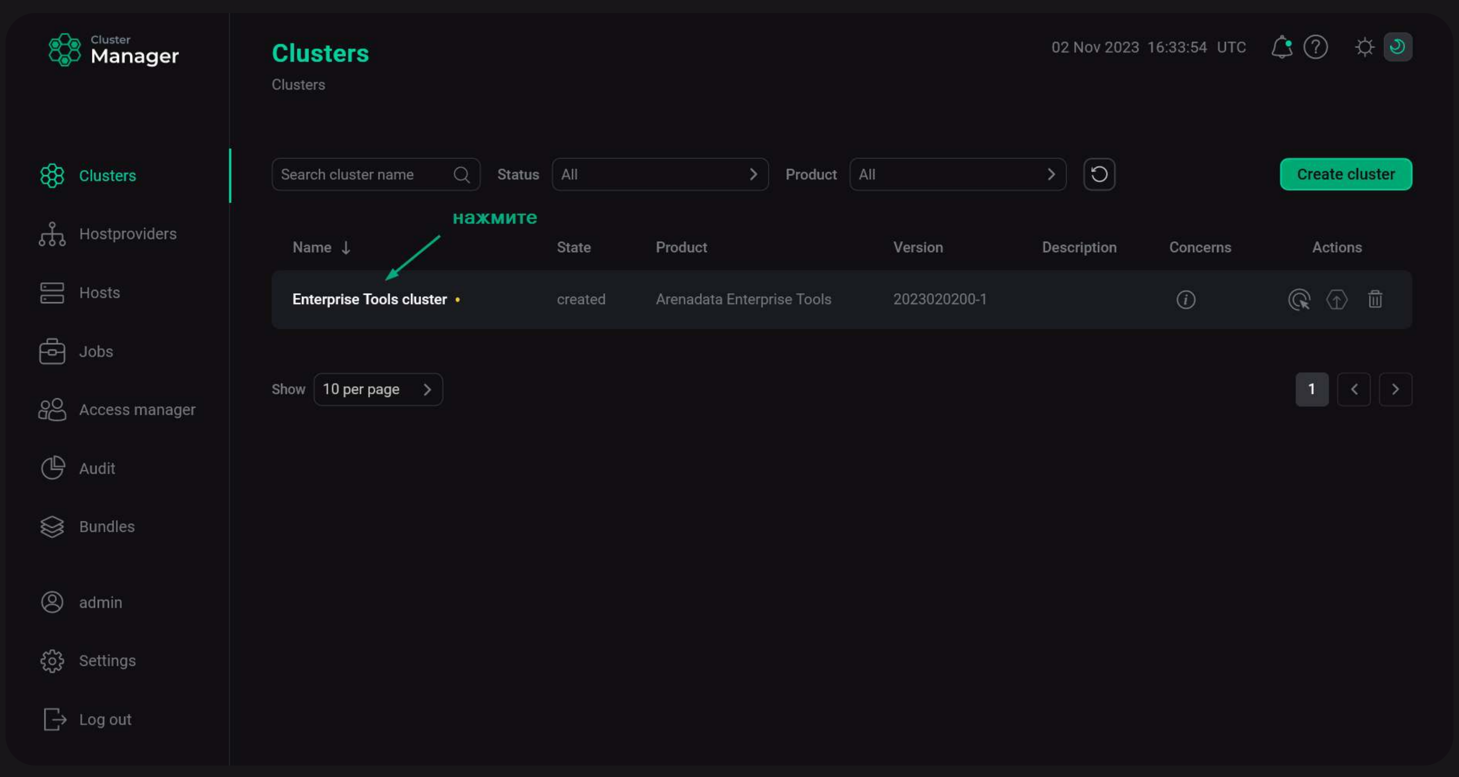
В ADCM **компонент** означает часть сервиса, которая должна быть развернута на одном или нескольких хостах кластера.

ВАЖНО

Изначально на хостах кластера нет ни одного компонента — распределение компонентов между хостами должно быть проведено вручную.

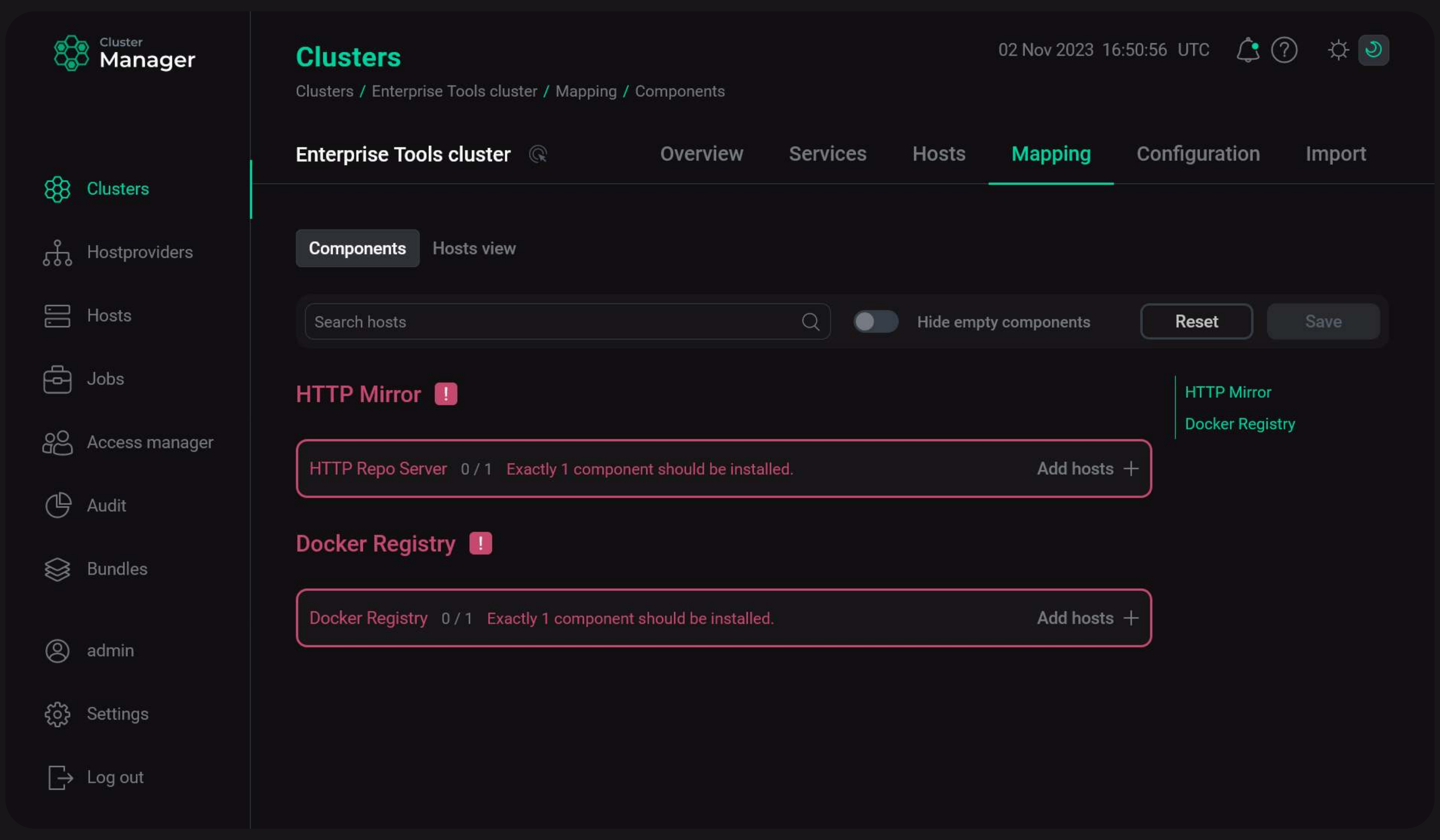
Для добавления компонентов на хосты кластера:

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
Product
Create cluster
нажмите
Name
State
Product
Version
Description
Concerns
Actions
Enterprise Tools cluster
created
Arenadata Enterprise Tools
2023020200-1
10 per page

Выбор кластера

2. Откройте вкладку ***Mapping*** на странице кластера.

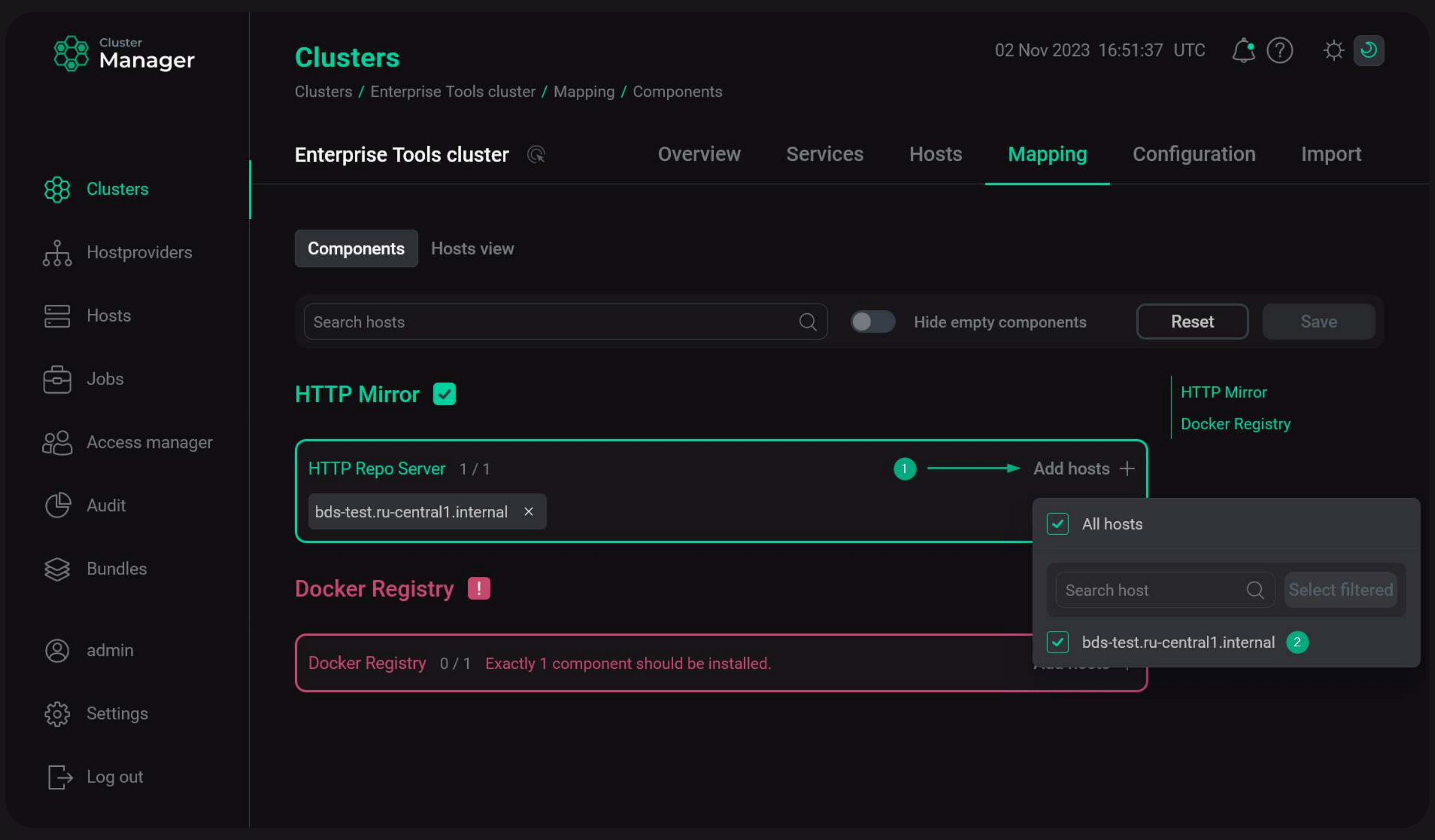

Clusters / Enterprise Tools cluster / Mapping / Components
Enterprise Tools cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Components
Hosts view
Search hosts
Hide empty components
Reset
Save
HTTP Mirror
Docker Registry
Add hosts

HTTP Repo Server 0 / 1 Exactly 1 component should be installed.

Docker Registry 0 / 1 Exactly 1 component should be installed.

Переход к распределению сервисных компонентов

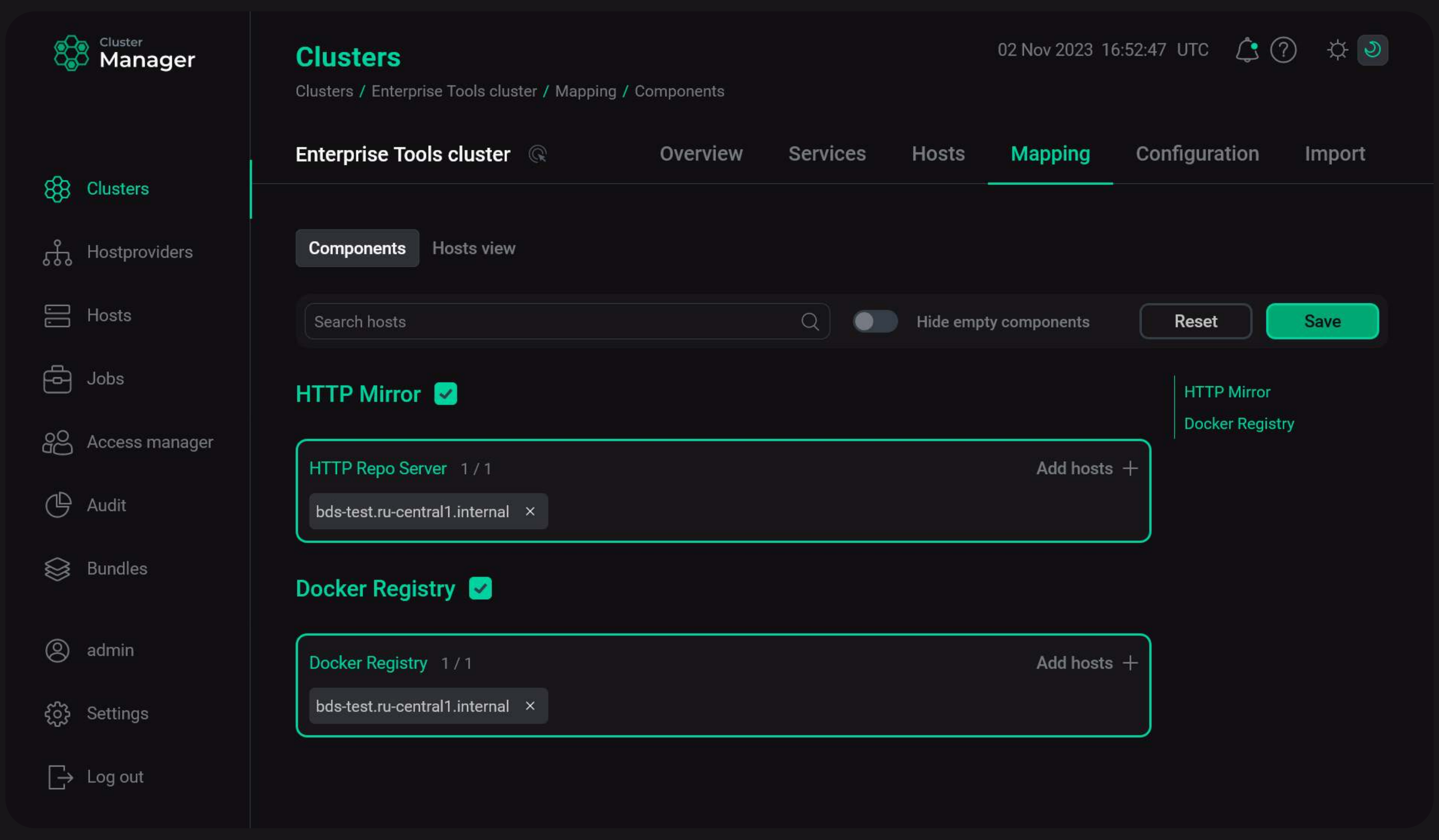
3. Распределите все компоненты между хостами кластера. Для назначения хостов компоненту нажмите ***Add hosts*** и выберите хосты в открывшемся окне.


HTTP Mirror
HTTP Repo Server 1 / 1
bds-test.ru-central1.internal
1
Add hosts
All hosts
Search host
Select filtered
bds-test.ru-central1.internal
2
Docker Registry

Docker Registry 0 / 1 Exactly 1 component should be installed.

Выбор хостов для компонента

4. После завершения распределения компонентов нажмите ***Save***.


Reset
Save
HTTP Mirror
HTTP Repo Server 1 / 1
bds-test.ru-central1.internal
Docker Registry
Docker Registry 1 / 1
bds-test.ru-central1.internal
Add hosts

Сохранение распределения компонентов

Установка кластера

Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Запуск установки

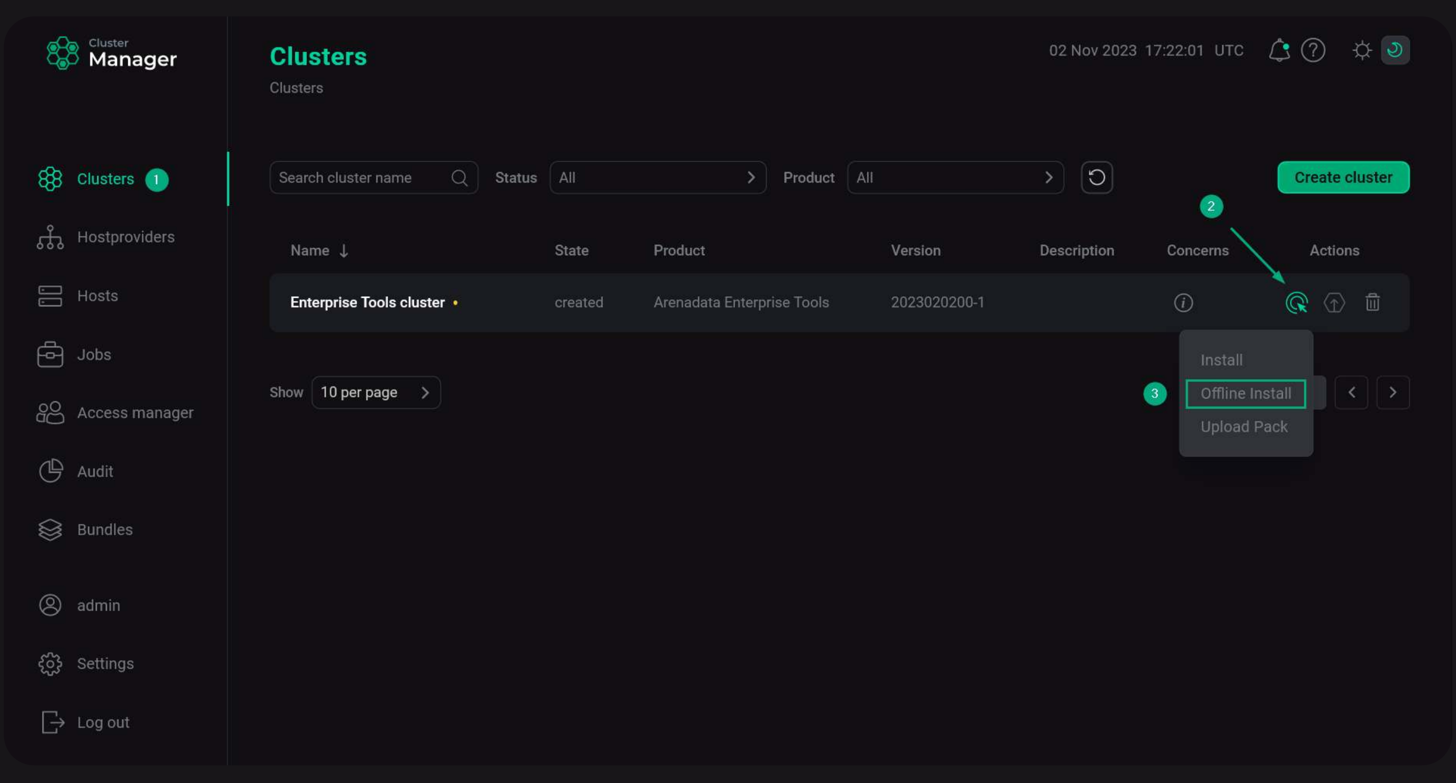
Шаг 2. Загрузка пакетов для offline-установки

Шаг 3. Проверка результатов

Шаг 1. Запуск установки

Для установки кластера Enterprise Tools выполните следующие шаги:

1. Запустите действие ***Offline Install*** для кластера, выбранного на странице ***Clusters***, нажав на иконку в столбце ***Actions***.

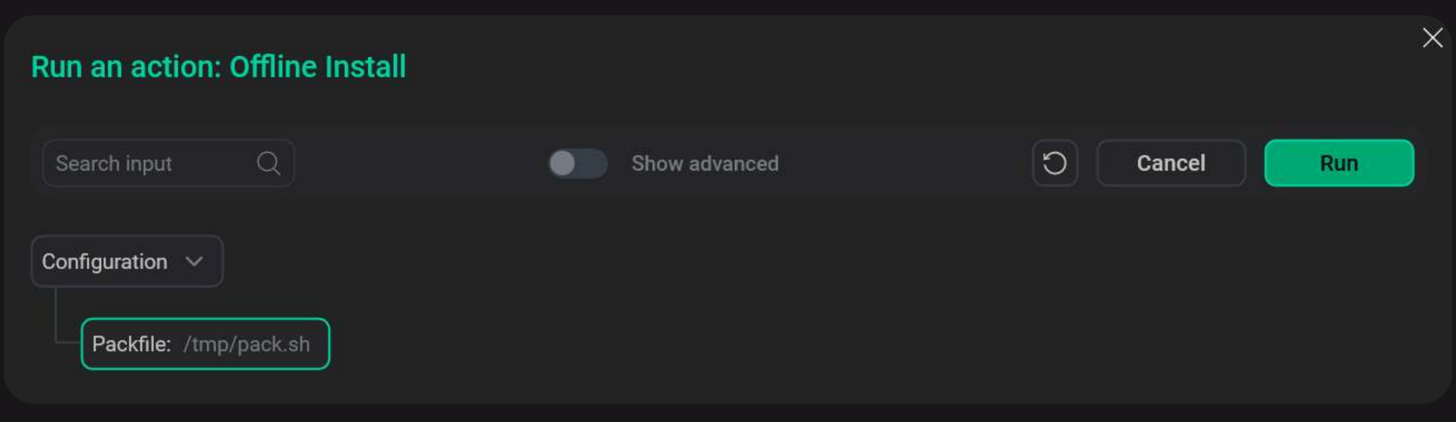


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Create cluster
Install
Offline Install
Upload Pack
1
2

Enterprise Tools cluster · created · Arenadata Enterprise Tools · 2023020200-1

Переход к установке кластера

2. В открывшемся диалоговом окне введите полный путь к shell-скрипту, полученному от команды поддержки Arenadata, и нажмите ***Run***.



Run an action: Offline Install
Search input
Show advanced
Cancel
Run
Configuration

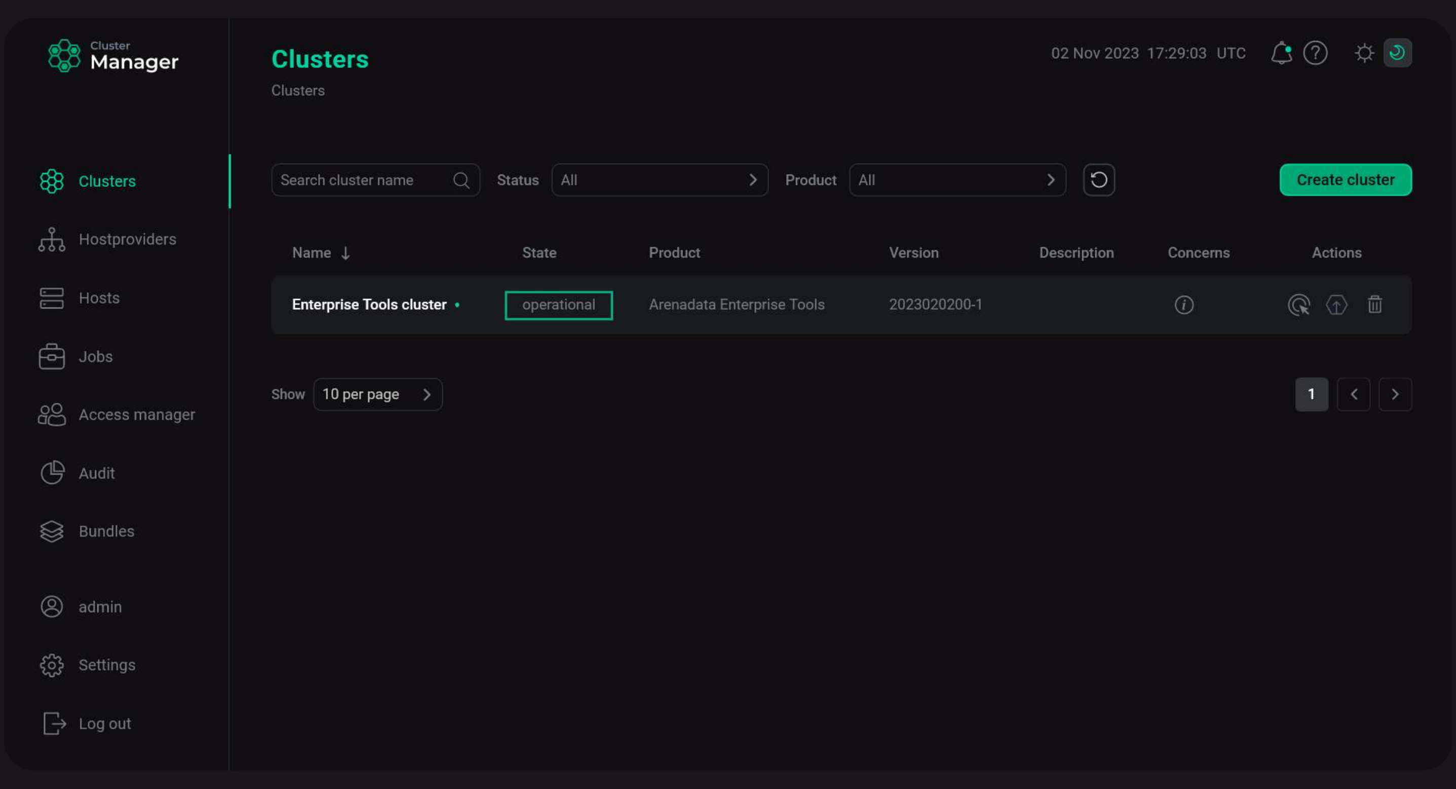
Packfile: /tmp/pack.sh

Подтверждение установки

3. Дождитесь завершения процесса установки.

ПРИМЕЧАНИЕ

В случае ошибок логи можно найти на странице ***Jobs***.



Cluster Manager
Clusters
Create cluster

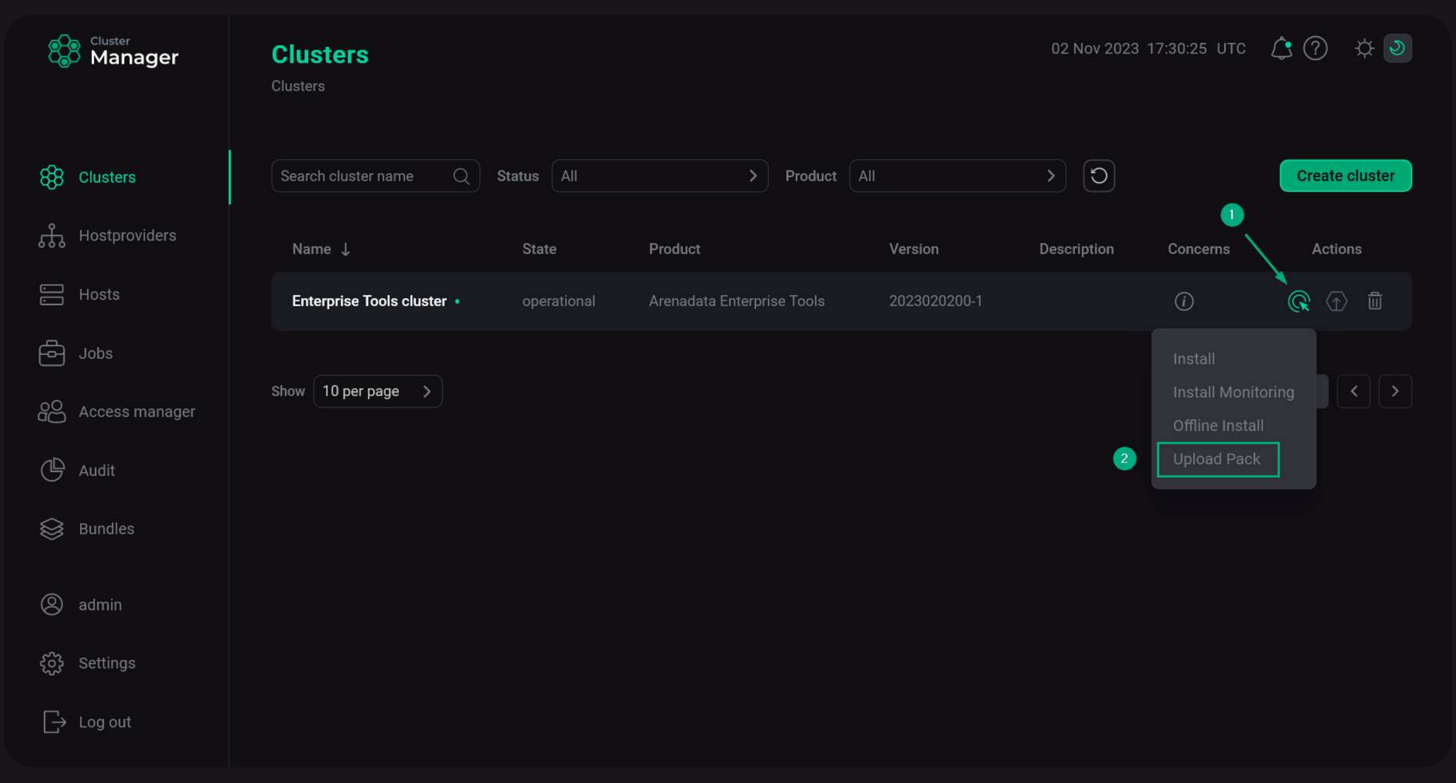
Enterprise Tools cluster · operational · Arenadata Enterprise Tools · 2023020200-1

Процесс установки завершен

Шаг 2. Загрузка пакетов для offline-установки

В дополнение к установке сервисов ET требуется загрузка всех установочных файлов и образов в инициализированный http-репозиторий и Docker Registry на хосте, где развернут кластер ET. Это важно для последующего offline-развертывания продуктов Arenadata. Для загрузки выполните шаги:

1. Запустите действие ***Upload Pack*** для кластера, выбранного на странице ***Clusters***, нажав на иконку в столбце ***Actions***.

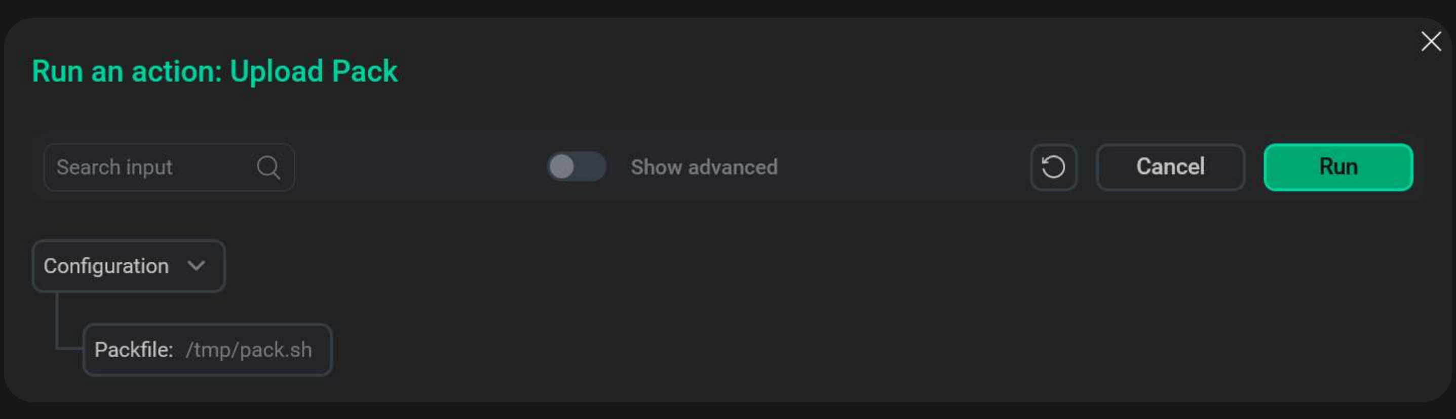


Cluster Manager
Clusters
Create cluster
Install
Install Monitoring
Offline Install
Upload Pack
1
2

Enterprise Tools cluster · operational · Arenadata Enterprise Tools · 2023020200-1

Переход к загрузке пакетов

2. В открывшемся диалоговом окне введите полный путь к shell-скрипту, полученному от команды поддержки Arenadata, и нажмите ***Run***.



Run an action: Upload Pack
Search input
Show advanced
Cancel
Run
Configuration

Packfile: /tmp/pack.sh

Подтверждение загрузки

3. Дождитесь завершения процесса загрузки.

ПРИМЕЧАНИЕ

В случае ошибок логи можно найти на странице ***Jobs***.

Шаг 3. Проверка результатов

Для проверки установки на хосте, где развернут кластер, выполните шаги:

1. Убедитесь, что порты `443` и `81` находятся в статусе `LISTEN`. Первый порт используется сервисом Docker Registry, второй порт — сервисом HTTP Mirror.

```
$ sudo netstat -ntlp|grep L|grep -E "81|443"
```

Результат команды:

```
tcp6       0      0 :::443                  :::*                    LISTEN      1694/docker-proxy-c
tcp6       0      0 :::81                   :::*                    LISTEN      1708/docker-proxy-c
```

2. Проверьте наличие директории */opt/arenadata/etools/httprepo/arenadata-repo*. Она должна содержать файлы, необходимые для offline-установки продуктов, выбранных вами на этапе получения shell-скрипта от команды поддержки Arenadata.

```
$ ls -la /opt/arenadata/etools/httprepo/arenadata-repo
```

Следующий пример показывает данные, которые могут храниться в директории в случае установки ADH.

```
total 0
drwxr-xr-x 5 root root 45 Sep 16 13:35 .
drwxr-xr-x 3 root root 28 Sep 16 13:35 ..
drwxr-xr-x 4 root root 38 Sep 16 13:35 ADH
drwxr-xr-x 4 root root 28 Sep 16 13:35 ADM
drwxr-xr-x 3 root root 20 Sep 16 13:35 zookeeper
```

Создание кластера

Евгения Кузина, Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Получение Enterprise-версии бандла

Шаг 2. Загрузка бандла в ADCM

Шаг 3. Создание кластера на базе загруженного бандла

В ADCM **кластер** означает набор сервисов, запущенных на группе хостов.

Дистрибутивы кластеров для ADCM поставляются в виде бандлов. В общем случае бандл представляет обычный архив, описывающий логику развертывания кластера.

Шаги для добавления нового кластера приведены ниже.

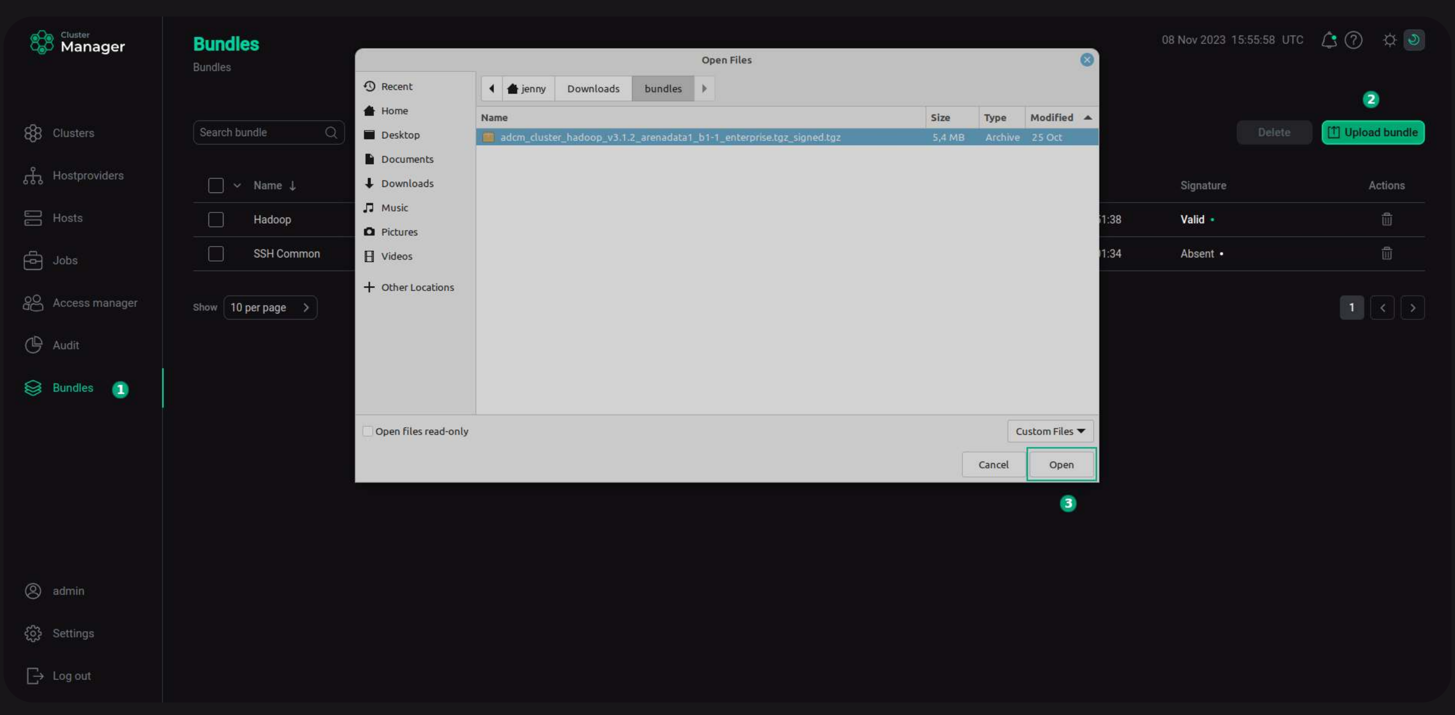
Шаг 1. Получение Enterprise-версии бандла

Пожалуйста, запросите бандл у команды поддержки Arenadata.

Шаг 2. Загрузка бандла в ADCM

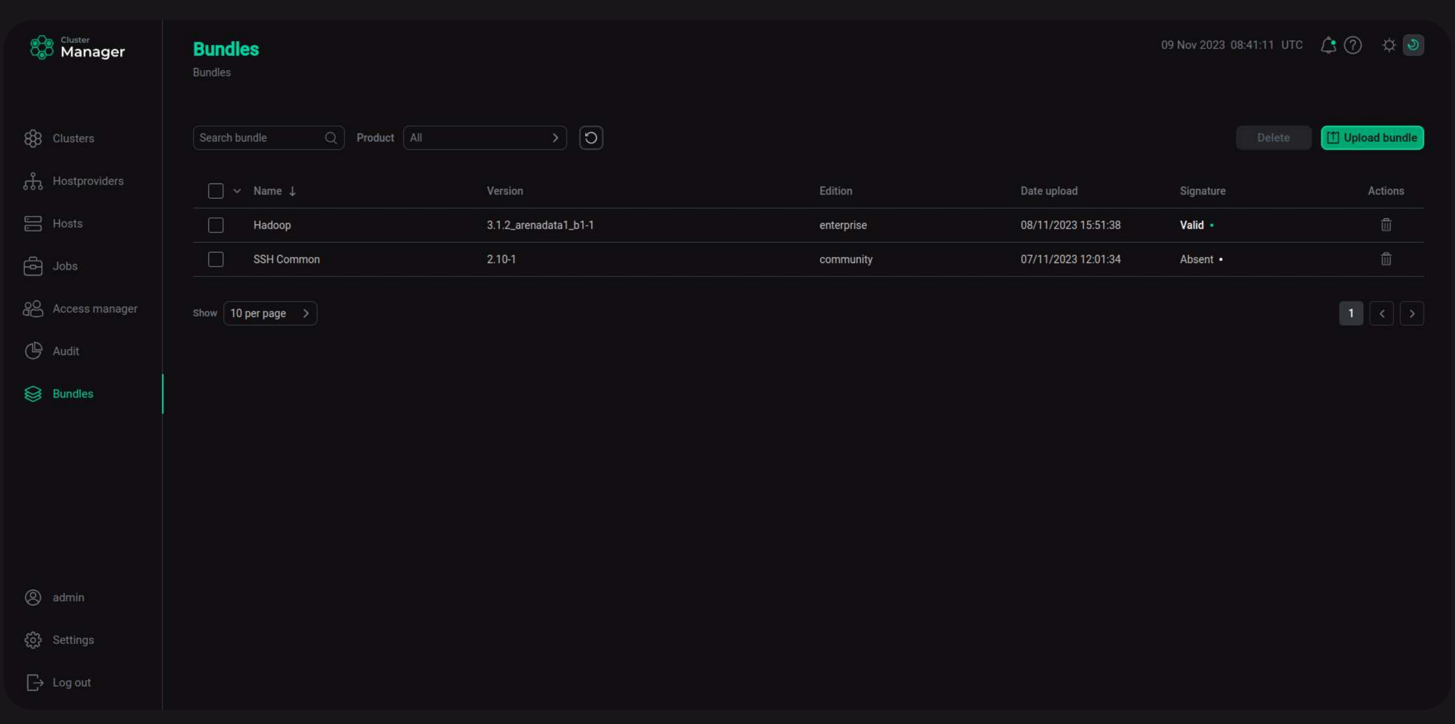
Чтобы загрузить бандл в ADCM:

1. Выберите пункт ***Bundles*** в левом навигационном меню и нажмите ***Upload bundle***.
2. Выберите бандл в открывшемся диалоговом окне.



Переход к загрузке бандла

3. После успешной загрузки бандл отображается на странице ***Bundles***.



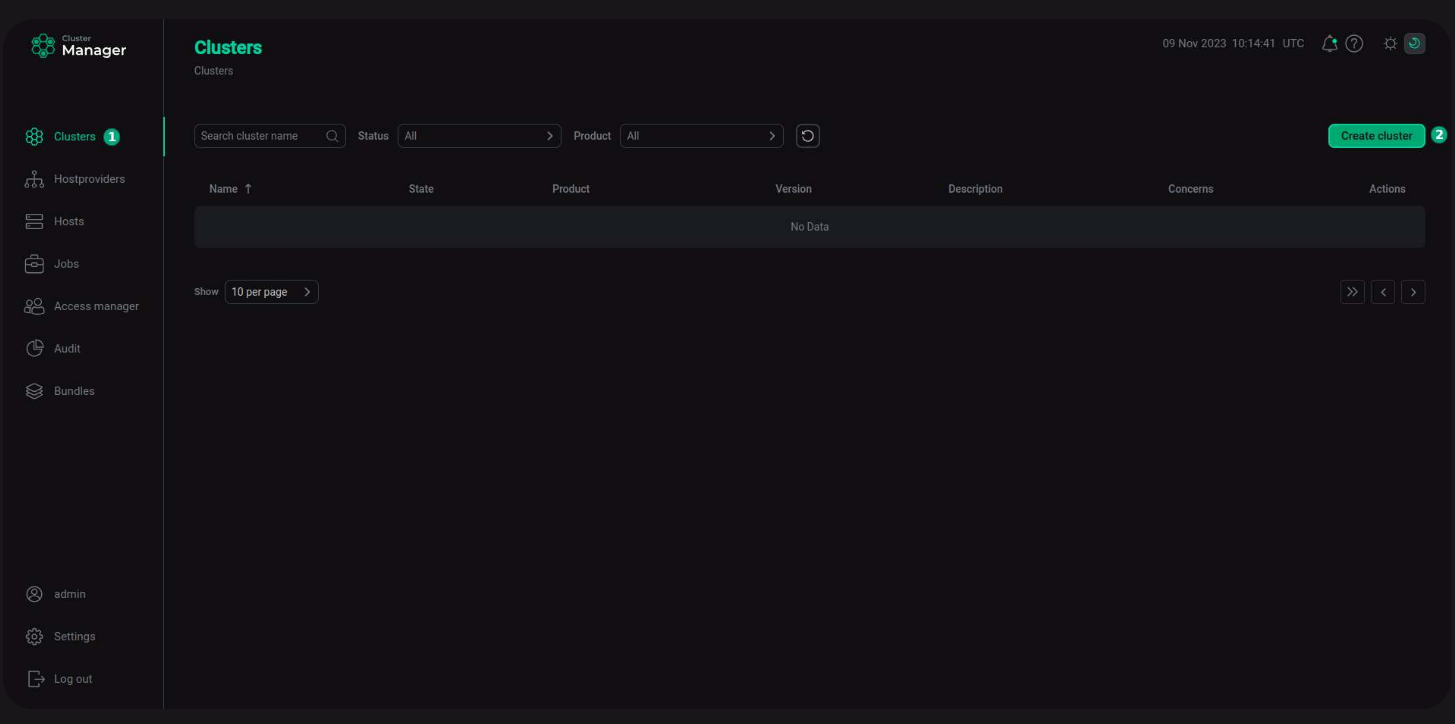
Результат успешной загрузки бандла

Шаг 3. Создание кластера на базе загруженного бандла

> **ВАЖНО**
>
> На этапе создания кластера через web-интерфейс ADCM генерируется только сам кластер. Это означает добавление записи о кластере в ADCM — сервисы на этом шаге еще не устанавливаются.

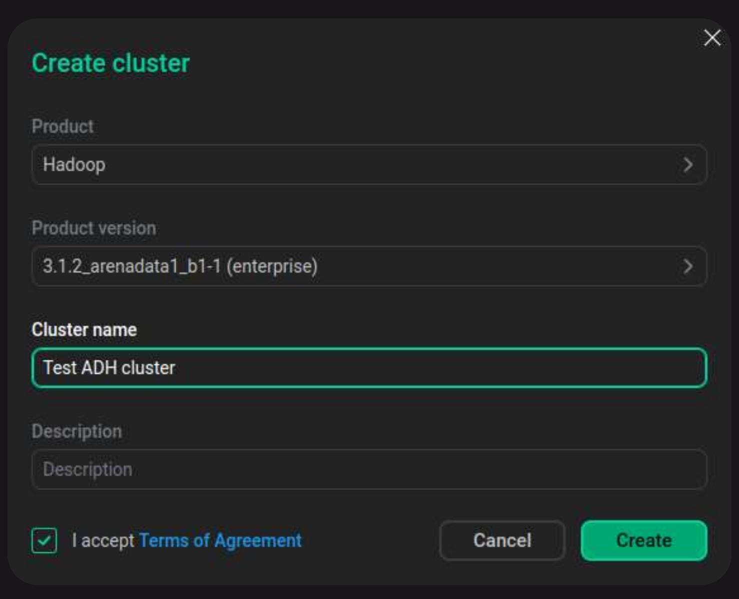
Для создания кластера необходимо выполнить шаги:

1. Выберите пункт ***Clusters*** в левом навигационном меню и нажмите ***Create cluster***.



Переход к созданию кластера

2. В открывшемся окне требуется описать новый кластер:
   - Выберите загруженный бандл в поле ***Product***.
   - Выберите версию бандла в поле ***Product version***. Несколько версий становятся доступны после загрузки нескольких бандлов для одного типа продуктов.
   - Введите имя кластера в поле ***Cluster name***.
   - При необходимости добавьте описание кластера в поле ***Description***.
   - Установите флаг ***I accept Terms of Agreement*** для подписания лицензионного соглашения. Флаг отображается только для бандлов версии Enterprise. Чтобы ознакомиться с текстом соглашения, перейдите по ссылке ***Terms of Agreement***. Подписать соглашение также возможно на странице ***Bundles***, нажав на соответствующий продуктовый бандл.
   - Нажмите ***Create***.



Create cluster

Product: Hadoop

Product version: 3.1.2_arenadata1_b1-1 (enterprise)

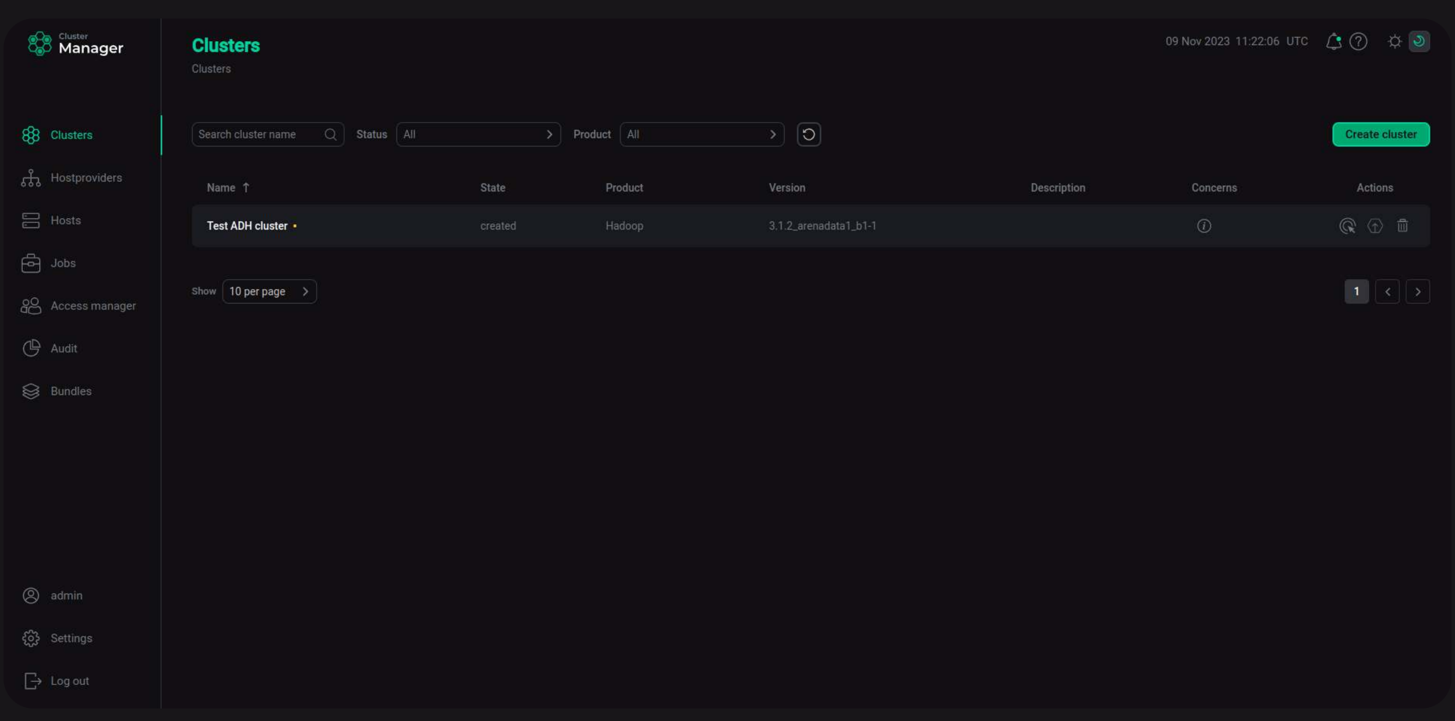
Cluster name: Test ADH cluster

Description

☑ I accept Terms of Agreement

Создание кластера

3. После успешного добавления кластер отображается на странице ***Clusters***.



Результат успешного добавления кластера

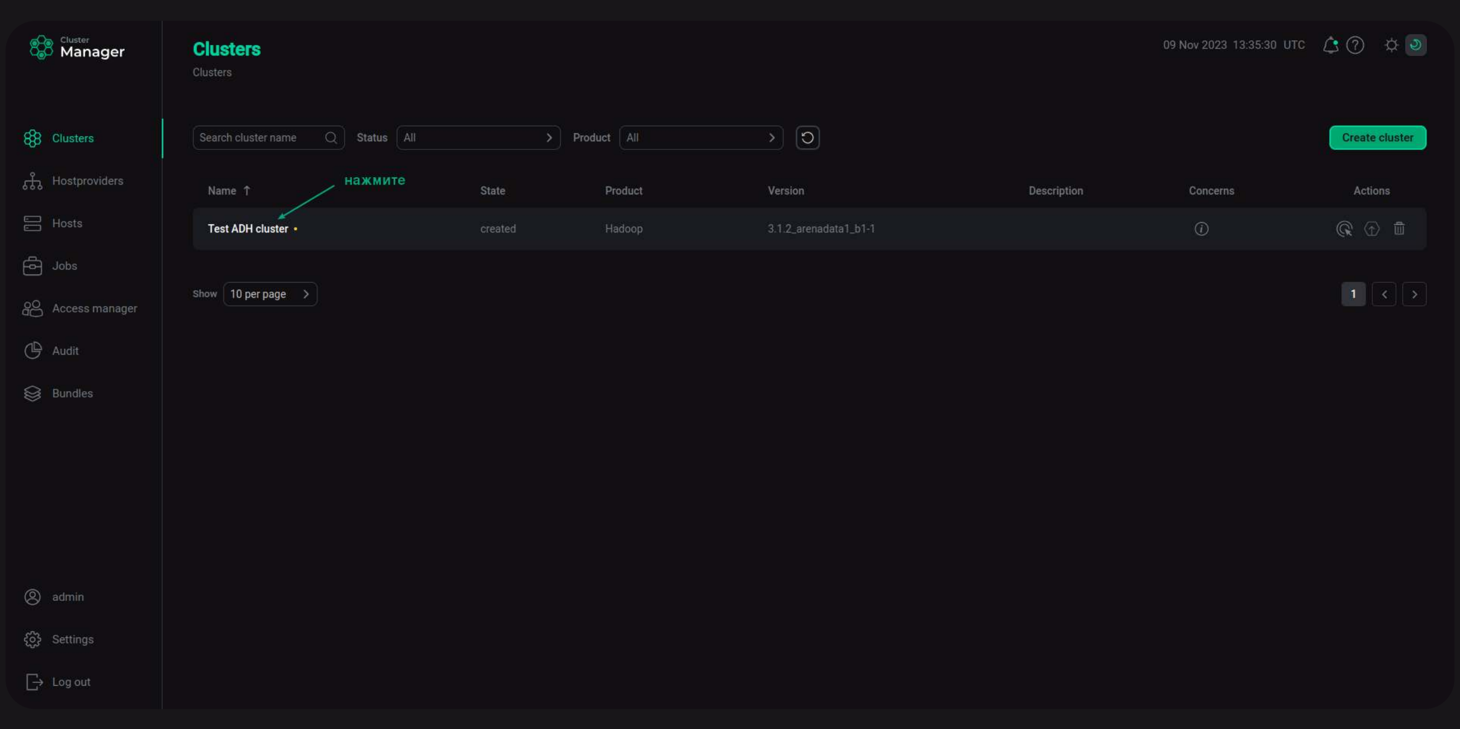
Добавление сервисов

Евгения Кузина, Дарья Барышева

Назад к содержанию

В ADCM **сервис** означает программное обеспечение, выполняющее некоторую функцию. Примеры сервисов в кластерах ADH: HDFS, HBase, Hive и другие. Для добавления сервисов в кластер:

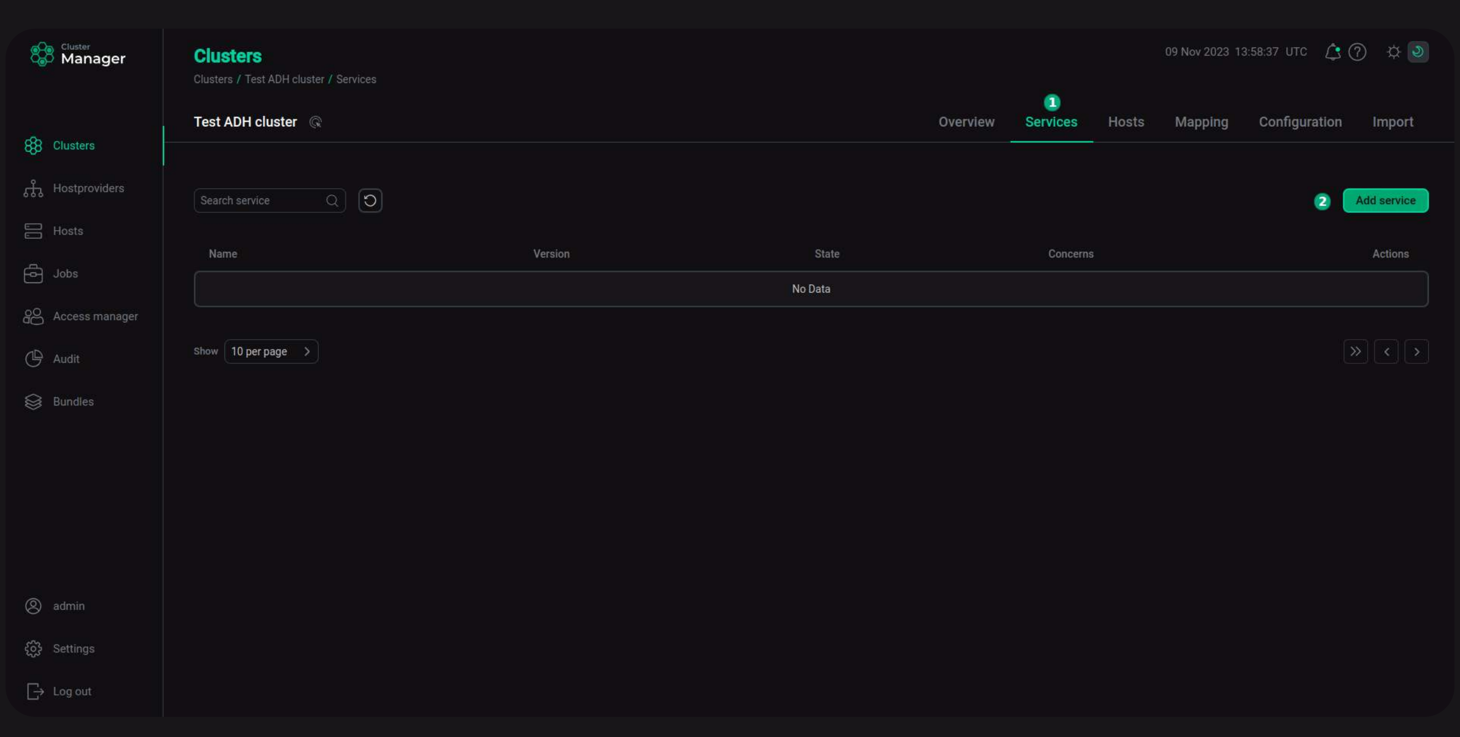
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.



Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

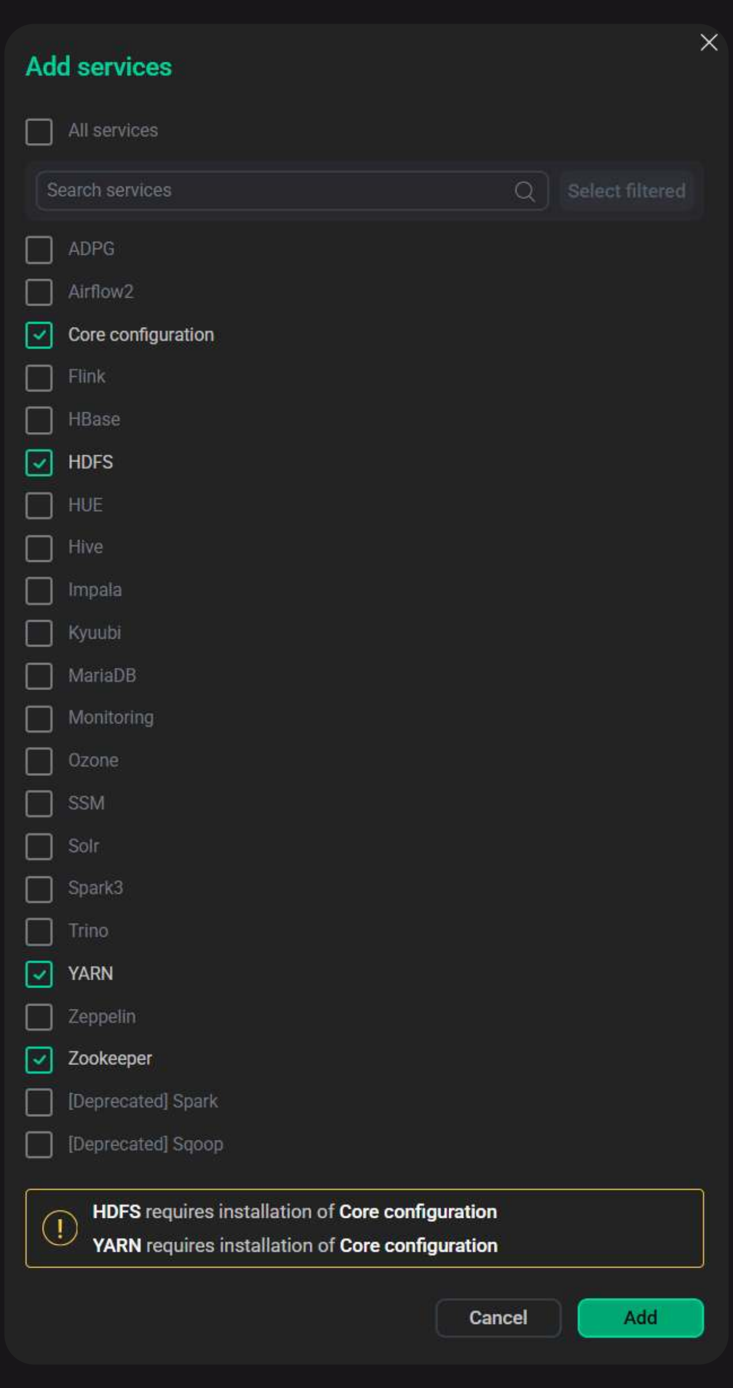
Выбор кластера

2. Откройте вкладку ***Services*** на странице кластера и нажмите ***Add service***.



Переход к добавлению сервисов

3. В открывшемся окне выберите сервисы для добавления в кластер и нажмите ***Add***.



☐ All services

☐ ADPG
☐ Airflow2
☑ Core configuration
☐ Flink
☐ HBase
☑ HDFS
☐ HUE
☐ Hive
☐ Impala
☐ Kyuubi
☐ MariaDB
☐ Monitoring
☐ Ozone
☐ SSM
☐ Solr
☐ Spark3
☐ Trino
☑ YARN
☐ Zeppelin
☑ Zookeeper
☐ [Deprecated] Spark
☐ [Deprecated] Sqoop

HDFS requires installation of **Core configuration**
YARN requires installation of **Core configuration**

Выбор сервисов

Краткое описание доступных сервисов приведено ниже.

Сервисы, которые могут быть добавлены в кластер ADH

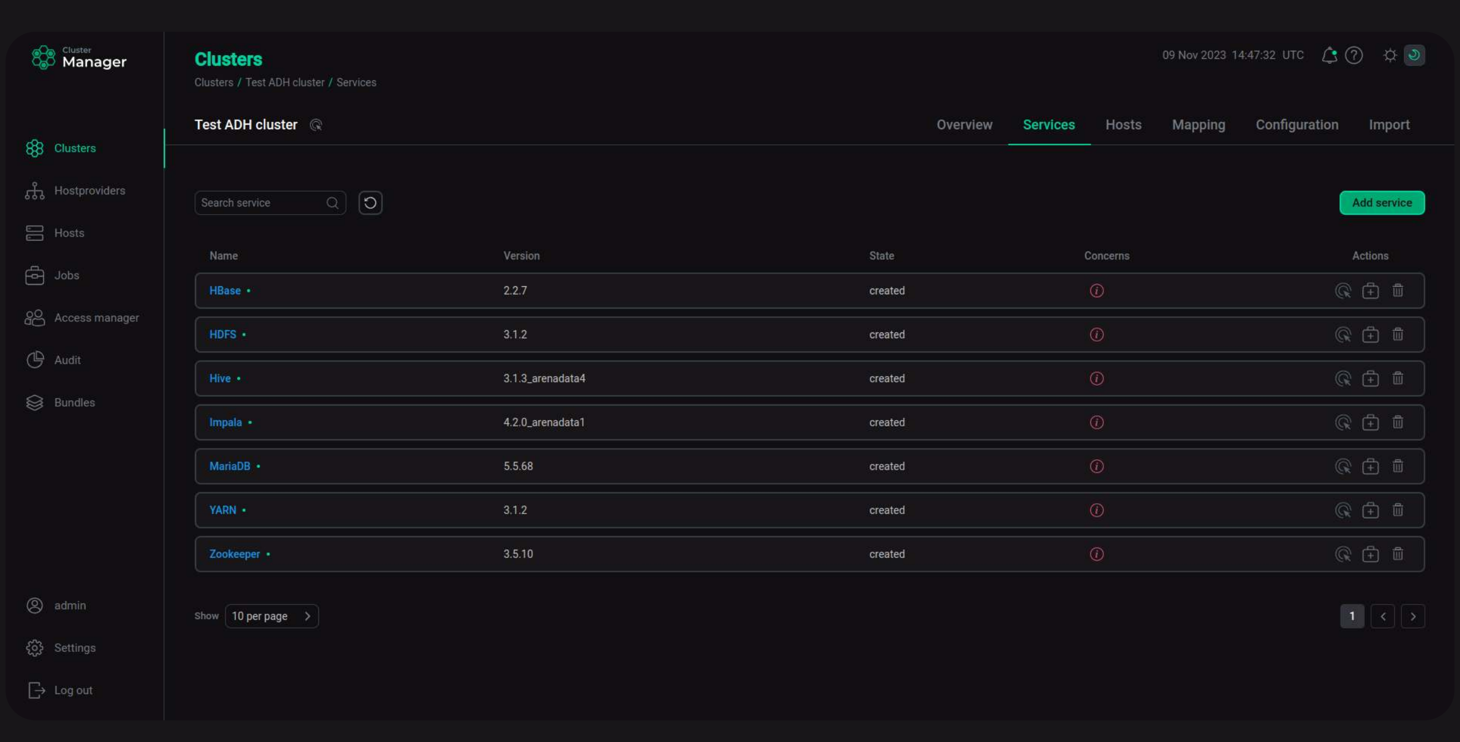
Сервис	Назначение
Airflow2	По сравнению с версией Airflow 1.x, Airflow2 предлагает дополнительные функции: высокая отказоустойчивость (High Availability), уменьшенная задержка выполнения задач, полный REST API, TaskFlow API, группы задач, независимые провайдеры и другие
Core configuration	Сервис, содержащий общие пакеты Hadoop, необходимые для всех остальных сервисов, кроме ADPG, Solr, Zookeeper и Monitoring. Этот сервис устанавливается на все узлы. Если уже установленный сервис Hadoop расширяется на другой хост, сервис Core configuration автоматически устанавливается на этот хост. Сервис Core configuration поддерживается начиная с версии ADH 3.3.6.2.b1. Конфигурационные параметры сервиса описаны в статье Конфигурационные параметры. После применения любых изменений в конфигурации сервиса необходимо выполнить его действие ***Update Core configuration***
Flink	Распределенная платформа, используемая в высоконагруженных приложениях Big Data для анализа данных, хранимых в кластерах Hyperwave. Наиболее частые сценарии использования: приложения, управляемые событиями (event-driven), потоковая и пакетная аналитика, конвейеры данных, ETL и другое
Flink2	Сервис Flink с версией 2.x. Сервис добавлен в режиме технологического превью, может иметь ряд ограничений и не предназначен для использования в производственных средах
HBase	Нереляционная распределенная база данных, написанная на Java и работающая поверх HDFS. Относится к классу колоночных СУБД, хранящих данные в формате key-value. Используется для произвольного доступа к большим данным на чтение и запись в режиме реального времени
HDFS	Распределенная файловая система, используемая в кластерах Hyperwave для хранения больших файлов. Обеспечивает возможность потокового доступа к информации, распределенной поблочно по нодам кластера
Hive	Сервис, предназначенный для создания корпоративных хранилищ данных (Data Warehouse, DWH) и анализа Big Data. Работает поверх HDFS и других совместимых систем, таких как Apache HBase. Использование Hive облегчает запись, чтение и управление большими наборами данных, хранимыми в распределенных системах
HUE	Apache HUE — это SQL-помощник с графическим интерфейсом, который предоставляет интерпретаторы для различных источников данных, таких как Impala, Hive, Kyuubi и многих других. Он обладает удобным редактором запросов с умной автоподстановкой, функцией "drag and drop", меню быстрой навигации и другими полезными функциями
Impala	Impala обеспечивает быстрые интерактивные SQL-запросы к данным, хранящимся в HDFS, HBase или S3-хранилище. В дополнение к унифицированной платформе хранения Impala также использует те же метаданные, синтаксис SQL (Hive SQL) и драйвер JDBC, что и Apache Hive. Это делает Impala унифицированной платформой для запросов в режиме реального времени или пакетных запросов
Kyuubi	Apache Kyuubi — это распределенный многопользовательский шлюз для предоставления SQL для DWH и DataLake. Kyuubi создает распределенные механизмы запросов SQL поверх различных видов современных вычислительных платформ, например, Apache Spark, Flink, Hive, Impala и т. д., чтобы получать и обрабатывать большие наборы распределенных данных из разнородных источников
Monitoring	Сервис, добавляемый в случаях, когда запланирована настройка мониторинга ADH через ADCM
Ozone	Apache Ozone (O3) — распределенное хранилище объектов, оптимизированное для работы как с сервисами Hadoop, так и с хранилищами S3. Ozone обладает высокой масштабируемостью, может использовать те же политики безопасности кластера, что и HDFS, и способен эффективно обрабатывать файлы любого размера
Solr	Поисковая платформа, основанная на проекте Apache Lucene. Основные функции Solr включают полнотекстовый поиск, фасетный поиск, выделение результатов поиска (highlighting), распределенное индексирование, интеграцию с базами данных, обработку документов со сложными форматами (Word, PDF и так далее), запросы с балансировкой нагрузки, централизованную настройку и другое
Spark3	Аналитический фреймворк, используемый для быстрой обработки больших объемов данных. Spark3 может работать в кластерах Hyperwave, используя standalone-режим YARN или Spark. Он может обрабатывать данные, поступающие в форматах HDFS, HBase, Cassandra, Hive и любых других форматах, поддерживаемых в Hadoop. Применяется для пакетной обработки и других задач: потоковой передачи, машинного обучения, интерактивных запросов и так далее. Версия Spark 3.x предлагает дополнительные функции: адаптивное выполнение Spark SQL, Dynamic Partition Pruning (DPP), обработку графов, расширенную поддержку Deep Learning и другое
Spark4	Сервис Spark с версией 4.x. Сервис добавлен в режиме технологического превью, может иметь ряд ограничений и не предназначен для использования в производственных средах
SSM	Smart Storage Manager — это сервис, цель которого — оптимизировать эффективность хранения и управления данными в Hadoop Distributed File System. SSM собирает данные о работе HDFS и информацию о состоянии системы и на основе собранных показателей может автоматически использовать такие методологии как кеш, политики хранения данных, управление гетерогенными хранилищами (HSM), сжатие данных и Erasure Coding. Кроме того, SSM предоставляет возможность настройки асинхронной репликации данных и пространства имен на резервный кластер с целью организации DR
Trino	Trino — это движок SQL-запросов с открытым исходным кодом, используемый для параллельной обработки данных, распределенных по хранилищам разного типа, таких как объектные хранилища (S3), базы данных и файловые системы
YARN	Сервис, необходимый для управления ресурсами кластера и планирования/мониторинга задач (jobs). Использует специальный демон (Resource Manager), абстрагирующий все вычислительные ресурсы кластера и управляющий их предоставлением распределенным приложениям
Zeppelin	Сервис, который играет роль web-блокнота и обеспечивает интерактивный анализ данных. Позволяет создавать запросы к данным в кластерах Hyperwave и отображать результаты в виде таблиц, графиков, диаграмм и так далее
Zookeeper	Сервис централизованного управления распределенными приложениями. Он используется в кластерах Hyperwave для обнаружения сбоев, выбора активных NameNode, мониторинга работоспособности (health checks), управления сеансами и так далее

Минимальный набор сервисов, рекомендуемый для кластеров ADH, приведен ниже:

- Core configuration;
- HDFS/Ozone;
- YARN;
- Zookeeper (опционально для Community-версии ADH).

Эти сервисы составляют ядро Hadoop и являются достаточными для организации распределенного хранения и обработки данных. Полный перечень сервисов будет зависеть от требований конкретного проекта.

4. В результате успешно добавленные сервисы отображаются на вкладке *Services*.



Name	Version	State	Concerns	Actions
HBase	2.2.7	created		
HDFS	3.1.2	created		
Hive	3.1.3_arenadata4	created		
Impala	4.2.0_arenadata1	created		
MariaDB	5.5.68	created		
YARN	3.1.2	created		
Zookeeper	3.5.10	created		

Результат успешного добавления сервисов в кластер

> **ПРИМЕЧАНИЕ**
>
> Опциональные сервисы могут быть добавлены в кластер позднее. Процесс добавления сервисов в уже развернутый кластер не отличается от описанного выше.

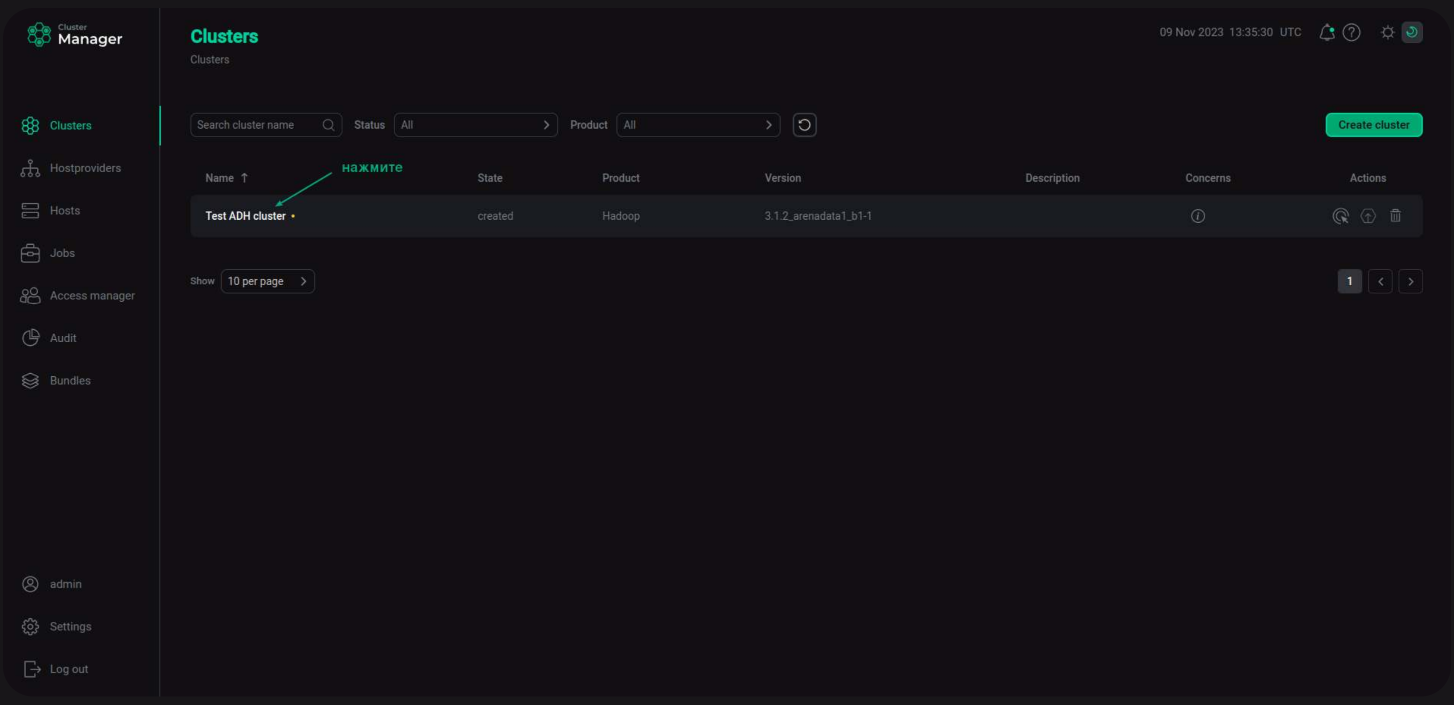
Добавление хостов в кластер

Евгения Кузина, Дарья Барышева

Назад к содержанию

Для добавления хостов в кластер:

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
09 Nov 2023 13:35:30 UTC
Create cluster
нажмите
Show
1

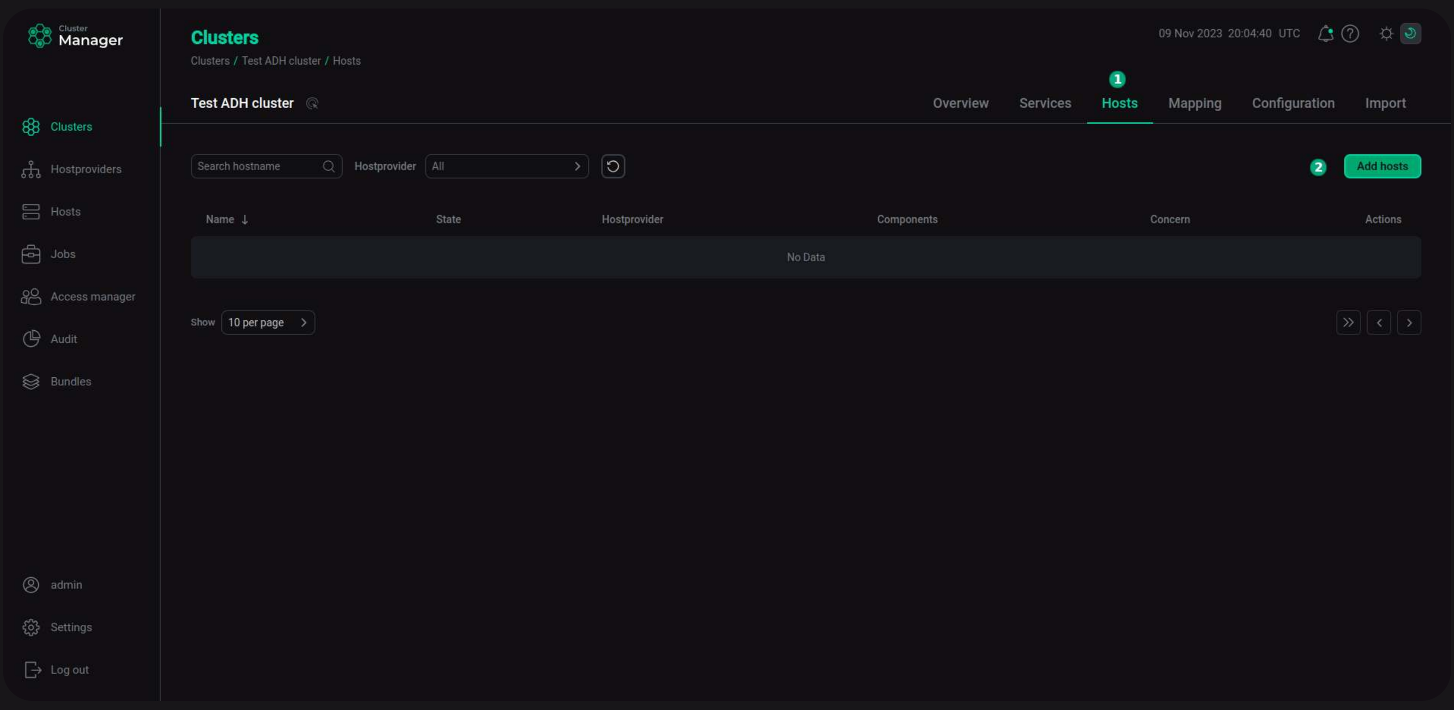
Search cluster name | Status: All | Product: All

Name ↑	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

Show 10 per page

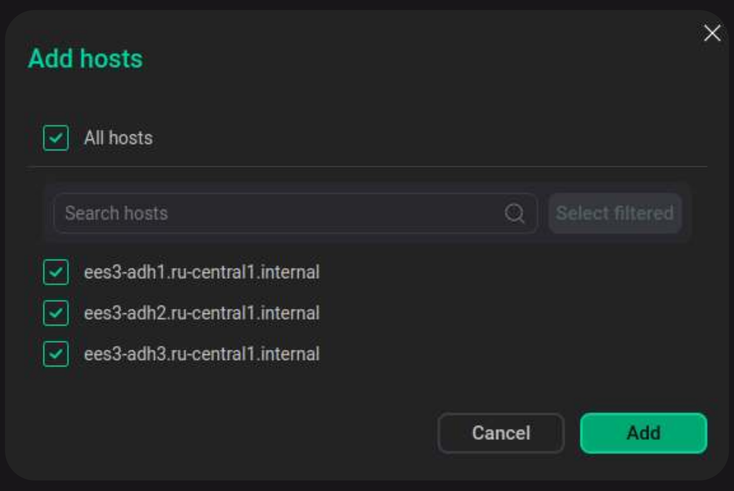
Выбор кластера

2. Откройте вкладку ***Hosts*** на странице кластера и нажмите ***Add hosts***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test ADH cluster / Hosts
09 Nov 2023 20:04:40 UTC
Test ADH cluster
Overview
Services
1
Hosts
Mapping
Configuration
Import
Search hostname
Hostprovider
All
2
Add hosts
Name ↓
State
Hostprovider
Components
Concern
Actions
No Data
Show
10 per page

Переход к добавлению хостов

3. В открывшемся окне выберите хосты для добавления в кластер и нажмите ***Add***.

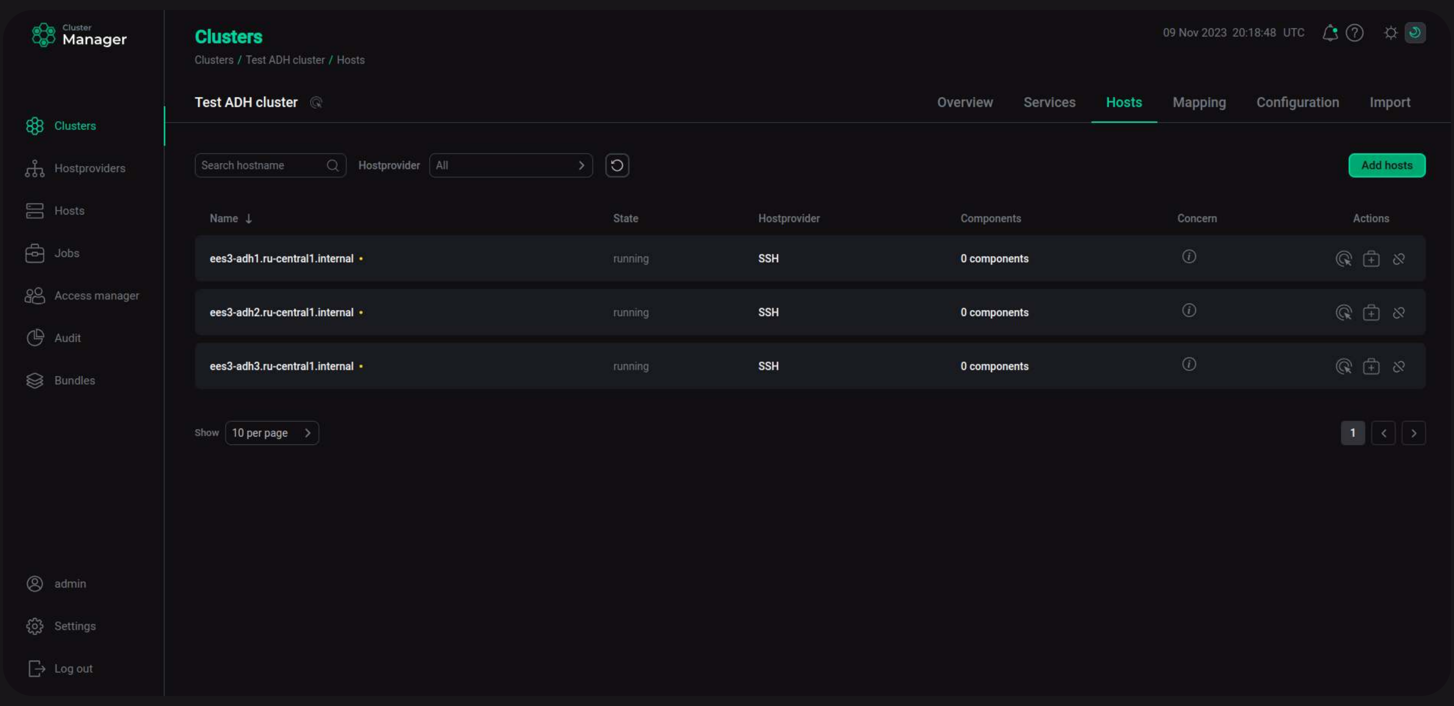

Add hosts
Search hosts
Select filtered
Cancel
Add

☑ All hosts

☑ ees3-adh1.ru-central1.internal
☑ ees3-adh2.ru-central1.internal
☑ ees3-adh3.ru-central1.internal

Выбор хостов

4. В результате успешно добавленные хосты отображаются на вкладке ***Hosts***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test ADH cluster / Hosts
09 Nov 2023 20:18:48 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Add hosts
Show
1

Search hostname | Hostprovider: All

Name ↓	State	Hostprovider	Components	Concern	Actions
ees3-adh1.ru-central1.internal	running	SSH	0 components		
ees3-adh2.ru-central1.internal	running	SSH	0 components		
ees3-adh3.ru-central1.internal	running	SSH	0 components		

Show 10 per page

Результат успешного добавления хостов в кластер

Добавление компонентов

Евгения Кузина, Дарья Барышева

Назад к содержанию

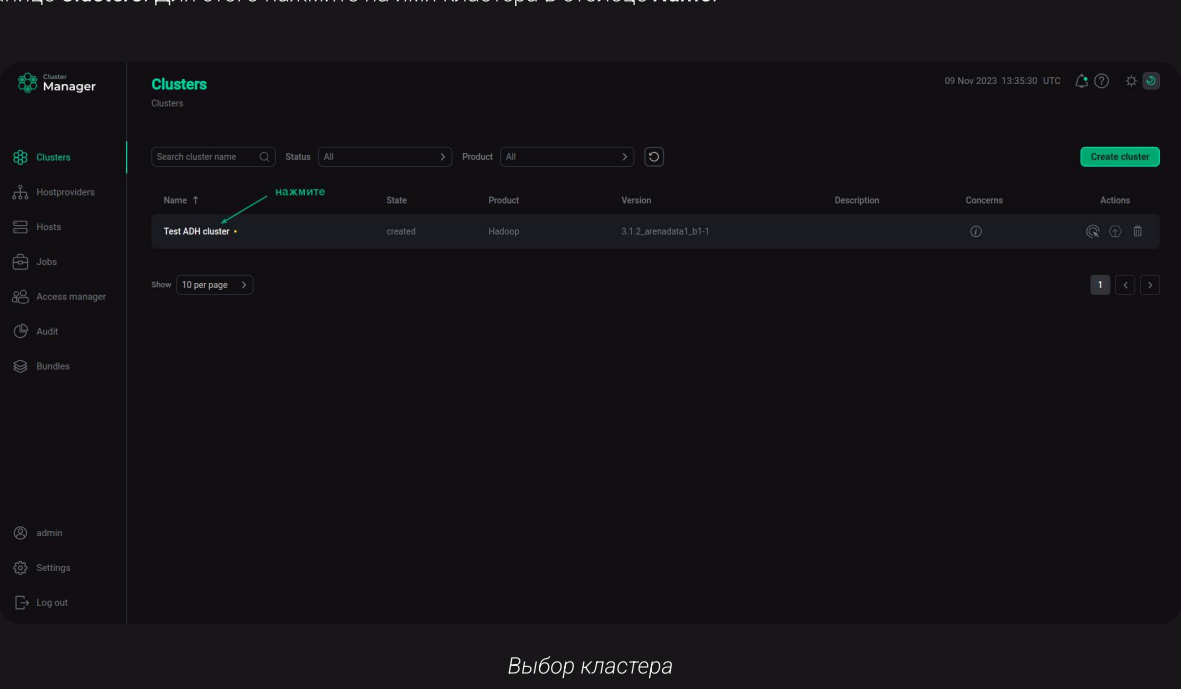
В ADCM **компонент** означает часть сервиса, которая должна быть развернута на одном или нескольких хостах кластера. Например, сервис HDFS состоит из таких компонентов, как DataNode, NameNode, Client и другие.

ВАЖНО

Изначально на хостах кластера нет ни одного компонента — распределение компонентов между хостами должно быть проведено вручную.

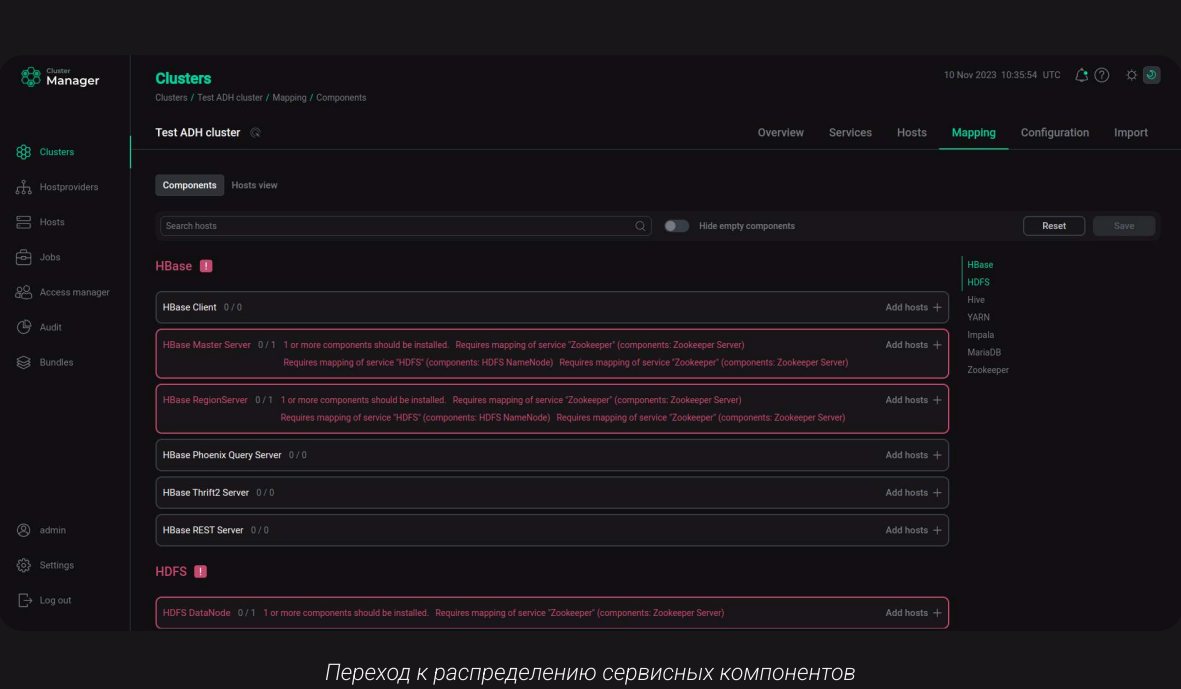
Для добавления компонентов на хосты кластера:

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.



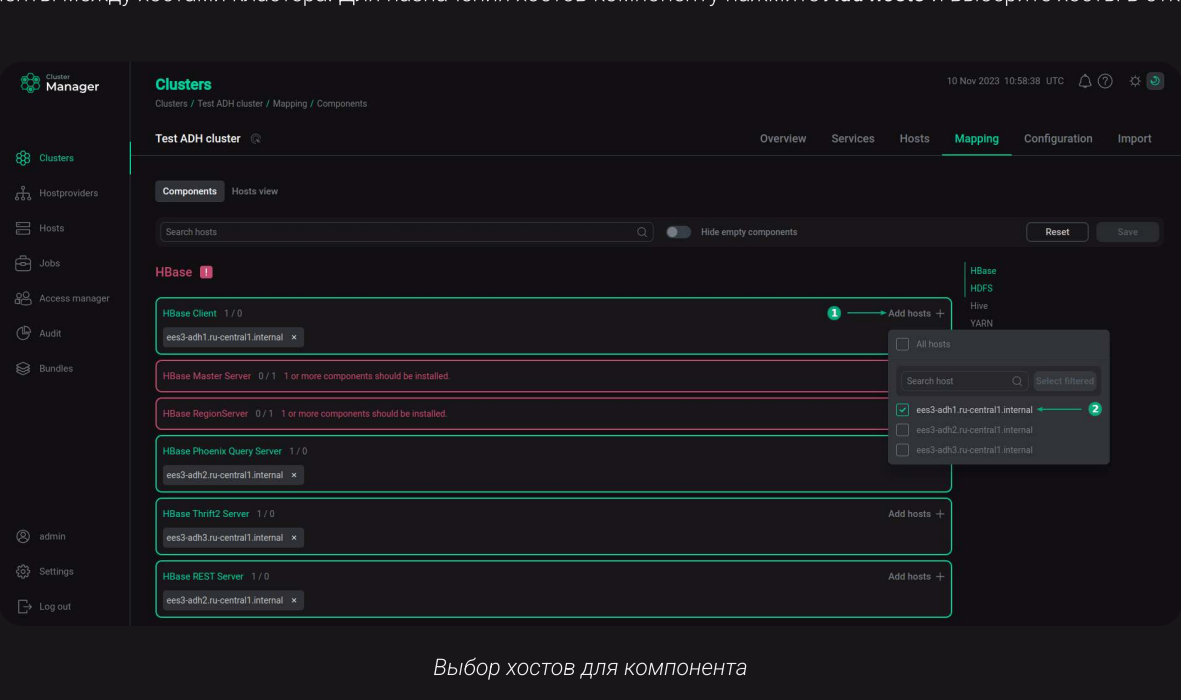
Выбор кластера

2. Откройте вкладку ***Mapping*** на странице кластера.



Переход к распределению сервисных компонентов

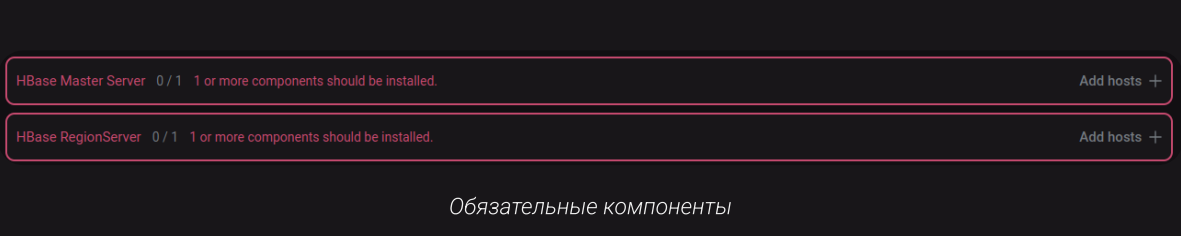
3. Распределите все компоненты между хостами кластера. Для назначения хостов компоненту нажмите ***Add hosts*** и выберите хосты в открывшемся окне.



Выбор хостов для компонента

Обязательные компоненты

Каждый сервис может иметь обязательные и опциональные компоненты. Обязательные подсвечиваются красным цветом. Также в интерфейсе ADCM для них выводится информация о том, в каком количестве необходимо установить каждый компонент.



HBase Master Server 0 / 1 1 or more components should be installed.

HBase RegionServer 0 / 1 1 or more components should be installed.

Обязательные компоненты

ВНИМАНИЕ

Без назначения нужного числа хостов обязательным компонентам сохранение общей схемы распределения невозможно.

Следующая таблица содержит примеры распределения компонентов в кластерах ADH, состоящих из 3, 6 и 9 хостов. Все примеры основаны на Enterprise-версии ADH.

Примеры распределения компонентов

3 хоста

Хост	Сервис	Компонент
Хост 1	Core configuration	Configuration server
	Flink	Flink Client Flink JobManager Flink TaskManager Flink SQL-Gateway
	HBase	HBase Client HBase RegionServer
	HDFS	HDFS Client HDFS DataNode HDFS JournalNode HDFS NameNode HDFS ZKFC
	Hive	Hive Client Hive Metastore Hive Tez
	Impala	Impala Daemon Impala Statestore Impala Catalog Service
	Monitoring	Grafana Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Recon Ozone HttpFS Ozone Datanode
	Solr	Solr Server
	Spark3	Spark3 Client Spark3 History server
	Trino	Trino Coordinator Trino Client
	YARN	MapReduce History server YARN Client YARN NodeManager
	ZooKeeper	ZooKeeper Server
Хост 2	Airflow2	Airflow2 Broker Airflow2 Flower Airflow2 Scheduler Airflow2 Server Airflow2 Worker
	Flink	Flink Client Flink History Server
	HBase	HBase Client HBase Master server HBase Phoenix query server HBase RegionServer
	HDFS	HDFS Client HDFS DataNode HDFS HttpFS server HDFS JournalNode
	Hive	Hive Client Hive HiveServer2 Hive Tez Hive TezUI
	Impala	Impala Daemon
	Monitoring	Prometheus Server Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Datanode
	Solr	Solr Server Solr prometheus exporter
	Spark3	Spark3 Client Spark3 Livy server
	Trino	Trino Worker Trino Client
	YARN	YARN Client YARN NodeManager YARN ResourceManager
	Zeppelin	Zeppelin Server
	ZooKeeper	ZooKeeper Server
Хост 3	Arenadata PostgreSQL	Arenadata PostgreSQL
	Flink	Flink Client
	HBase	HBase Client HBase RegionServer HBase Thrift2 server HBase REST Server
	HDFS	HDFS Client HDFS DataNode HDFS JournalNode HDFS NameNode HDFS ZKFC
	Hive	Hive Client Hive Tez
	HUE	HUE Server
	Impala	Impala Daemon Impala Client
	Kyuubi	Kyuubi Server
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone S3G Ozone Datanode
	Solr	Solr Server
	Spark3	Spark3 Client
	SSM	SSM Server SSM Agent
	Trino	Trino Worker Trino Client
	YARN	YARN Client YARN NodeManager YARN Timeline server
	ZooKeeper	ZooKeeper Server

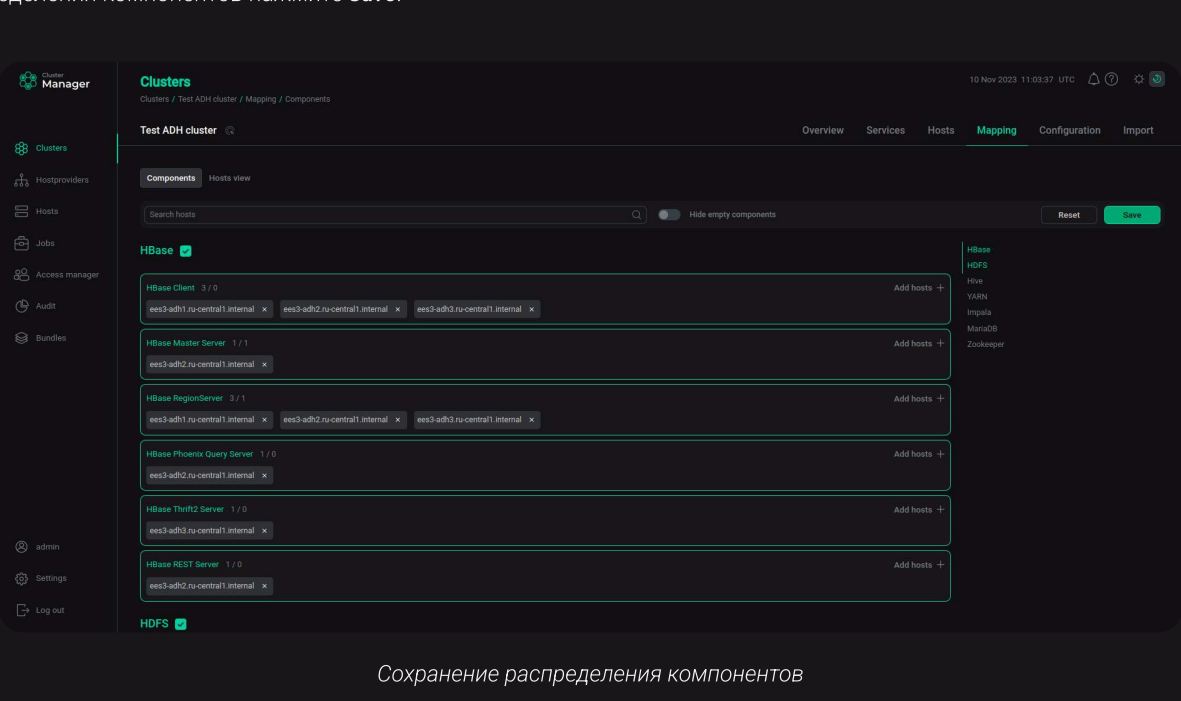
6 хостов

Хост	Сервис	Компонент
Хост 1	Core configuration	Configuration server
	Flink	Flink JobManager Flink TaskManager Flink SQL-Gateway
	HDFS	HDFS JournalNode HDFS NameNode HDFS ZKFC
	Hive	Hive Metastore
	Impala	Impala Statestore Impala Catalog Service
	Monitoring	Grafana Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Recon Ozone HttpFS
	Solr	Solr Server Solr prometheus exporter
	Spark3	Spark3 History server
	Trino	Trino Coordinator Trino Client
	YARN	MapReduce History server
	ZooKeeper	ZooKeeper Server
Хост 2	Airflow2	Airflow2 Broker Airflow2 Flower Airflow2 Scheduler Airflow2 Server Airflow2 Worker
	HBase	HBase Master server HBase Phoenix query server
	HDFS	HDFS JournalNode HDFS HttpFS server
	Hive	Hive HiveServer2 Hive TezUI
	Monitoring	Prometheus Server Node Exporter
	Solr	Solr Server
	Spark3	Spark3 Livy server
	Trino	Trino Worker Trino Client
	YARN	YARN ResourceManager
	Zeppelin	Zeppelin Server
	ZooKeeper	ZooKeeper Server
Хост 3	Arenadata PostgreSQL	Arenadata PostgreSQL
	HBase	HBase Thrift2 server HBase REST Server
	HDFS	HDFS NameNode HDFS ZKFC HDFS JournalNode
	HUE	HUE Server
	Kyuubi	Kyuubi Server
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Solr	Solr Server
	SSM	SSM Server SSM Agent
	YARN	YARN Timeline server
	ZooKeeper	ZooKeeper Server
Хост 4	Flink	Flink Client Flink History Server
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon Impala Client
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Datanode
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 5	Flink	Flink Client
	HBase	HBase Client HBase RegionServer
	HDFS	HDFS DataNode HDFS Client
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 6	Flink	Flink Client
	HBase	HBase Client HBase RegionServer
	HDFS	HDFS DataNode HDFS Client
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone S3G
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server

9 хостов

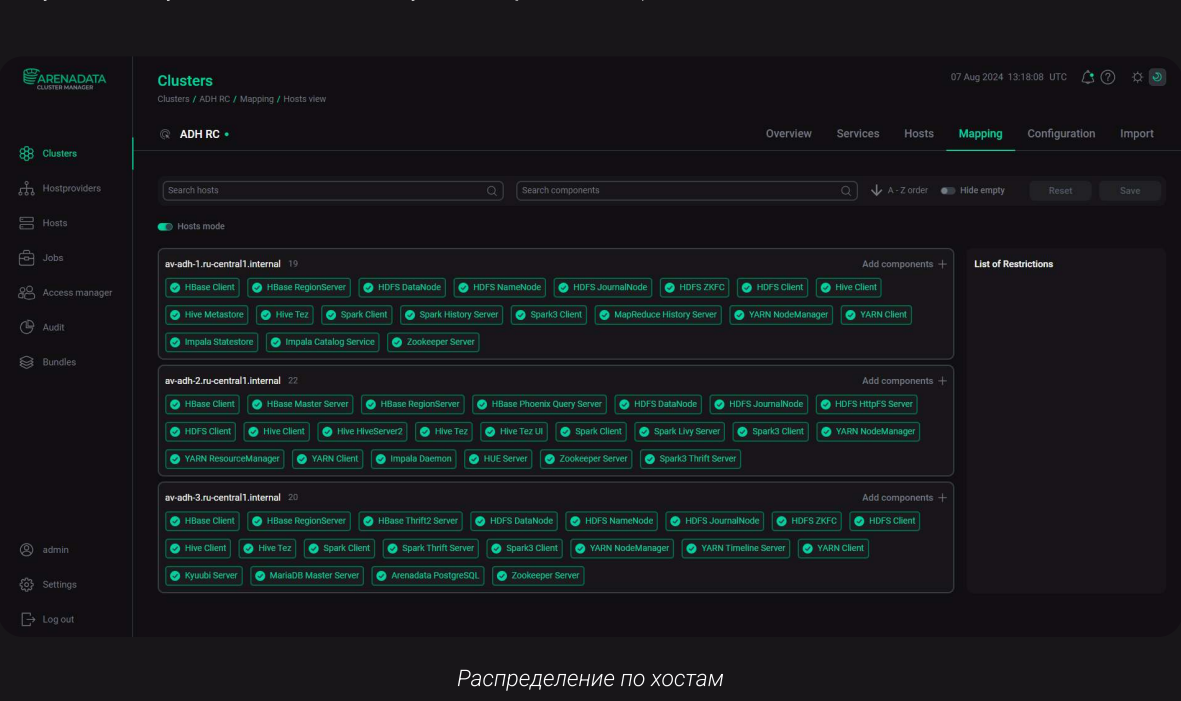
Хост	Сервис	Компонент
Хост 1	Core configuration	Configuration server
	Flink	Flink JobManager Flink TaskManager Flink SQL-Gateway
	HDFS	HDFS JournalNode HDFS NameNode HDFS ZKFC
	Hive	Hive Metastore
	Impala	Impala Statestore Impala Catalog Service
	Monitoring	Grafana Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Recon Ozone HttpFS
	Solr	Solr Server
	Spark3	Spark3 History server
	Trino	Trino Coordinator Trino Client
	YARN	MapReduce History server
	ZooKeeper	ZooKeeper Server
Хост 2	Airflow2	Airflow2 Broker Airflow2 Flower Airflow2 Scheduler Airflow2 Server Airflow2 Worker
	Flink	Flink TaskManager
	HBase	HBase Master server HBase Phoenix query server
	HDFS	HDFS JournalNode HDFS HttpFS server
	Hive	Hive HiveServer2 Hive TezUI
	Monitoring	Prometheus Server Node Exporter
	Solr	Solr Server Solr prometheus exporter
	Spark3	Spark3 Livy server
	YARN	YARN ResourceManager
	Zeppelin	Zeppelin Server
	ZooKeeper	ZooKeeper Server
Хост 3	HBase	HBase Thrift2 server HBase REST Server
	HDFS	HDFS NameNode HDFS ZKFC HDFS JournalNode
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone Datanode
	Solr	Solr Server
	Trino	Trino Worker Trino Client
	YARN	YARN Timeline server
	Zeppelin	Zeppelin Server
	ZooKeeper	ZooKeeper Server
Хост 4	Arenadata PostgreSQL	Arenadata PostgreSQL
	HDFS	HDFS JournalNode HDFS NameNode HDFS ZKFC HDFS HttpFS server
	HBase	HBase Master server
	HUE	HUE Server
	Kyuubi	Kyuubi Server
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Solr	Solr Server
	SSM	SSM Server SSM Agent
	Trino	Trino Worker Trino Client
	ZooKeeper	ZooKeeper Server
Хост 5	HDFS	HDFS JournalNode HDFS NameNode HDFS ZKFC
	HBase	HBase Master server
	Monitoring	Node Exporter
	Ozone	Ozone Manager Ozone Storage Container Manager Ozone S3G
	Solr	Solr Server
	Trino	Trino Worker Trino Client
	ZooKeeper	ZooKeeper Server
Хост 6	Flink	Flink Client Flink History Server
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon Impala Client
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Solr	Solr Server
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 7	Flink	Flink Client
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Solr	Solr Server
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 8	Flink	Flink Client
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server
Хост 9	Flink	Flink Client
	HDFS	HDFS DataNode HDFS Client
	HBase	HBase Client HBase RegionServer
	Hive	Hive Tez Hive Client
	Impala	Impala Daemon
	Monitoring	Node Exporter
	Ozone	Ozone Datanode
	Spark3	Spark3 Client
	Trino	Trino Worker Trino Client
	YARN	YARN NodeManager YARN Client
	ZooKeeper	ZooKeeper Server

4. После завершения распределения компонентов нажмите ***Save***.



Сохранение распределения компонентов

Начиная с версии 2.2.0 ADCM поддерживает альтернативный способ сопоставления — по хостам. Для этого переведите в активное положение переключатель ***Hosts mode*** и распределите компоненты между хостами путем нажатия на кнопку ***Add components*** напротив каждого хоста.



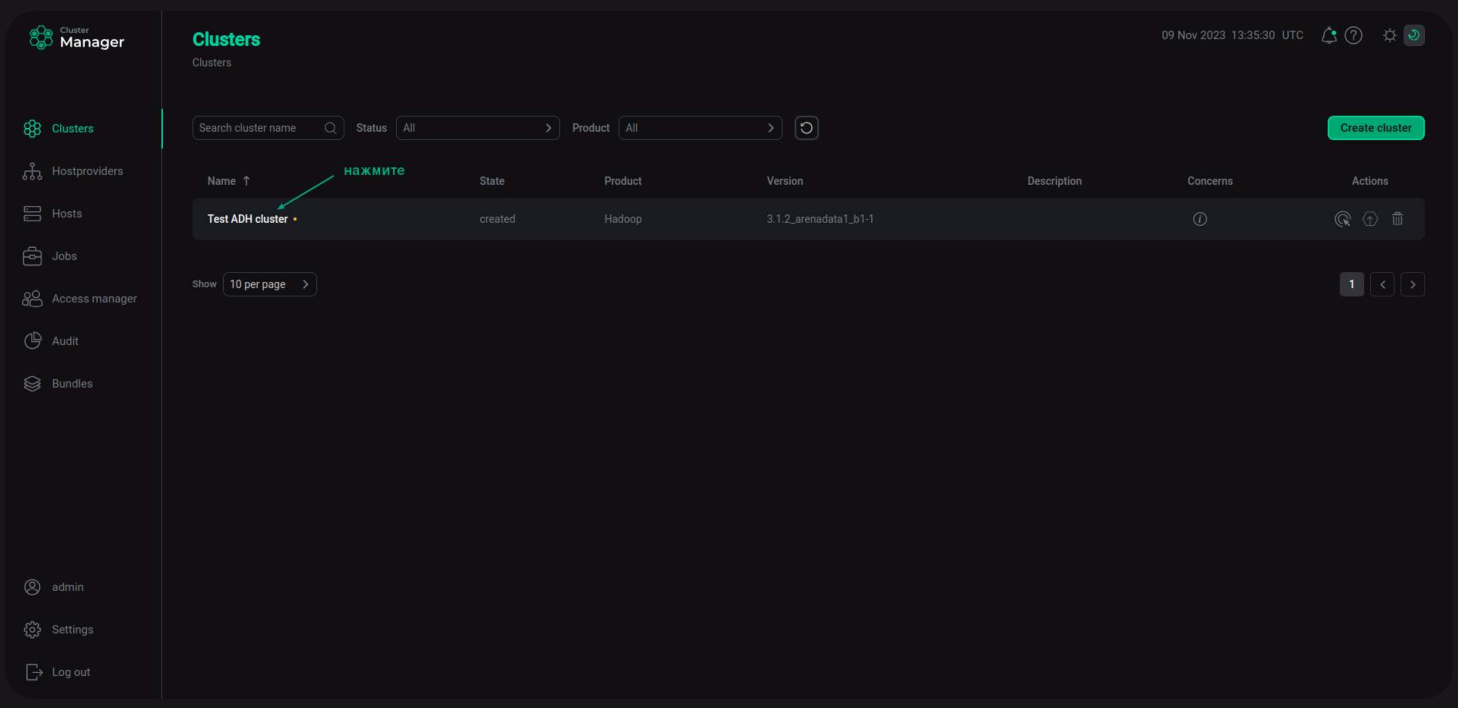
Распределение по хостам

Настройка сервисов

Евгения Кузина, Дарья Барышева

Назад к содержанию

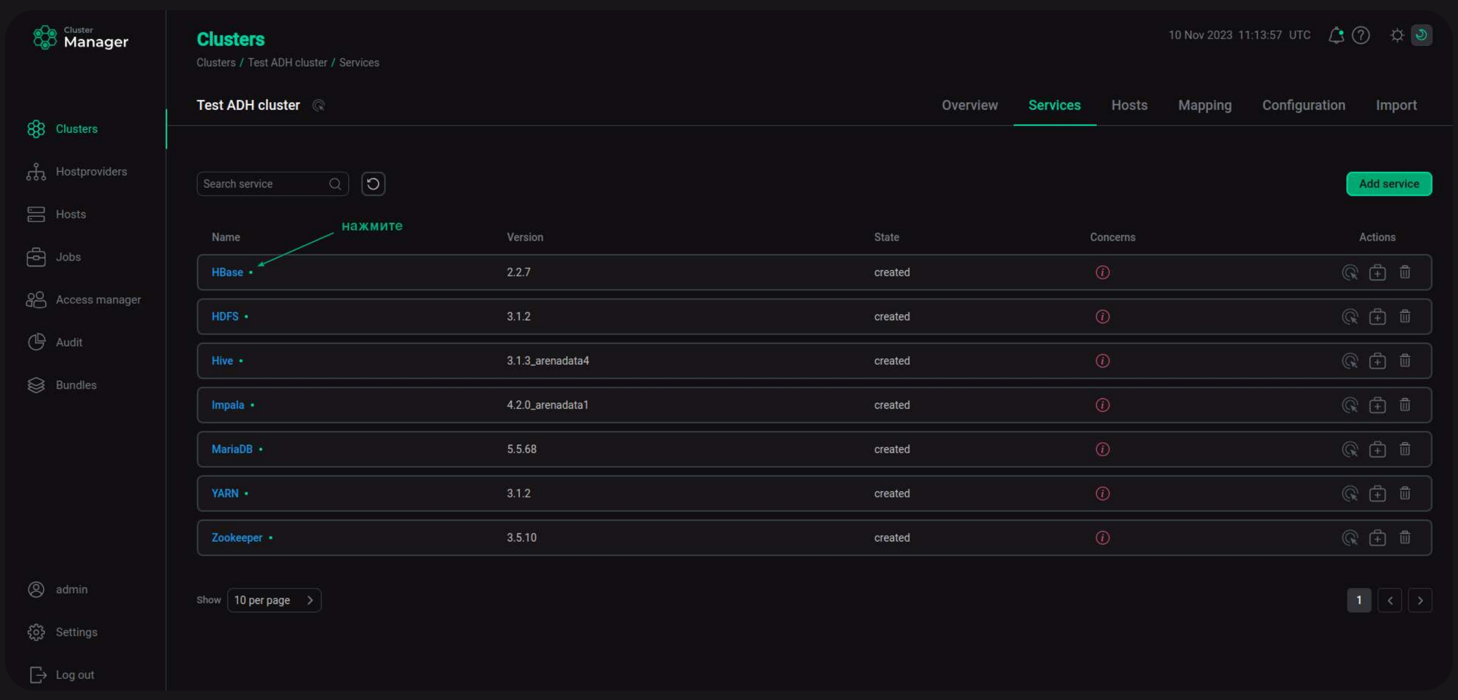
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
09 Nov 2023 13:35:30 UTC
Search cluster name
Status
All
Product
All
Create cluster
нажмите
Show
10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

Выбор кластера

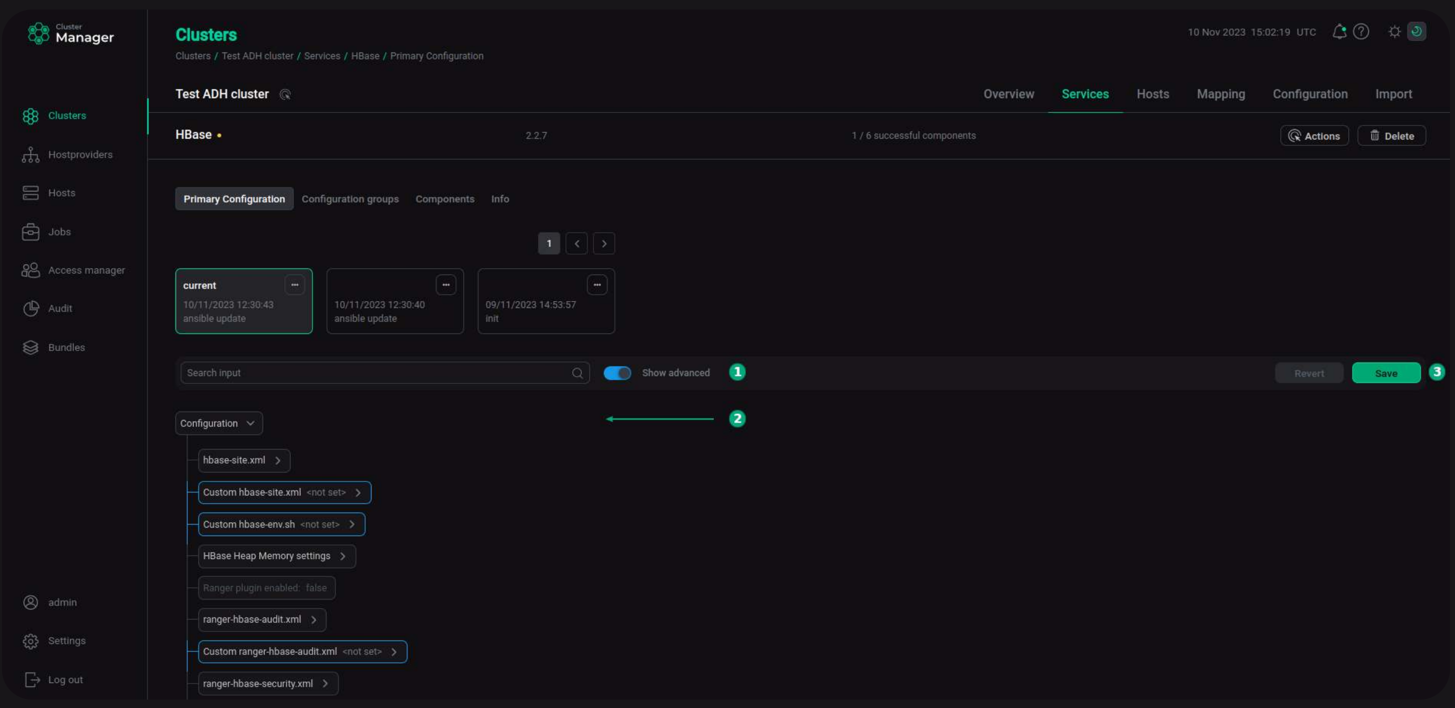
2. Откройте вкладку ***Services*** на странице кластера. Нажмите на имя сервиса в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test ADH cluster / Services
10 Nov 2023 11:13:57 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add service
нажмите
Show
10 per page
1

Name	Version	State	Concerns	Actions
HBase	2.2.7	created		
HDFS	3.1.2	created		
Hive	3.1.3_arenadata4	created		
Impala	4.2.0_arenadata1	created		
MariaDB	5.5.68	created		
YARN	3.1.2	created		
Zookeeper	3.5.10	created		

Переход к настройке сервиса

3. В открывшемся окне:
   - Откройте вкладку ***Primary Configuration***.
   - Переведите в активное состояние переключатель ***Show advanced***.
   - Заполните конфигурационные параметры выбранного сервиса. Поля, подсвеченные красным, обязательны для заполнения.
   - Нажмите ***Save***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test ADH cluster / Services / HBase / Primary Configuration
10 Nov 2023 15:02:19 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
HBase
2.2.7
1 / 6 successful components
Actions
Delete
Primary Configuration
Configuration groups
Components
Info
1
current
10/11/2023 12:30:43
ansible update
10/11/2023 12:30:40
ansible update
09/11/2023 14:53:57
init
Search input
Show advanced
1
Revert
Save
3
2
Configuration
hbase-site.xml
Custom hbase-site.xml <not set>
Custom hbase-env.sh <not set>
HBase Heap Memory settings
Ranger plugin enabled: false
ranger-hbase-audit.xml
Custom ranger-hbase-audit.xml <not set>
ranger-hbase-security.xml

Настройка сервиса

ПРИМЕЧАНИЕ

Более подробную информацию по всем конфигурационным параметрам можно получить в статье Конфигурационные параметры.

Настройка кластера

Евгения Кузина, Дарья Барышева

Содержание

Назад к содержанию

Primary configuration

Configuration groups

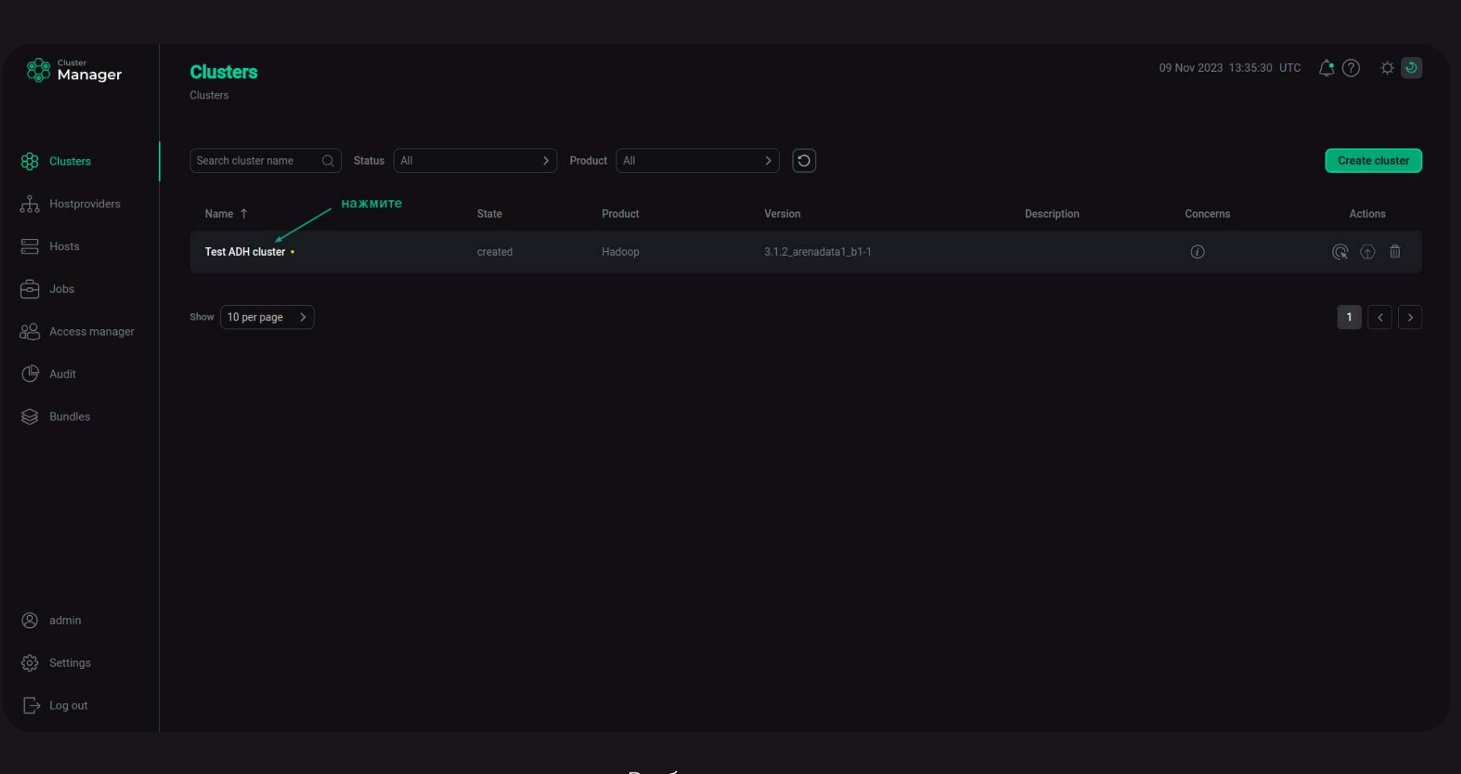
Ansible settings

ВАЖНО

В большинстве случаев кастомная настройка кластера не требуется — значения параметров можно оставить по умолчанию.

После добавления нового кластера можно выполнить его настройку. Для этого выполните шаги:

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.



Выбор кластера

2. Откройте вкладку ***Configuration*** на странице кластера. Вкладка ***Configuration*** состоит из трех подразделов: ***Primary configuration***, ***Configuration groups***, ***Ansible settings***.
3. Заполните необходимые конфигурационные параметры на нужной вкладке и нажмите ***Save***.

Primary configuration

Вкладка ***Configuration*** → ***Primary configuration*** содержит основные конфигурационные параметры кластера.

Доступные для редактирования параметры перечислены ниже:

- Раздел ***Reliability control***. Отображается, если активирован переключатель ***Show advanced***. Раздел содержит следующие настройки:
  - ***retries*** — количество попыток для операций в кластере (по умолчанию 20);
  - ***delay*** — задержка между попытками в секундах (по умолчанию 10);
  - ***timeout*** — тайм-аут проверки сетевых портов в секундах (по умолчанию 60).
- Пути к репозиториям, которые будут использованы в процессе установки в зависимости от операционной системы кластера (CentOS 7, Red Hat 7, Ubuntu 22, ALT Linux 8, Astra Linux, RedOS 7):
  - ***monitoring*** — репозиторий для установки мониторинга;
  - ***arenadata*** — YUM-репозиторий Arenadata;
  - ***arenadata_postgres*** — YUM-репозиторий Arenadata PostgreSQL;
  - ***prometheus_repo*** — YUM-репозиторий Prometheus;
  - ***arenadata_enterprise*** — YUM-репозиторий Arenadata Enterprise;
- Параметр ***DockerRegistry***, указывающий адрес Docker Registry. Значение по умолчанию — *hub.adsw.io* — Docker Registry из репозитория Arenadata.
- Раздел ***Kerberos***, содержащий параметры, перечисленные в таблице ниже. Некоторые параметры становятся видимыми только при установленном флажке ***Show advanced***.

Параметр	Описание	Значение по умолчанию
Enable Kerberos	Включает аутентификацию Kerberos	False
Authentication on WEB UIs	Включает аутентификацию Kerberos для веб-интерфейсов	False
SPNEGO Signature Value	Секретный ключ, используемый для генерации и проверки подписи в протоколе HTTP-аутентификации SPNEGO (в формате Base64)	—
Kerberos KDC type	Тип KDC. За дополнительной информацией обратитесь к статье Kerberos	MIT
KDC hosts	Один или несколько хостов KDC с запущенными серверами FreeIPA	—
Realm	Kerberos realm для подключения к серверу FreeIPA	—
Domains	Один или несколько доменов, ассоциированных с FreeIPA	—
Kadmin server	Хост, на котором работает `kadmin`	—
Kadmin principal	Имя принципала, используемое для подключения через `kadmin`, например `admin@RU-CENTRAL1.INTERNAL`	—
Kadmin password	Пароль администратора IPA	—
Keytabs directory	Каталог с keytab-файлами	/etc/security/keytabs
Additional realms	Дополнительные Kerberos realms	—
Delay between kinit invocation attempts	Задержка между попытками вызова `kinit`	5
Number of retries for kinit invocation attempts	Количество попыток вызова `kinit`	10
Trusted Active Directory server	Сервер Active Directory для обеспечения односторонних доверительных отношений между realms из MIT Kerberos KDC	—
Trusted Active Directory realm	Active Directory realm для обеспечения односторонних доверительных отношений между realms из MIT Kerberos KDC	—
Custom krb5.conf	Активирует добавление кастомных параметров в файл *krb5.conf*	False
krb5.conf	Дополнительные параметры для записи в файл *krb5.conf*	—
Admin DN	Полное отличительное имя (DistinguishedName) пользователя admin с правами `create/modify/delete/pwdchange` над пользовательскими учётными записями в целевом Organizational Unit	—
LDAP URL	LDAP URL, который состоит из префикса протокола `ldap://` или `ldaps://`, имени хоста или IP-адреса и порта AD-сервера. Например, `ldaps://192.168.4.2:636`	—
Container DN	Отличительное имя (DistinguishedName) контейнера	—
TLS CA certificate Path	Путь к сертификату CA в файловой системе хоста, который предварительно размещён или будет записан из поля ***TLS CA certificate***	—
TLS CA certificate (optional)	Сертификат CA, который записывается на хост по пути из ***TLS CA certificate Path*** во время выполнения действия ***Manage Kerberos***. Сертификат не сохраняется в ADCM	—
Custom ldap.conf	Активирует добавление кастомных параметров в файл *ldap.conf*	False
ldap.conf	Дополнительные параметры для записи в файл *ldap.conf*	—
IpaClient No NTP Autoconf	Отключает автоконфигурирование NTP во время установки клиента IPA	False
IpaClient No DNS Lookup	Отключает поиск DNS для сервера FreeIPA во время установки клиента IPA	True
Custom smb.conf	Использование кастомного файла конфигурации *smb.conf*	False
smb.conf	Кастомный файл конфигурации *smb.conf*	—
Set up Kerberos utils	Активирует возможность устанавливать и удалять клиенты и утилиты Kerberos. Данный параметр учитывается в действиях ***Expand*** и ***Install***	True
Configure Kerberos on hosts	Активирует возможность конфигурировать кластер, включая *krb5.conf*, *ldap.conf*	True
Set up principals and keytabs	Активирует возможность создавать, пересоздавать и удалять принципалов и keytab-файлы. Пароли для принципалов генерируются случайным образом перед созданием keytab-файлов. Данный параметр учитывается в действиях ***Expand*** и ***Install***. Бандл ADCM установит owner и разрешения для keytab-файлов только если этот флаг установлен. В случае отсутствия прав администратора пользователь должен предоставить подготовленный keytab-файл с правильно установленными owner и разрешениями (см. Рекомендации для пользовательских keytab-файлов)	True
Configure services and clients	Активирует возможность обновлять конфигурации сервисов и клиентов	True
Run service checks	Активирует возможность проводить проверку сервисов	True

- Раздел ***Credential encryption*** с настройками провайдера шифрования:
  - ***Credstore password*** — пароль от провайдера шифрования;
  - ***Credstore options*** — способ хранения пароля: пароль не используется (`no password`), хранить пароль в файле (`password-file`) или в переменной среды (`password in the environment`).
- Раздел ***SSL*** с настройками, перечисленными в таблице ниже.

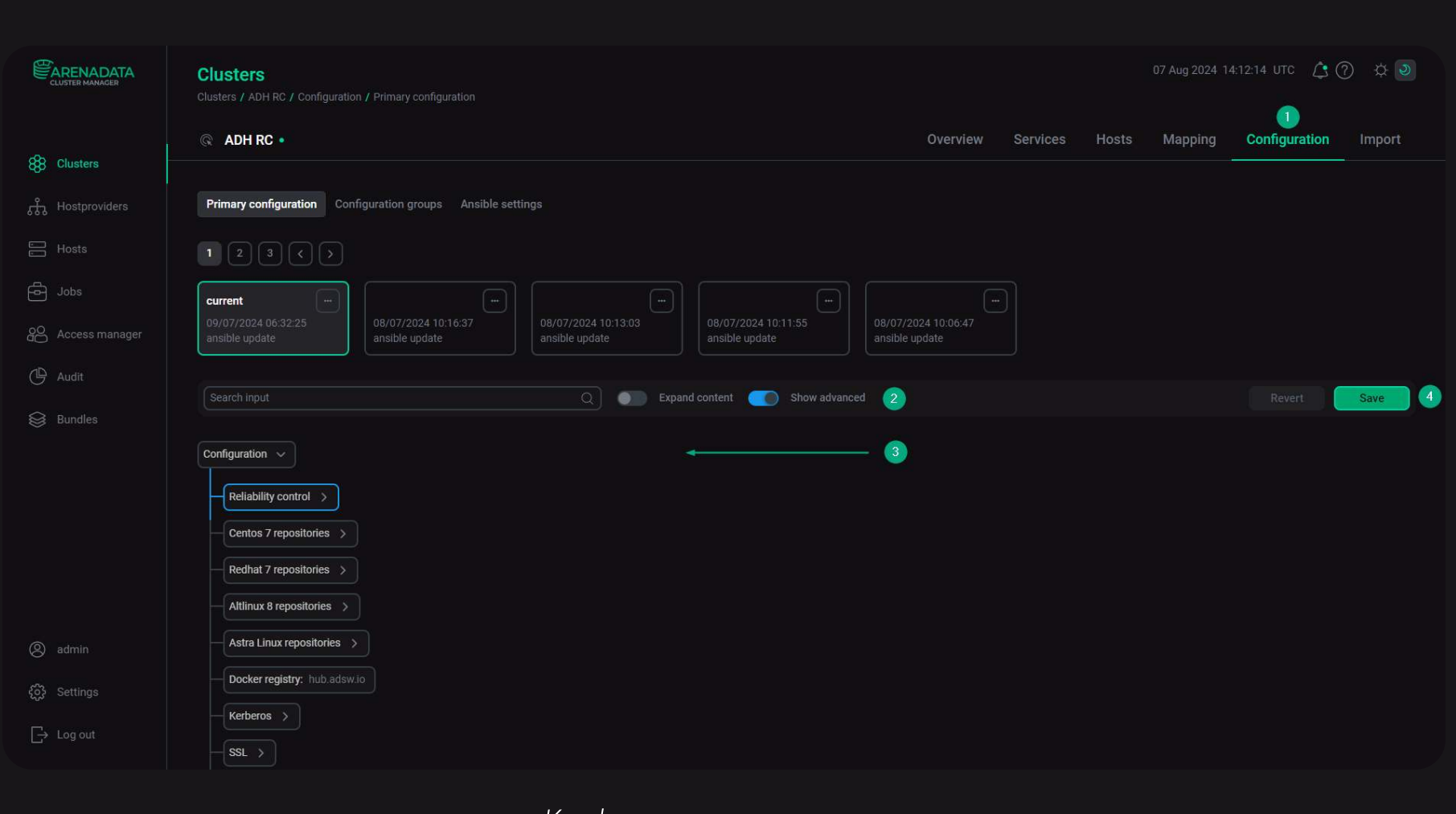
Параметр	Описание	Значение по умолчанию
Enable SSL	Включает использование SSL	false
Keystore path	Путь к keystore-файлу	/etc/ssl/keystore.jks
Keystore password	Пароль для keystore-файла	—
Truststore path	Путь к truststore-файлу	/etc/ssl/truststore.jks
Truststore password	Пароль для truststore-файла	—
TLS Version	Версия протокола TLS. См. Установка версии TLS для кластера ADH	TLSv1.2
Run service checks	Активирует возможность проводить проверку сервисов	true

- Параметр ***java.io.tmpdir***, указывающий переменную среды для управления `java.io.tmpdir` для YARN, HBase, Hive, Spark3.
- Параметр ***ad-runtime-utils*** содержит шаблон файла конфигурации, который задает параметры пути установки runtime detection для заданной среды выполнения (Java, Python и т. д.). Шаблон содержит настройки кластера, а также настройки, специфичные для сервисов, таких как Trino, Flink и Ozone. Этот параметр можно использовать для явного указания версии Java. Подробнее читайте на странице Software Requirements — Java.
  - Раздел `default` настраивает поведение кластера по умолчанию.
  - Раздел `autodetect` определяет алгоритм автоматического определения различных версий Java. Для каждой версии (8, 17, 21, 23, 24) настраивается следующее:
    - `env_var` — переменная окружения, указывающая версию (например, `JAVA8_HOME`, `JAVA17_HOME`);
    - `paths` — список путей для поиска установленной Java.
  - Раздел `services` определяет, для каких сервисов требуются конкретные версии Java.

ad-runtime-utils

```
default:
  runtimes:
    java:
      version: "8"
      env_var: JAVA_HOME

autodetect:
  runtimes:
    java:
      "8":
        env_var: JAVA8_HOME
        paths:
          - /usr/java/jdk1.8
          - /usr/java/jre1.8
          - /usr/lib/jvm/j2sdk1.8-oracle
          - /usr/lib/jvm/j2sdk1.8-oracle/jre
          - /usr/lib/jvm/java-8-oracle
          - /usr/lib/jdk8-latest
          - /usr/lib/jvm/java-1.8.0
          - /usr/lib/jvm/java-1.8.0-oracle
          - /usr/lib/jvm/bellsoft-java8-amd64
          - /usr/lib/jvm/java-arenadata-openjdk-8

      "17":
        env_var: JAVA17_HOME
        paths:
          - /usr/lib/jvm/bellsoft-java17-amd64
          - /usr/lib/jvm/bellsoft-java17
          - /usr/lib/jvm/liberica-jdk-17*
          - /usr/lib/jvm/liberica-jre-17*
          - /usr/lib/jvm/zulu-17*
          - /usr/lib/jvm/temurin-17*
          - /usr/lib/jvm/adoptopenjdk-17*
          - /usr/lib/jvm/jdk-17*
          - /usr/lib/jvm/java-17*
          - /usr/lib/jvm/openjdk-17*
          - /usr/lib/jvm/openjdk-17*
          - /opt/java/bellsoft-java17-amd64
          - /opt/java/temurin-17*
          - /usr/local/java/bellsoft-java17-amd64
          - /usr/lib/jvm/java-arenadata-openjdk-17
          - /usr/lib64/jvm/java-17*

      "21":
        env_var: JAVA21_HOME
        paths:
          - /usr/lib/jvm/java-21*
          - /usr/lib/jvm/jdk-21*
          - /usr/lib/jvm/java-arenadata-openjdk-21

      "23":
        env_var: JAVA23_HOME
        paths:
          - /usr/lib/jvm/bellsoft-java23-amd64
          - /usr/lib/jvm/bellsoft-java23
          - /usr/lib/jvm/liberica-jdk-23*
          - /usr/lib/jvm/liberica-jre-23*
          - /usr/lib/jvm/zulu-23*
          - /usr/lib/jvm/temurin-23*
          - /usr/lib/jvm/adoptopenjdk-23*
          - /usr/lib/jvm/jdk-23*
          - /usr/lib/jvm/java-23-openjdk
          - /usr/lib/jvm/openjdk-23*
          - /opt/java/bellsoft-java23-amd64
          - /opt/java/temurin-23*
          - /usr/local/java/bellsoft-java23-amd64
          - /usr/lib/jvm/java-arenadata-openjdk-23

      "24":
        env_var: JAVA24_HOME
        paths:
          - /usr/lib/jvm/bellsoft-java24-amd64
          - /usr/lib/jvm/bellsoft-java24
          - /usr/lib/jvm/liberica-jdk-24*
          - /usr/lib/jvm/liberica-jre-24*
          - /usr/lib/jvm/zulu-24*
          - /usr/lib/jvm/temurin-24*
          - /usr/lib/jvm/adoptopenjdk-24*
          - /usr/lib/jvm/jdk-24*
          - /usr/lib/jvm/java-24-openjdk
          - /usr/lib/jvm/openjdk-24*
          - /opt/java/bellsoft-java24-amd64
          - /opt/java/temurin-24*
          - /usr/local/java/bellsoft-java24-amd64
          - /usr/lib/jvm/java-arenadata-openjdk-24

services:
  TRINO:
      path: /etc/trino/conf/trino-java.yaml
  FLINK2:
      path: /etc/flink2/conf/flink2-java.yaml
  OZONE:
      path: /etc/ozone/conf/ozone-java.yaml
  SPARK4:
      path: /etc/spark4/conf/spark4-java.yaml
```

- Параметр ***Precheck packages***, активирующий возможность проверять наличие пакетов в репозитории перед установкой. Значение по умолчанию — `false`.
- ***Cluster UUID*** — отображает универсальный уникальный идентификатор кластера (universally unique identifier).
- Параметр ***Ranger plugins version***, позволяющий переопределить номер версии плагинов Ranger.

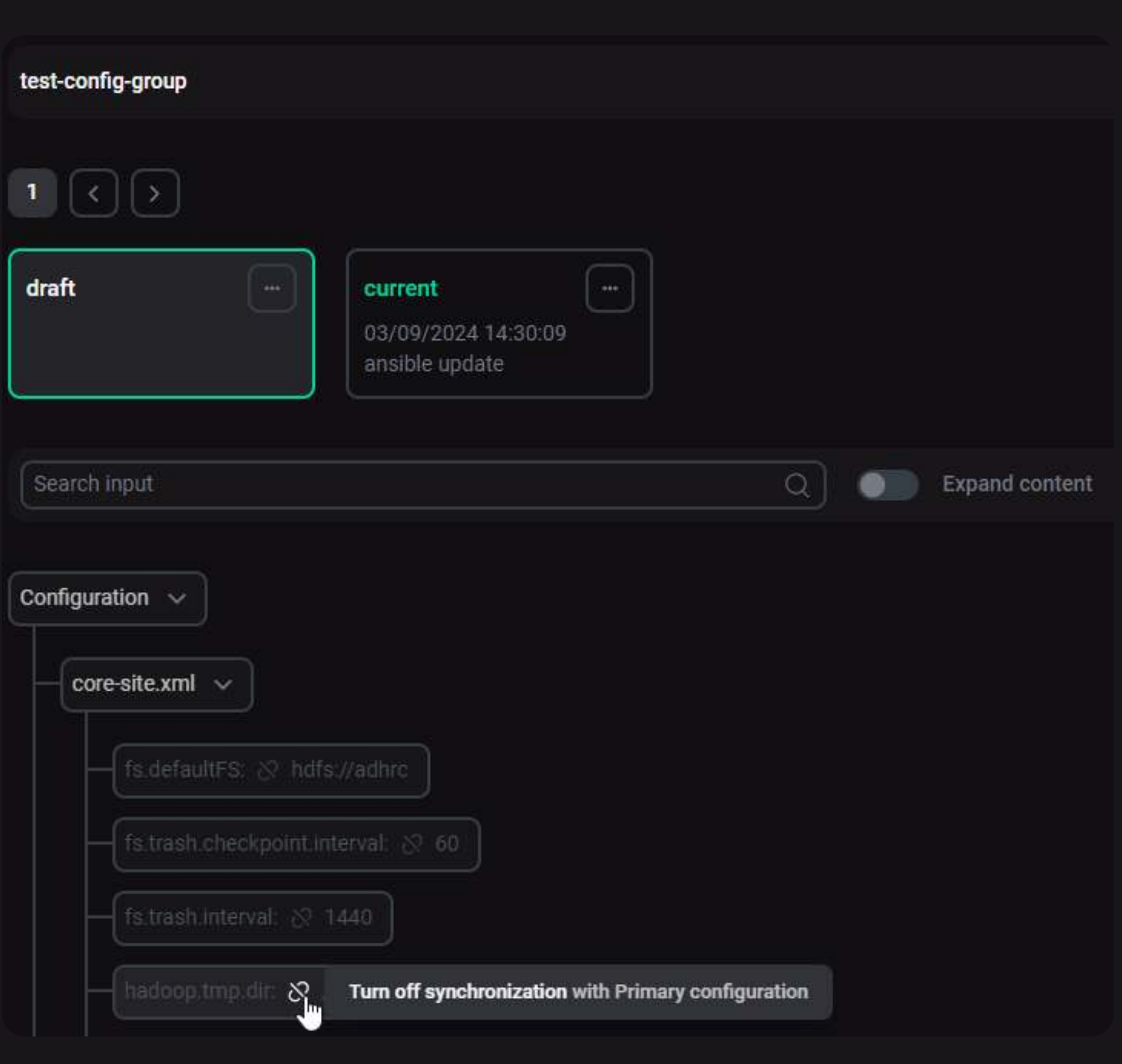


Конфигурирование кластера

Configuration groups

Вкладка ***Configuration*** → ***Configuration groups*** предназначена для настройки конфиг-групп кластера.

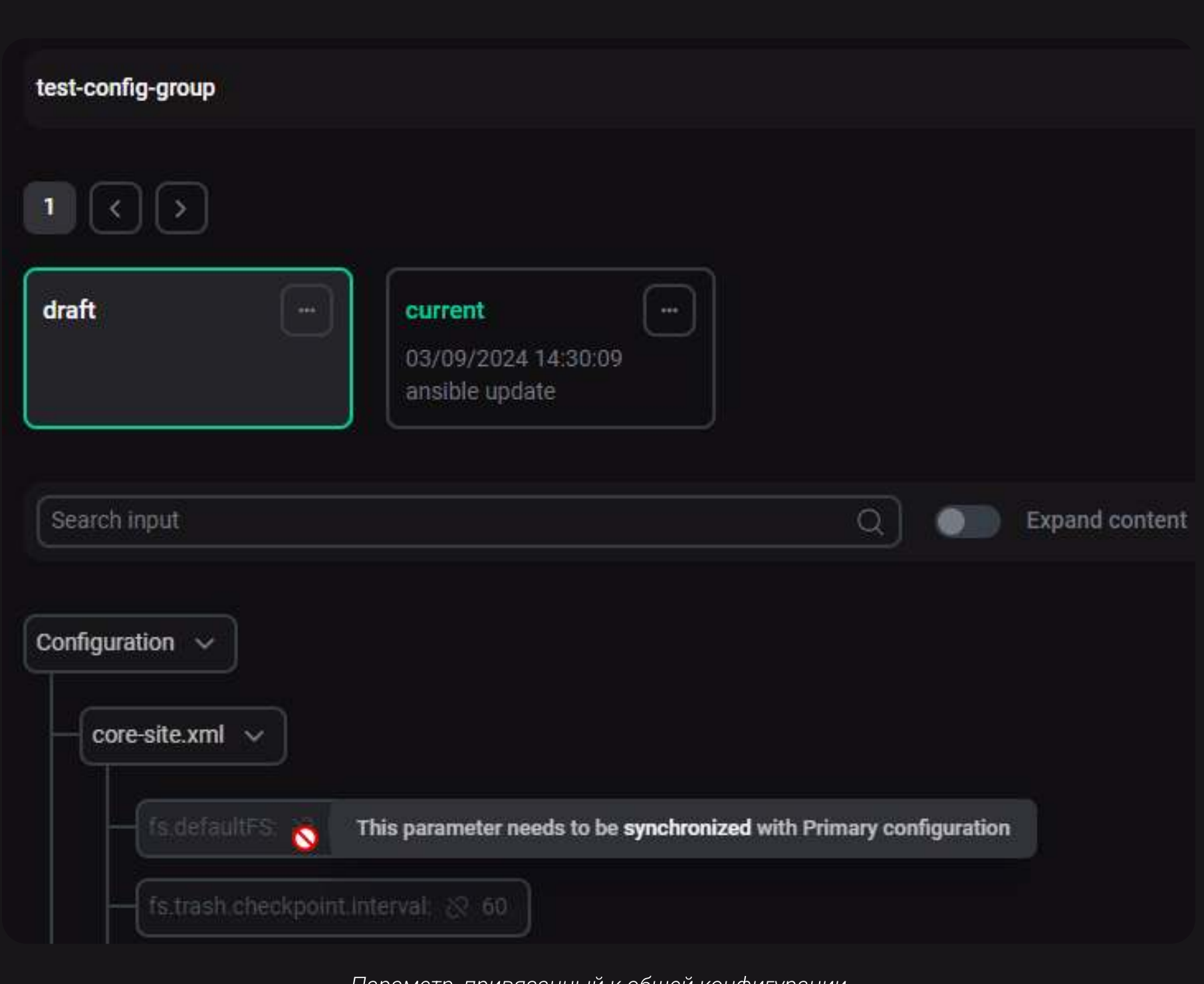
Чтобы настроить параметры в конфиг-группе, выберите параметры, которые вы хотите настроить иначе, нажмите иконку ***Turn off synchronization***, отредактируйте их и нажмите ***Save***.



Turn off synchronization with Primary configuration

Вкладка Configuration на странице конфиг-группы

Некоторые параметры нельзя изменить в рамках конфиг-группы, поскольку они всегда должны быть синхронизированы с общей конфигурацией.



This parameter needs to be synchronized with Primary configuration

Параметр, привязанный к общей конфигурации

Ansible settings

Вкладка ***Configuration*** → ***Ansible settings*** предназначена для ввода конфигурационных настроек Ansible на уровне кластера. Вкладка доступна начиная с версии ADCM 2.2.0.

Параметры Ansible

Имя	Описание	По умолчанию
forks	Число параллельных процессов, генерируемых при коммуникации с удаленными хостами	5

Импорт настроек ET

Евгения Кузина, Дарья Барышева

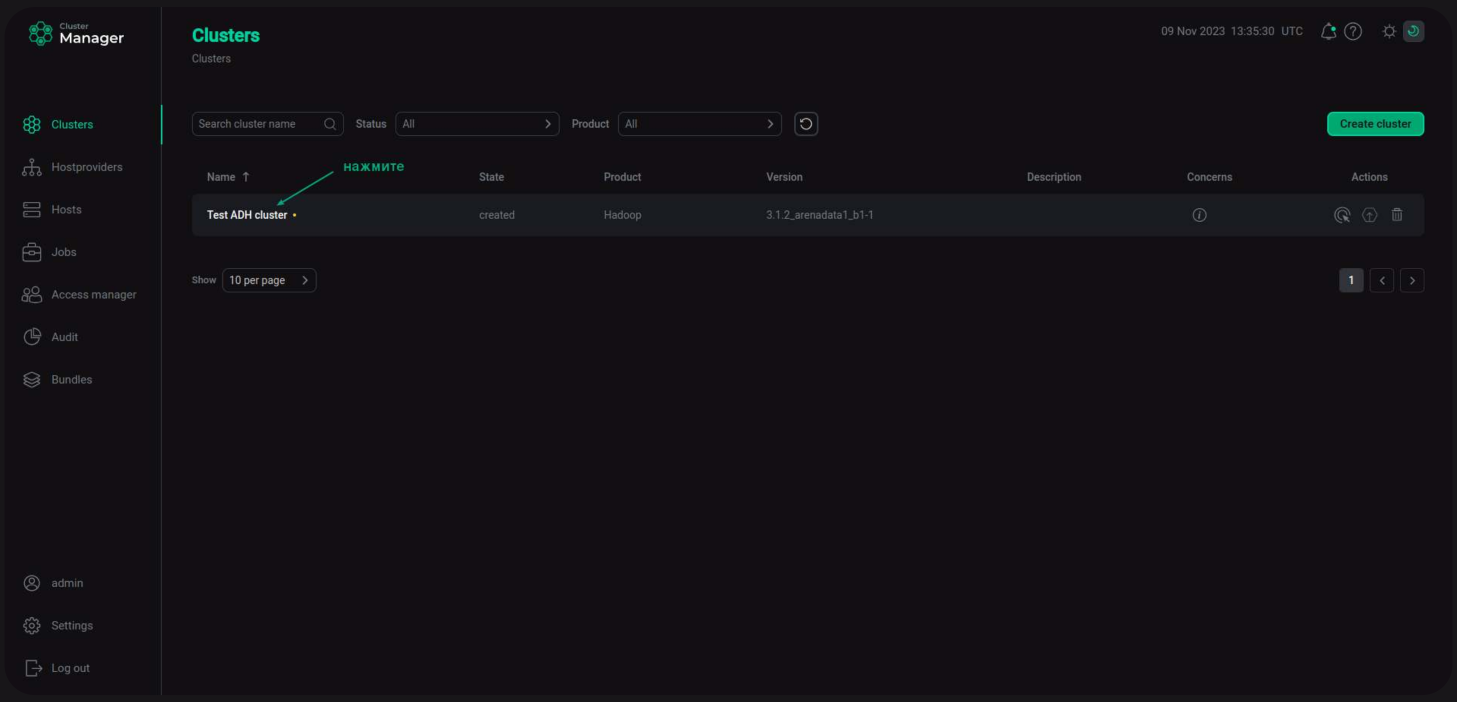
Назад к содержанию

ВАЖНО

Этот шаг необходим только в случае offline-установки.

Для установки кластера в режиме offline требуется интегрировать его с ранее созданным кластером Enterprise Tools (ET):

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
09 Nov 2023 13:35:30 UTC
Search cluster name
Status
All
Product
All
Create cluster
нажмите
Show
10 per page
1

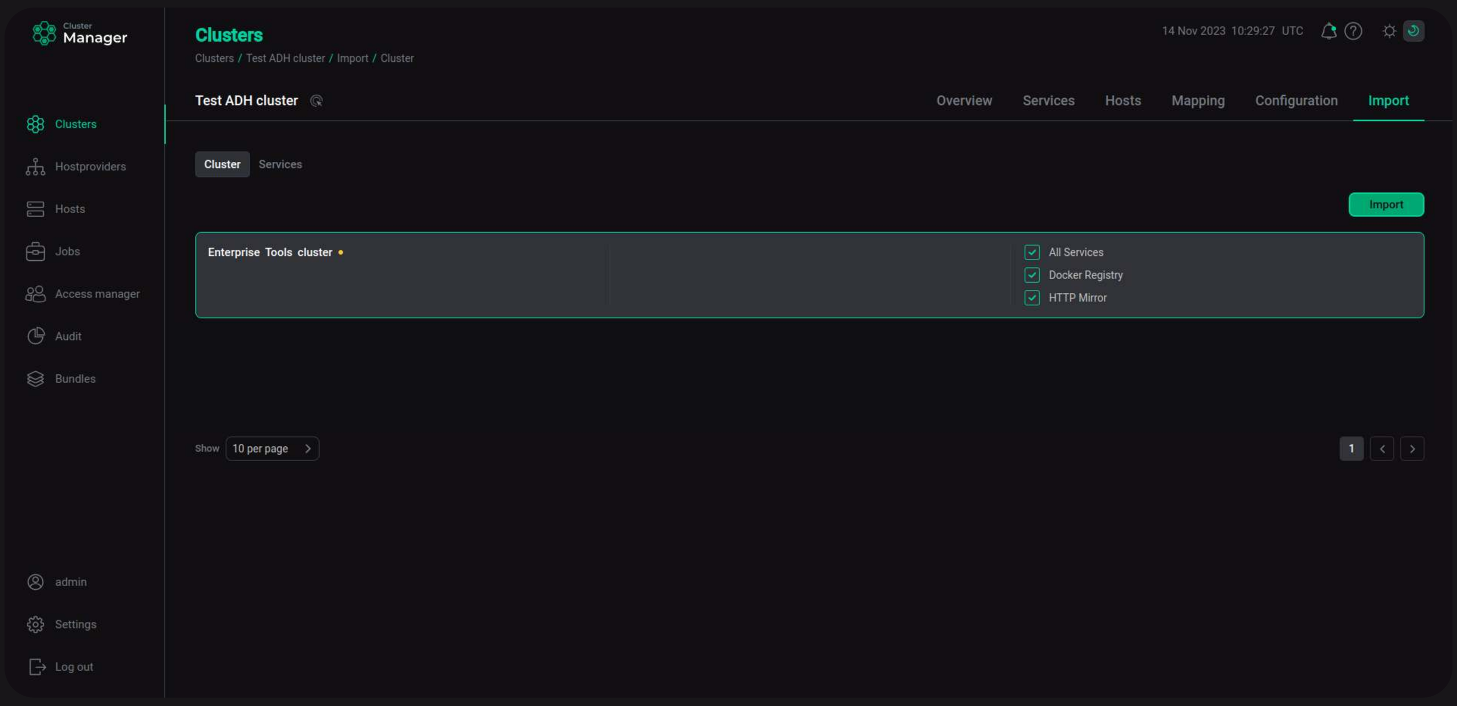
Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

Выбор кластера

2. Откройте вкладку ***Import*** на странице кластера.

3. Выберите сервисы ***Docker Registry*** и ***HTTP Mirror*** предварительно созданного кластера ET.

4. Нажмите ***Import***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / Test ADH cluster / Import / Cluster
14 Nov 2023 10:29:27 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Cluster
Services
Import
Show
10 per page
1

Enterprise Tools cluster

☑ All Services
☑ Docker Registry
☑ HTTP Mirror

Интеграция с кластером ET

ПРИМЕЧАНИЕ

После успешного импорта данные, хранимые локально в кластере Enterprise Tools, могут быть использованы на этапе установки продуктового кластера. Таким образом, их загрузка через Интернет не требуется.

Установка кластера

Евгения Кузина, Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Запуск установки
  Установка всего кластера
  Установка отдельных сервисов
Шаг 2. Просмотр результатов установки
Шаг 3. Проверка статуса кластера после установки

Шаг 1. Запуск установки

Существует два способа установки:

1. **Установка всего кластера**. В этом методе все сервисы кластера устанавливаются автоматически друг за другом после применения действия ***Install*** к кластеру.
2. **Установка отдельных сервисов**. В этом методе каждый сервис устанавливается вручную путем применения к нему одного или нескольких действий. Этот способ также можно использовать для добавления новых сервисов в уже развернутый кластер.

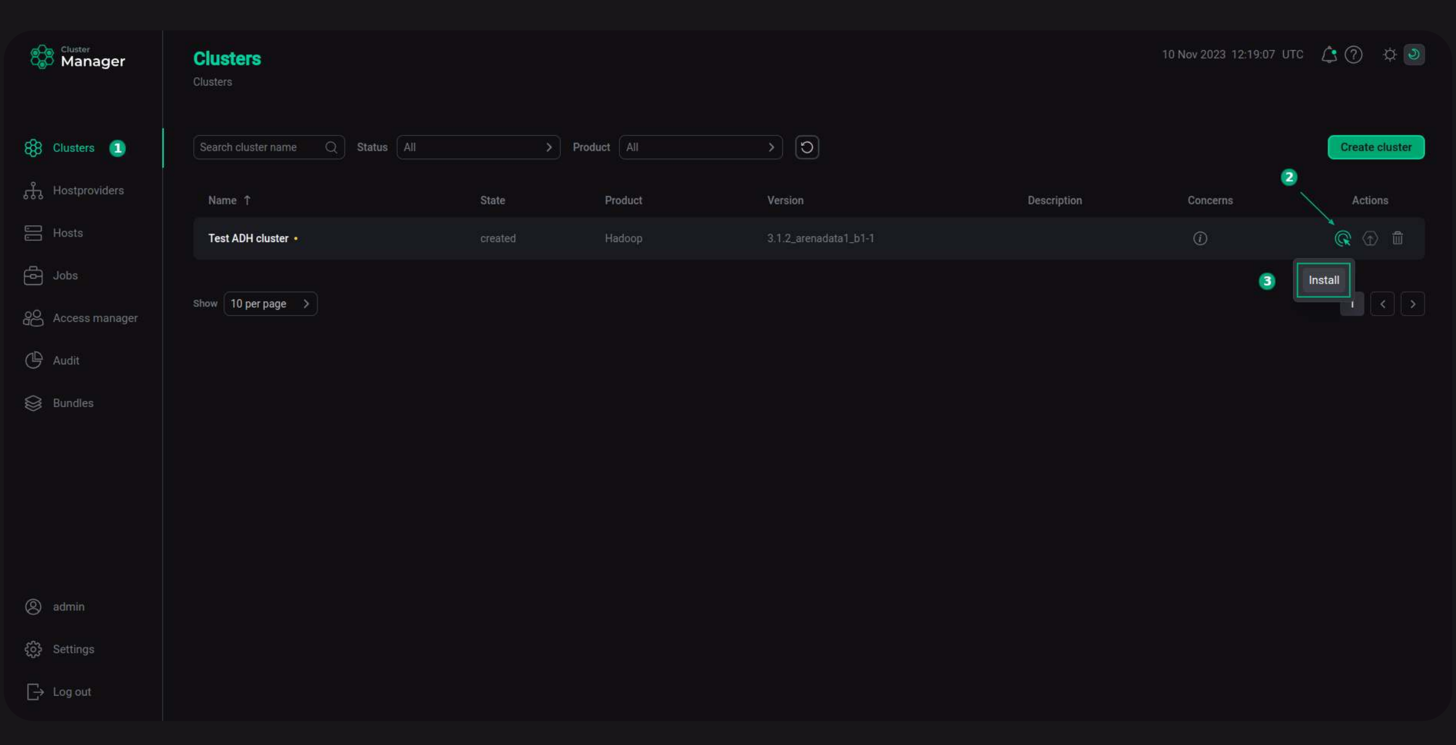
В обоих методах каждый сервис запускается автоматически после установки.

> **ВАЖНО**
>
> Установка должна производиться на чистые хосты без предустановленного подобного ПО.

Установка всего кластера

Чтобы установить все сервисы кластера за одну операцию, выполните шаги:

1. Примените действие ***Install*** к кластеру, выбранному на странице ***Clusters***, нажав на иконку в столбце ***Actions***.

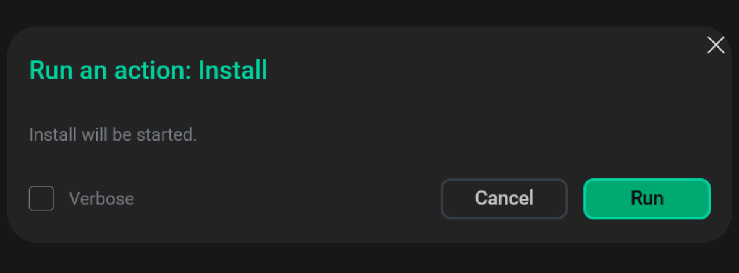


Cluster Manager
Clusters
Clusters
10 Nov 2023 12:19:07 UTC
Clusters 1
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status All
Product All
Create cluster
2
3
Install
Show 10 per page

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	created	Hadoop	3.1.2_arenadata1_b1-1			

Переход к установке всего кластера

2. Подтвердите действие в открывшемся окне.



Run an action: Install

Install will be started.

☐ Verbose

Cancel Run

Подтверждение действия

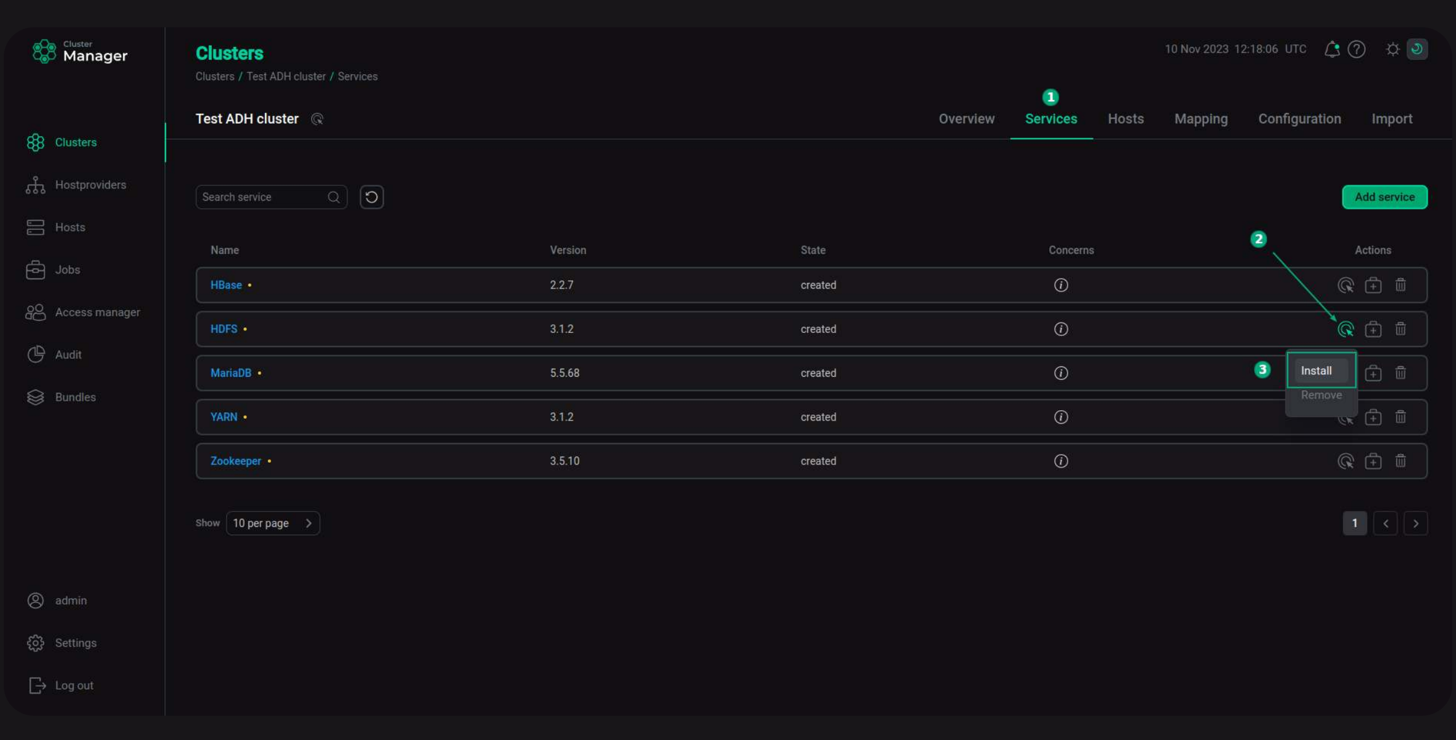
3. Дождитесь завершения установки.

Установка отдельных сервисов

Чтобы установить один или несколько сервисов вручную, необходимо предварительно добавить эти сервисы в кластер и затем применить к ним действия, описанные ниже.

> **ВНИМАНИЕ**
>
> Обратите внимание, что сервисы устанавливаются по очереди друг за другом. Не пытайтесь установить следующий сервис до окончания установки предыдущего.

1. Откройте вкладку ***Services*** на странице кластера. Для сервиса, который должен быть установлен, нажмите на иконку в столбце ***Actions*** и выберите действие ***Install*** из выпадающего списка.

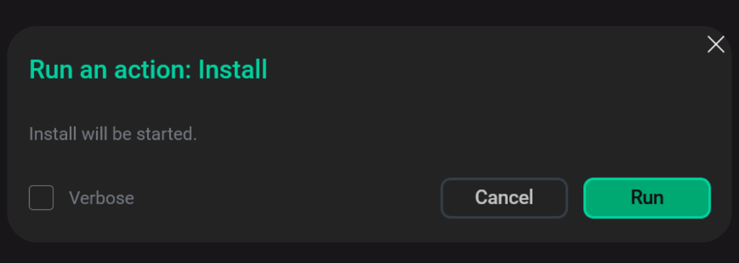


Cluster Manager
Clusters
Clusters / Test ADH cluster / Services
10 Nov 2023 12:18:06 UTC
Test ADH cluster
Overview
1
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
Add service
2
3
Install
Remove
Show 10 per page
1

Name	Version	State	Concerns	Actions
HBase	2.2.7	created		
HDFS	3.1.2	created		
MariaDB	5.5.68	created		
YARN	3.1.2	created		
Zookeeper	3.5.10	created		

Переход к установке отдельных сервисов

2. Подтвердите действие в открывшемся окне.



Run an action: Install

Install will be started.

☐ Verbose

Cancel Run

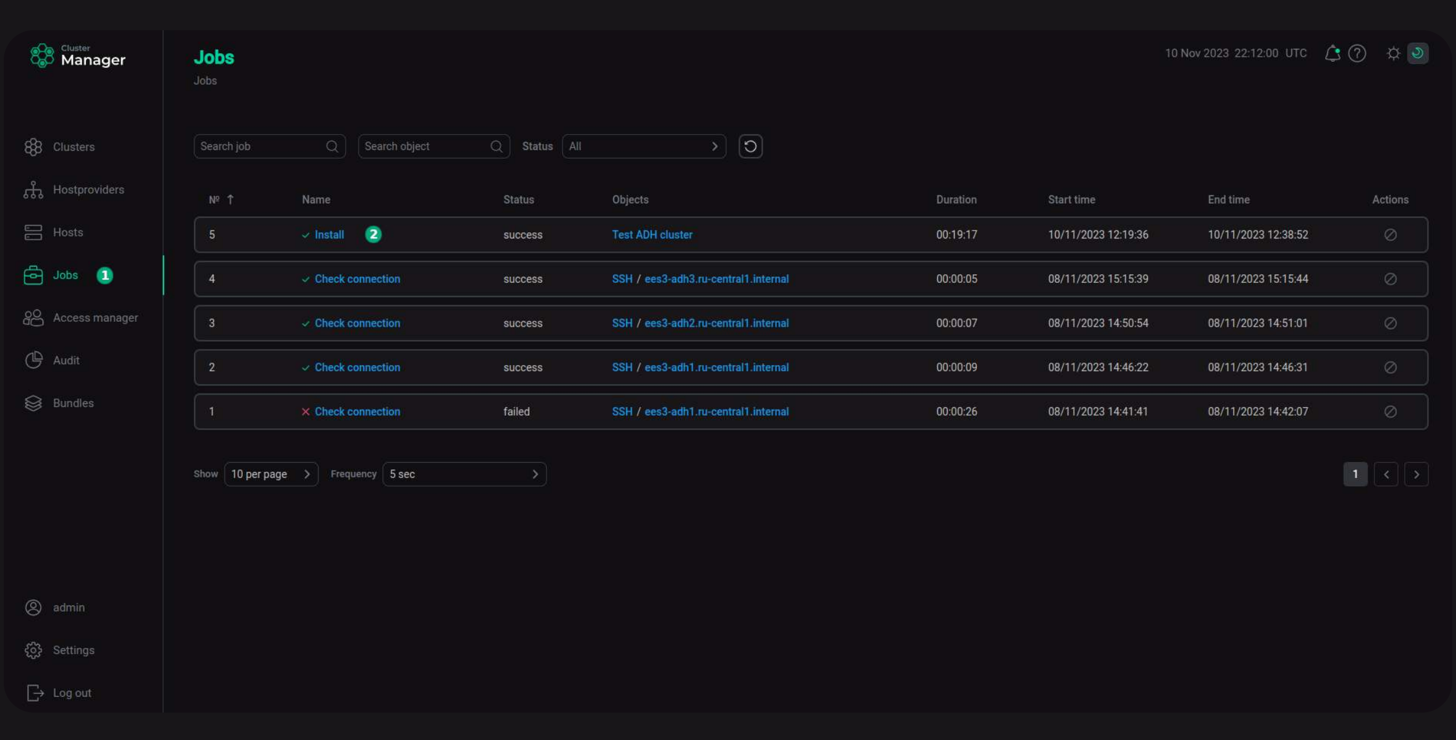
Подтверждение действия

3. Дождитесь завершения установки и убедитесь, что статус сервиса изменился с `created` на `installed`.
4. Повторите предыдущие шаги для всех сервисов, которые необходимо установить.

> **ВАЖНО**
>
> При установке сервисов вручную важно соблюдать следующий порядок:
>
> 1. Zookeeper
> 2. HDFS
> 3. YARN
> 4. HBase
> 5. Hive
> 6. Spark3
> 7. Monitoring

Шаг 2. Просмотр результатов установки

ADCM запускает отдельную задачу (job) для установки, как и для любого другого процесса. Для получения информации о статусе задач предназначена страница ***Jobs***.

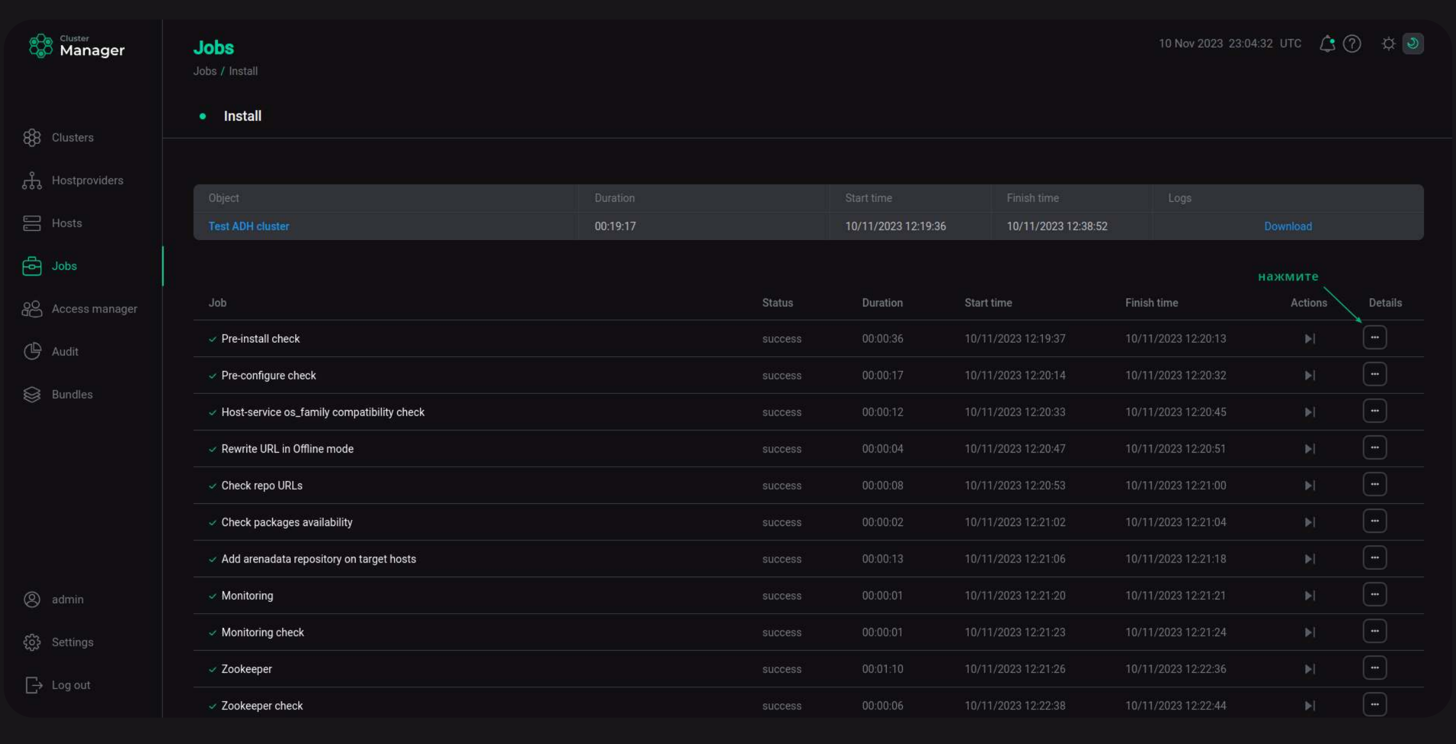


Cluster Manager
Jobs
Jobs
10 Nov 2023 22:12:00 UTC
Clusters
Hostproviders
Hosts
Jobs 1
Access manager
Audit
Bundles
admin
Settings
Log out
Search job
Search object
Status All
2
Show 10 per page
Frequency 5 sec
1

№	Name	Status	Objects	Duration	Start time	End time	Actions
5	✓ Install	success	Test ADH cluster	00:19:17	10/11/2023 12:19:36	10/11/2023 12:38:52	
4	✓ Check connection	success	SSH / ees3-adh3.ru-central1.internal	00:00:05	08/11/2023 15:15:39	08/11/2023 15:15:44	
3	✓ Check connection	success	SSH / ees3-adh2.ru-central1.internal	00:00:07	08/11/2023 14:50:54	08/11/2023 14:51:01	
2	✓ Check connection	success	SSH / ees3-adh1.ru-central1.internal	00:00:09	08/11/2023 14:46:22	08/11/2023 14:46:31	
1	✕ Check connection	failed	SSH / ees3-adh1.ru-central1.internal	00:00:26	08/11/2023 14:41:41	08/11/2023 14:42:07	

Страница Jobs

Если установка завершается успешно, соответствующая ей задача на странице ***Jobs*** переходит из статуса `running` в `success`; в противном случае — в статус `failed`. Чтобы получить более детальную информацию о возникших ошибках, можно перейти на страницу задачу, кликнув по ней в списке задач на странице ***Jobs***. Откроется окно с шагами выполнения задачи. Кликните многоточие в столбце ***Details***, чтобы получить дополнительную информацию об определенном шаге.



Cluster Manager
Jobs
Jobs / Install
10 Nov 2023 23:04:32 UTC
Install
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
нажмите

Object	Duration	Start time	Finish time	Logs
Test ADH cluster	00:19:17	10/11/2023 12:19:36	10/11/2023 12:38:52	Download

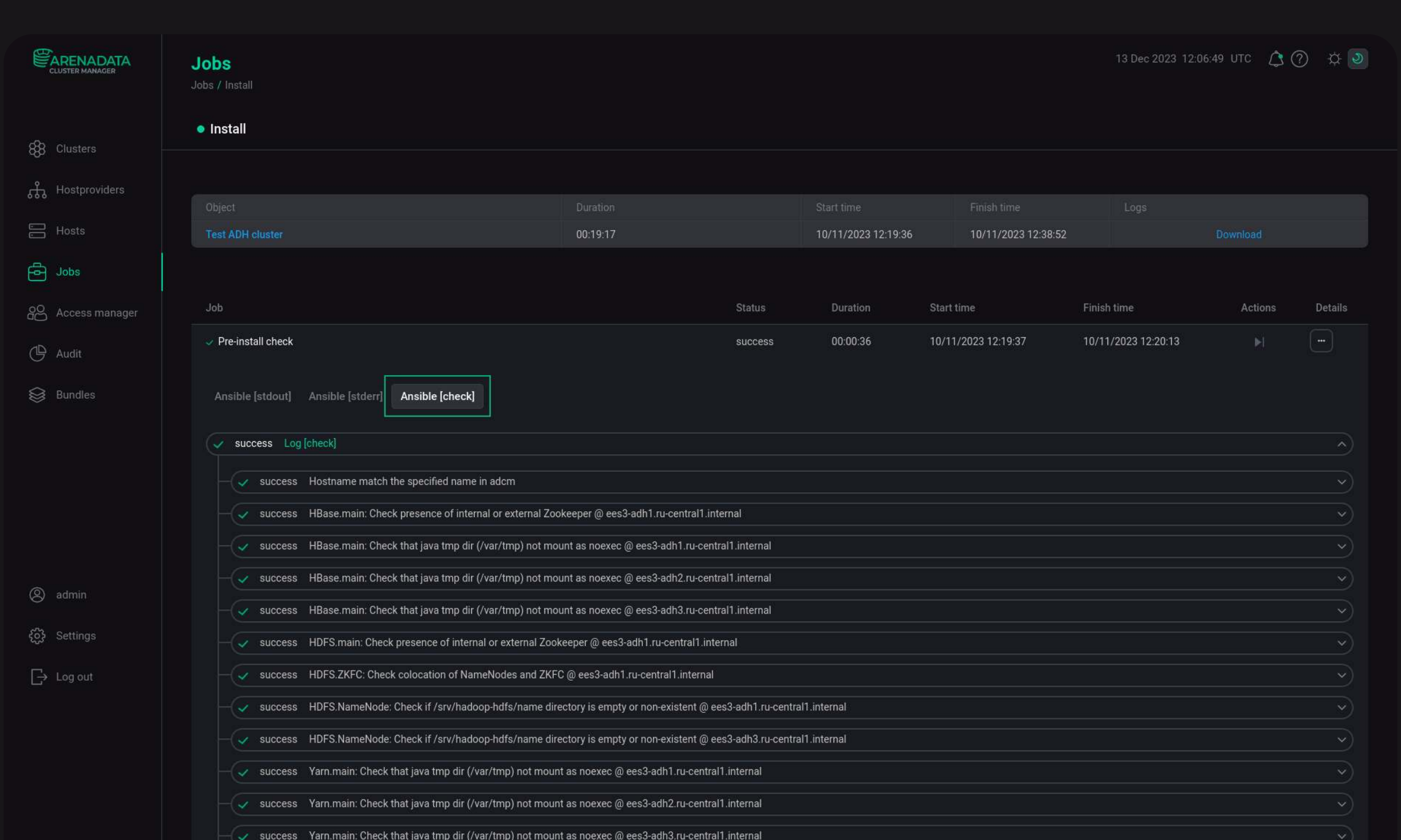
Job	Status	Duration	Start time	Finish time	Actions	Details
✓ Pre-install check	success	00:00:36	10/11/2023 12:19:37	10/11/2023 12:20:13		
✓ Pre-configure check	success	00:00:17	10/11/2023 12:20:14	10/11/2023 12:20:32		
✓ Host-service os_family compatibility check	success	00:00:12	10/11/2023 12:20:33	10/11/2023 12:20:45		
✓ Rewrite URL in Offline mode	success	00:00:04	10/11/2023 12:20:47	10/11/2023 12:20:51		
✓ Check repo URLs	success	00:00:08	10/11/2023 12:20:53	10/11/2023 12:21:00		
✓ Check packages availability	success	00:00:02	10/11/2023 12:21:02	10/11/2023 12:21:04		
✓ Add arenadata repository on target hosts	success	00:00:13	10/11/2023 12:21:06	10/11/2023 12:21:18		
✓ Monitoring	success	00:00:01	10/11/2023 12:21:20	10/11/2023 12:21:21		
✓ Monitoring check	success	00:00:01	10/11/2023 12:21:23	10/11/2023 12:21:24		
✓ Zookeeper	success	00:01:10	10/11/2023 12:21:26	10/11/2023 12:22:36		
✓ Zookeeper check	success	00:00:06	10/11/2023 12:22:38	10/11/2023 12:22:44		

Переход на страницу с подробностями выполнения

Страница задачи содержит два основных раздела: ***ansible [stdout]*** и ***ansible [stderr]***. Это технические логи, содержащие информацию из стандартных I/O-потоков stdout/stderr.

В дополнение к ним может выводиться третий опциональный раздел ***ansible [check]***. Он содержит краткие описания наиболее часто возникающих ошибок. По сравнению с предыдущими разделами, эти описания, как правило, более простые и конкретные.

При возникновении ошибок в процессе установки рекомендуется изучить логи всех трех разделов.



ARENADATA CLUSTER MANAGER
Jobs
Jobs / Install
13 Dec 2023 12:06:49 UTC
Install
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Ansible [stdout]
Ansible [stderr]
Ansible [check]
Download

Object	Duration	Start time	Finish time	Logs
Test ADH cluster	00:19:17	10/11/2023 12:19:36	10/11/2023 12:38:52	Download

Job	Status	Duration	Start time	Finish time	Actions	Details
✓ Pre-install check	success	00:00:36	10/11/2023 12:19:37	10/11/2023 12:20:13		

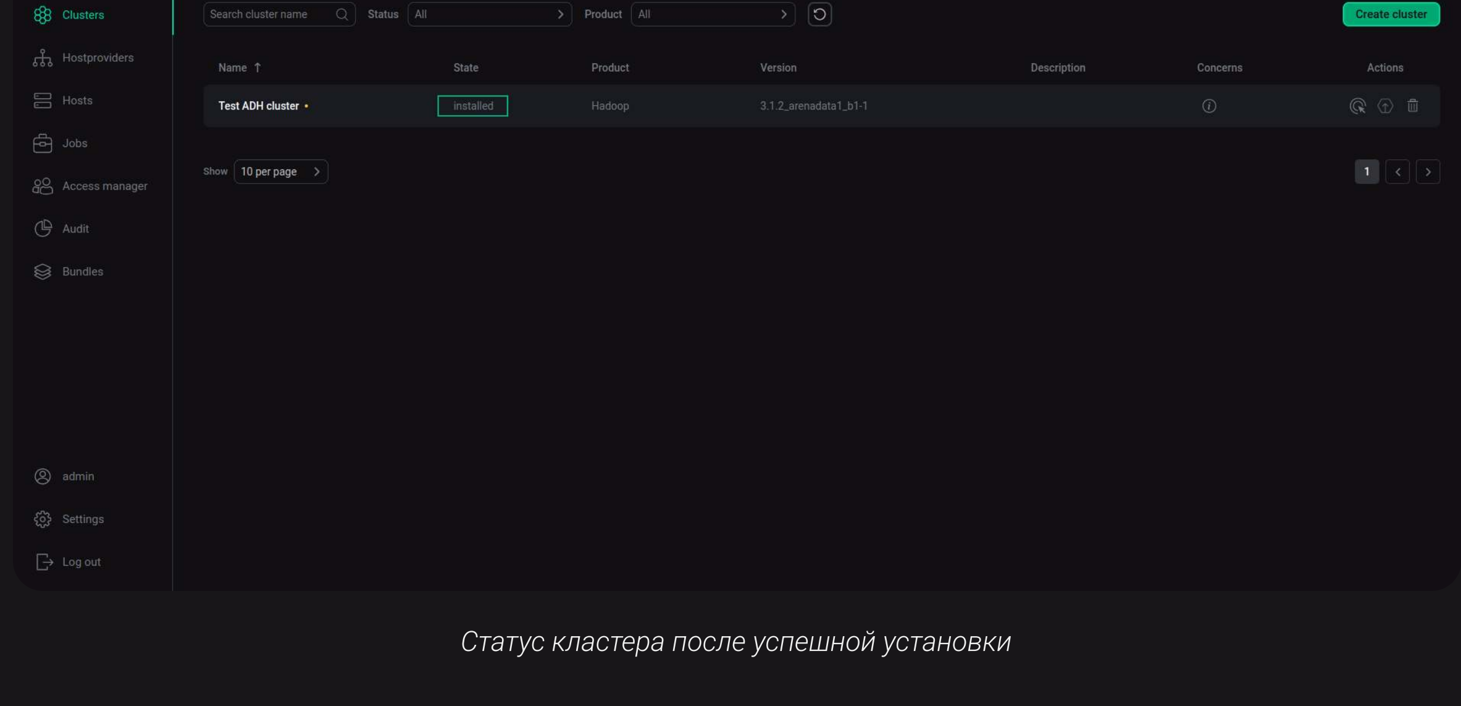
```
success  Log [check]
  success  Hostname match the specified name in adcm
  success  HBase.main: Check presence of internal or external Zookeeper @ ees3-adh1.ru-central1.internal
  success  HBase.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh1.ru-central1.internal
  success  HBase.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh2.ru-central1.internal
  success  HBase.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh3.ru-central1.internal
  success  HDFS.main: Check presence of internal or external Zookeeper @ ees3-adh1.ru-central1.internal
  success  HDFS.ZKFC: Check colocation of NameNodes and ZKFC @ ees3-adh1.ru-central1.internal
  success  HDFS.NameNode: Check if /srv/hadoop-hdfs/name directory is empty or non-existent @ ees3-adh1.ru-central1.internal
  success  HDFS.NameNode: Check if /srv/hadoop-hdfs/name directory is empty or non-existent @ ees3-adh3.ru-central1.internal
  success  Yarn.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh1.ru-central1.internal
  success  Yarn.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh2.ru-central1.internal
  success  Yarn.main: Check that java tmp dir (/var/tmp) not mount as noexec @ ees3-adh3.ru-central1.internal
  success  YARN.main: Enable HA ResourceManager @ ees3-adh2.ru-central1.internal
```

Подробности выполнения задачи

Шаг 3. Проверка статуса кластера после установки

В результате успешной установки происходит обновление статуса кластера и его сервисов:

- Статус кластера, отображаемый в столбце ***State*** на странице ***Clusters***, меняет свое значение с `created` на `installed`.

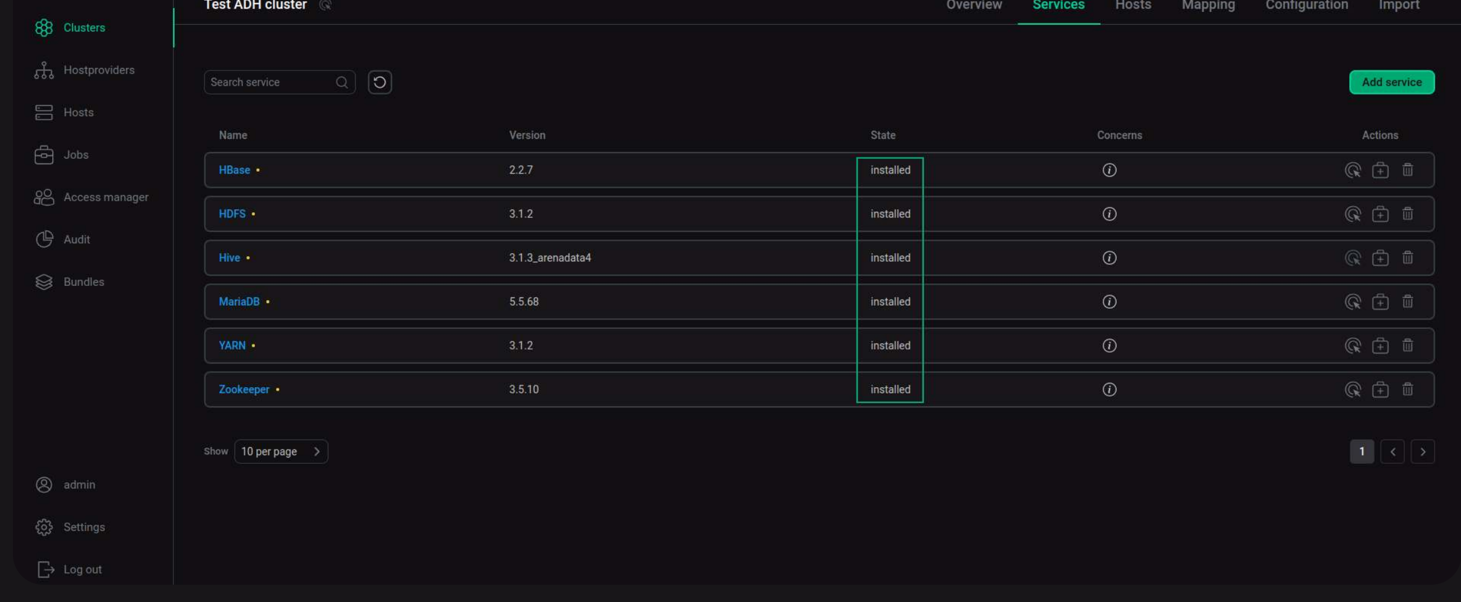


Cluster Manager
Clusters
Clusters
10 Nov 2023 23:29:12 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status All
Product All
Create cluster
Show 10 per page
1

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1			

Статус кластера после успешной установки

- Статус сервисов, отображаемый на вкладке ***Services*** страницы кластера, меняет свое значение с `created` на `installed`.



Cluster Manager
Clusters
Clusters / Test ADH cluster / Services
10 Nov 2023 23:38:28 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
Add service
Show 10 per page
1

Name	Version	State	Concerns	Actions
HBase	2.2.7	installed		
HDFS	3.1.2	installed		
Hive	3.1.3_arenadata4	installed		
MariaDB	5.5.68	installed		
YARN	3.1.2	installed		
Zookeeper	3.5.10	installed		

Статус сервисов кластера после успешной установки

Установка мониторинга

Дарья Барышева

Мониторинг — это опциональный, но рекомендуемый компонент ADH, который можно настроить, используя один из следующих подходов:

- добавить в кластер ADH сервис Monitoring (доступен, начиная с версии ADH 4.0.0);
- добавить сервисы мониторинга в кластер Enterprise Tools и интегрировать его с кластером ADH (поддержка этого способа установки мониторинга будет прекращена в будущих релизах ADH).

Сервис мониторинга

Елена Дворядкина, Елена Костюченко

Содержание

Назад к содержанию

Обзор

Шаг 1. Добавление сервиса

Шаг 2. Добавление компонентов

Шаг 3. Настройка сервиса

Шаг 4. Установка сервиса

Шаг 5. Проверка результатов

- Метрики в формате Prometheus
- Веб-интерфейс Prometheus
- Веб-интерфейс Grafana

Сервис Monitoring разворачивает в ADH свой сервер Prometheus, который реализует сбор и хранение метрик мониторинга кластера ADH, а также поддерживает возможность использования веб-интерфейса Grafana для визуализации и анализа информации. В данной статье описываются шаги, необходимые для подключения этого сервиса.

Обзор

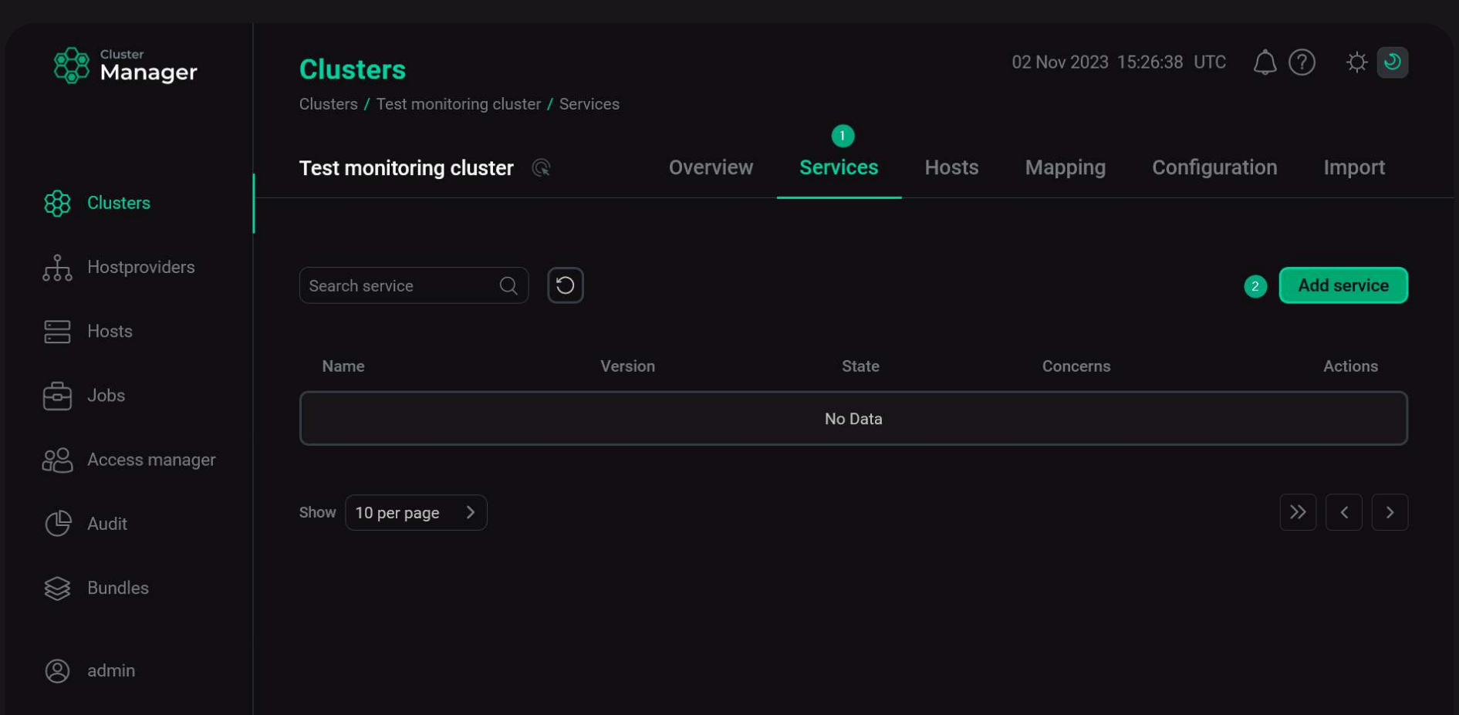
При добавлении сервиса Monitoring в ADH на все или выборочные хосты кластера можно установить Node Exporter — агент мониторинга, считывающий с хоста системные метрики, которые будет собирать Prometheus. Prometheus также может собирать метрики сервисов ADH. Эти метрики будут доступны в формате Prometheus на портах и конечных точках (endpoints), указанных в параметрах сервисных конфигураций. Просматривать и анализировать данные, которые собирает сервис мониторинга, можно также в веб-интерфейсе Prometheus или Grafana.

ПРИМЕЧАНИЕ

- Если у вас уже настроена совместимая с Prometheus система мониторинга (например, свой сервер Prometheus или VictoriaMetrics), можно использовать ее, чтобы собирать метрики ADH. Для этого в своей системе мониторинга укажите настройки доступа к метрикам ADH, используя параметры установленного в ADH Prometheus, которые конфигурируются на странице сервиса Monitoring в интерфейсе ADCM.
- Вы также можете использовать механизм Federation, чтобы перенести все метрики из сервера Prometheus, развернутого в ADH, в свой Prometheus.

Шаг 1. Добавление сервиса

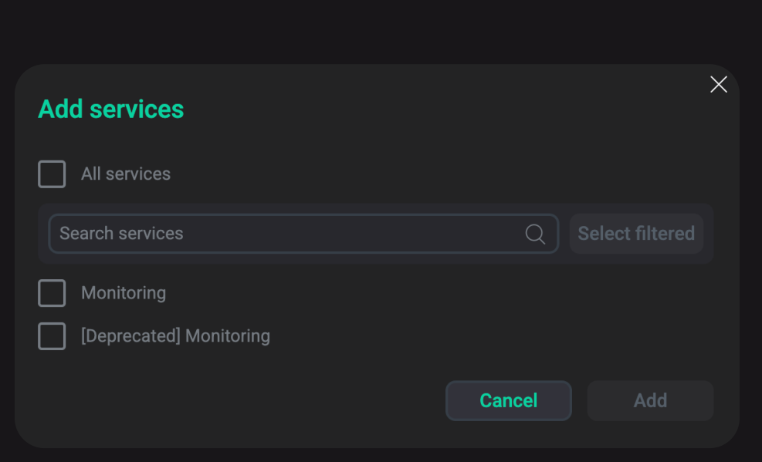
1. В интерфейсе ADCM откройте страницу ***Clusters*** и кликните по имени кластера ADH. Затем на открывшейся странице кластера перейдите на вкладку ***Services*** и нажмите ***Add services***.



Cluster Manager
Clusters
Clusters / Test monitoring cluster / Services
02 Nov 2023 15:26:38 UTC
Test monitoring cluster
Overview
1
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
2
Add service
Name
Version
State
Concerns
Actions
No Data
Show
10 per page

Переход к добавлению сервисов

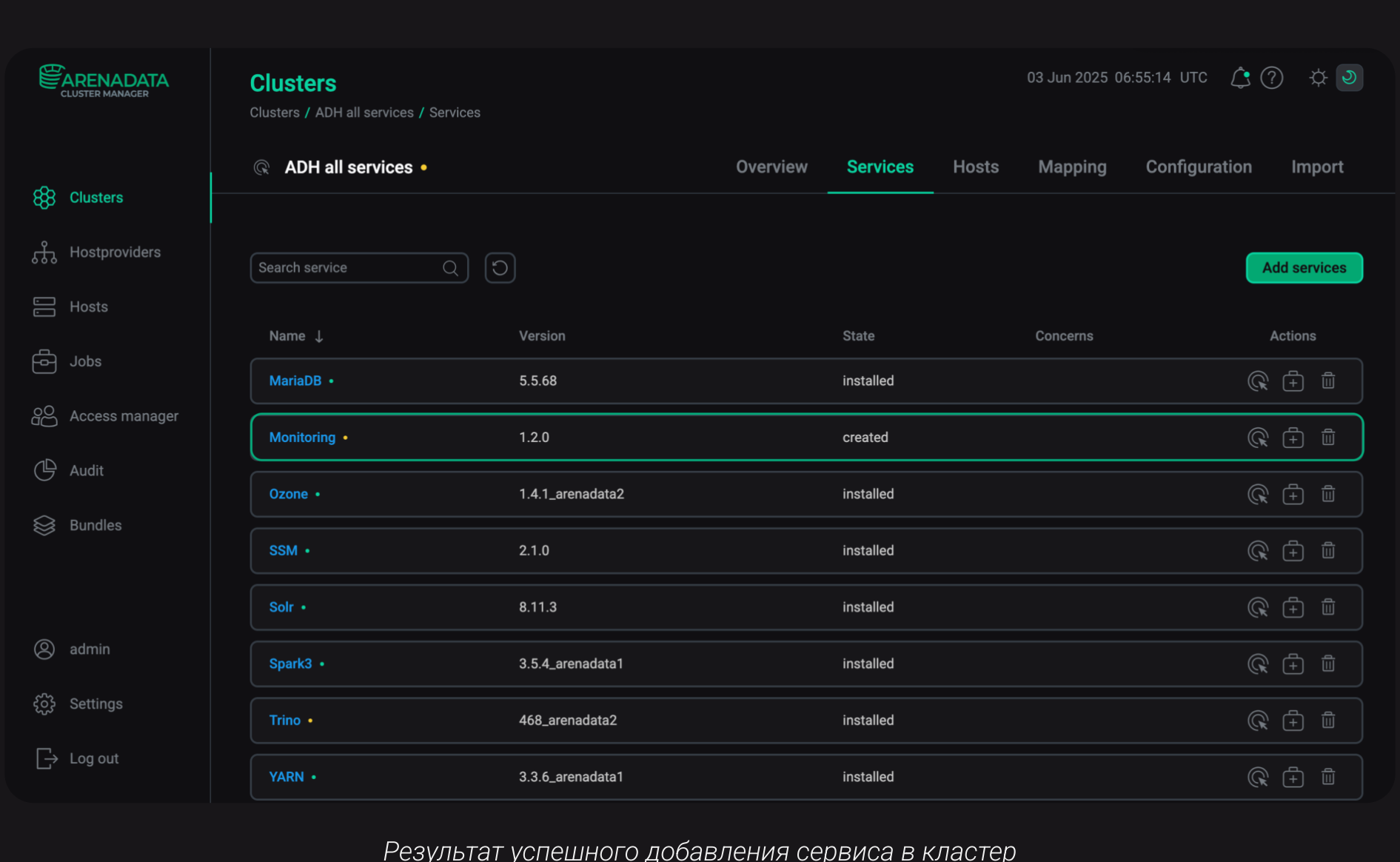
2. В открывшемся окне выберите сервис ***Monitoring*** и нажмите ***Add***.



Add services
All services
Search services
Select filtered
Monitoring
[Deprecated] Monitoring
Cancel
Add

Выбор сервиса

В результате добавленный сервис отобразится на вкладке ***Services***.



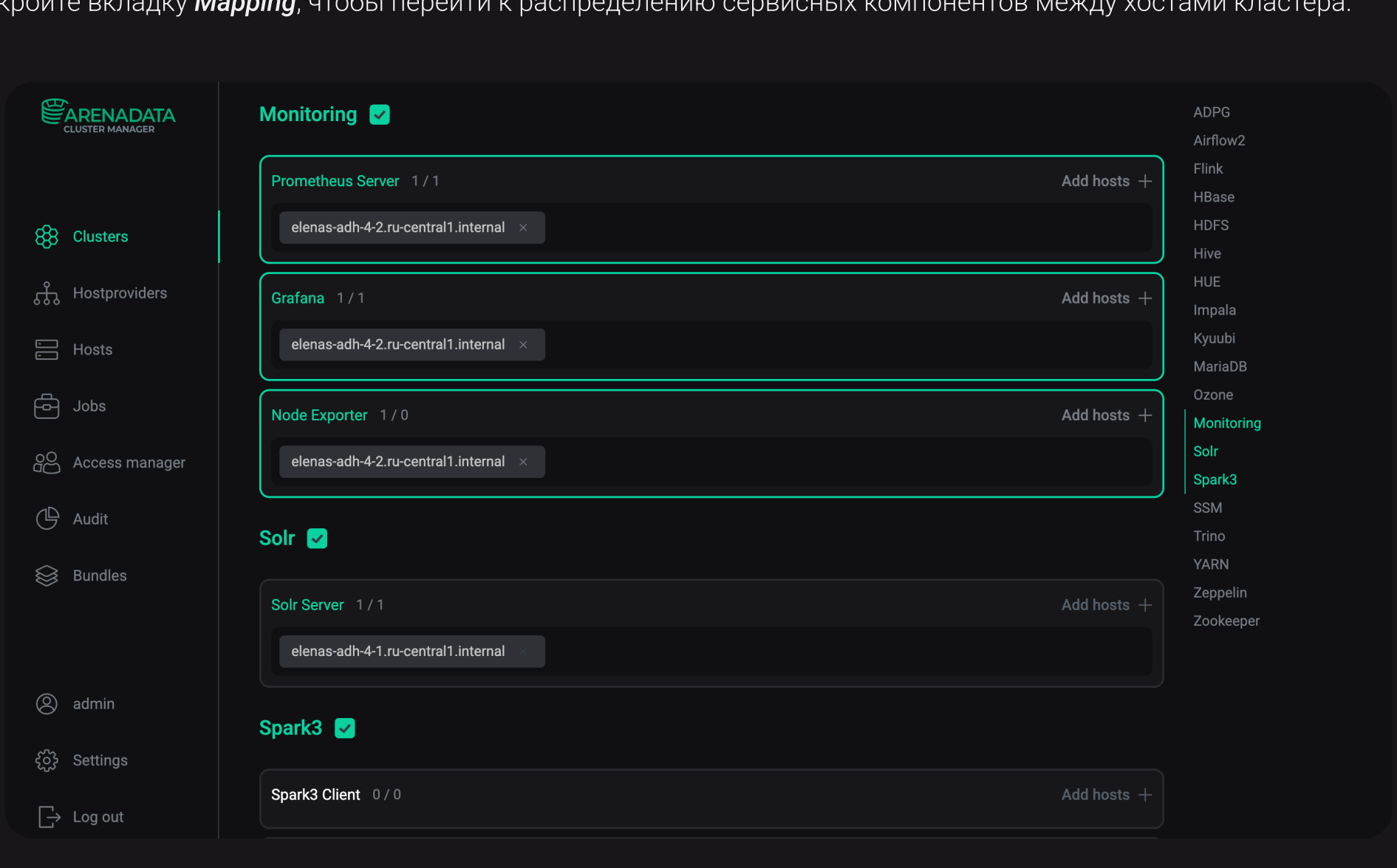
ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH all services / Services
03 Jun 2025 06:55:14 UTC
ADH all services
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
Add services

Name	Version	State	Concerns	Actions
MariaDB	5.5.68	installed		
Monitoring	1.2.0	created		
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Solr	8.11.3	installed		
Spark3	3.5.4_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

Результат успешного добавления сервиса в кластер

Шаг 2. Добавление компонентов

1. На странице кластера откройте вкладку ***Mapping***, чтобы перейти к распределению сервисных компонентов между хостами кластера.



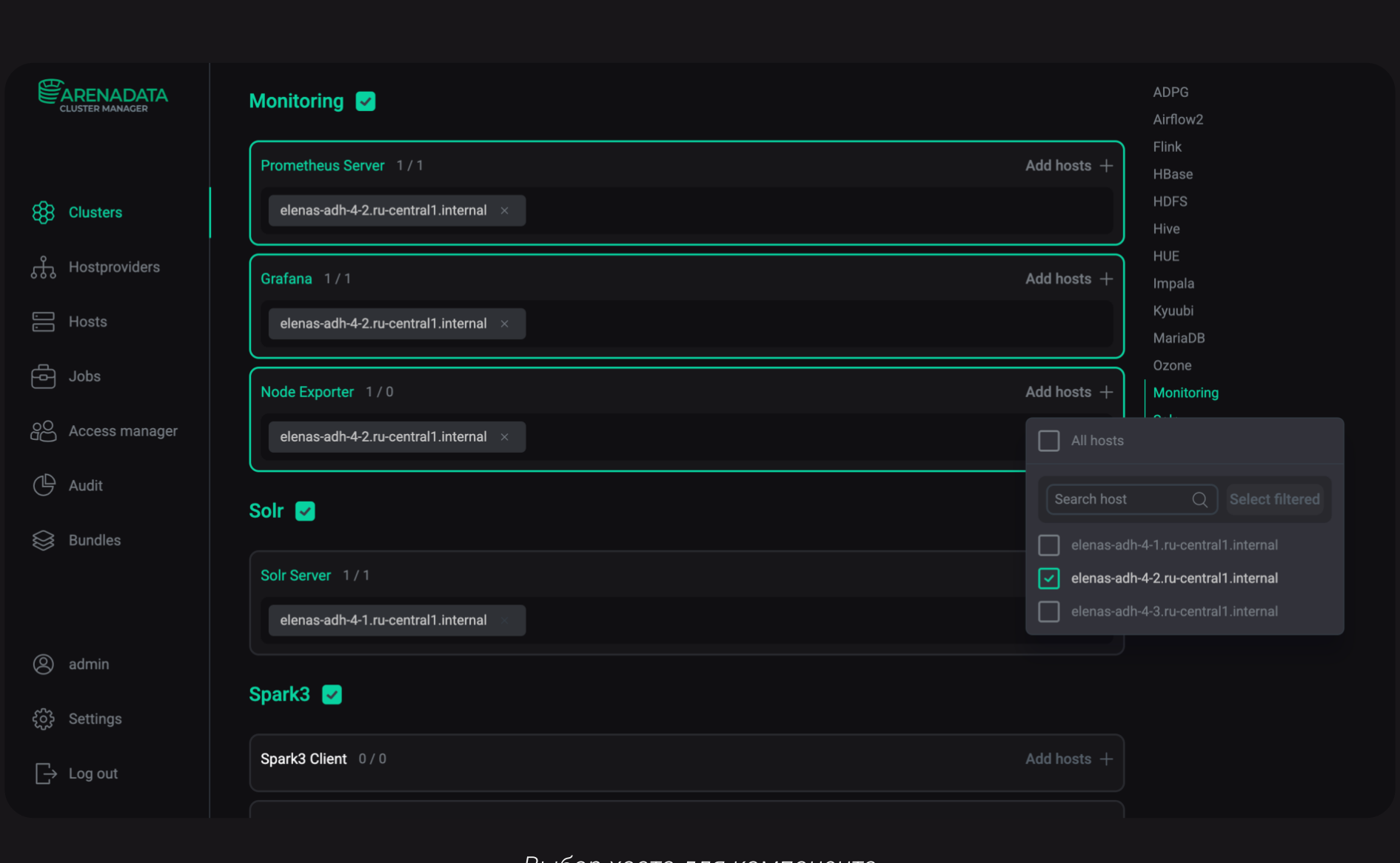
ARENADATA CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Monitoring
Prometheus Server 1 / 1
Add hosts
elenas-adh-4-2.ru-central1.internal
Grafana 1 / 1
Add hosts
elenas-adh-4-2.ru-central1.internal
Node Exporter 1 / 0
Add hosts
elenas-adh-4-2.ru-central1.internal
Solr
Solr Server 1 / 1
Add hosts
elenas-adh-4-1.ru-central1.internal
Spark3
Spark3 Client 0 / 0
Add hosts
ADPG
Airflow2
Flink
HBase
HDFS
Hive
HUE
Impala
Kyuubi
MariaDB
Ozone
Monitoring
Solr
Spark3
SSM
Trino
YARN
Zeppelin
Zookeeper

Переход к распределению сервисных компонентов

Компоненты сервиса Monitoring

Компонент	Описание
Prometheus Server	Хранит все метрики кластера ADH и позволяет Grafana запрашивать их
Grafana	Позволяет визуализировать метрики ADH в виде графиков и диаграмм, организованных в дашборды
Node Exporter	Собирает системные метрики и отправляет их серверу Prometheus

2. Назначьте хосты компонентам сервиса Monitoring — нажмите ***Add hosts*** и выберите нужные хосты в открывшемся окне.



ARENADATA CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Monitoring
Prometheus Server 1 / 1
Add hosts
elenas-adh-4-2.ru-central1.internal
Grafana 1 / 1
Add hosts
elenas-adh-4-2.ru-central1.internal
Node Exporter 1 / 0
Add hosts
elenas-adh-4-2.ru-central1.internal
Solr
Solr Server 1 / 1
elenas-adh-4-1.ru-central1.internal
Spark3
Spark3 Client 0 / 0
Add hosts
All hosts
Search host
Select filtered
elenas-adh-4-1.ru-central1.internal
elenas-adh-4-2.ru-central1.internal
elenas-adh-4-3.ru-central1.internal
ADPG
Airflow2
Flink
HBase
HDFS
Hive
HUE
Impala
Kyuubi
MariaDB
Ozone
Monitoring

Выбор хоста для компонента

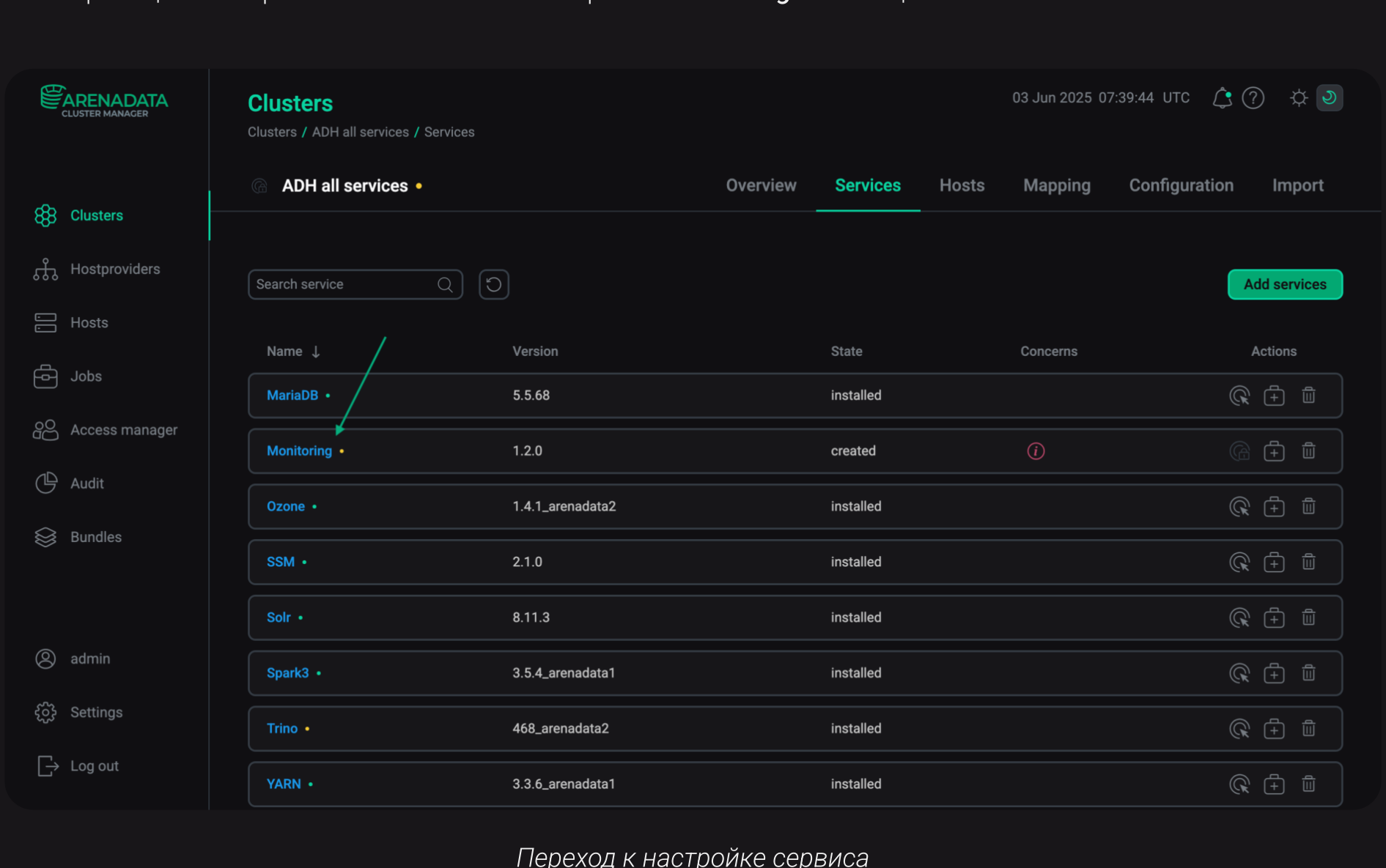
ВНИМАНИЕ

Компонент Prometheus Server не рекомендуется устанавливать на хосты с ADH — используйте для него отдельные хосты. Иначе при выходе из строя хоста ADH и/или критически большой нагрузке на него информация о соответствующих проблемах не будет сохранена.

3. После завершения распределения компонентов нажмите ***Save***, чтобы сохранить изменения.

Шаг 3. Настройка сервиса

1. Откройте вкладку ***Services*** на странице кластера и кликните по имени сервиса ***Monitoring*** в столбце ***Name***.

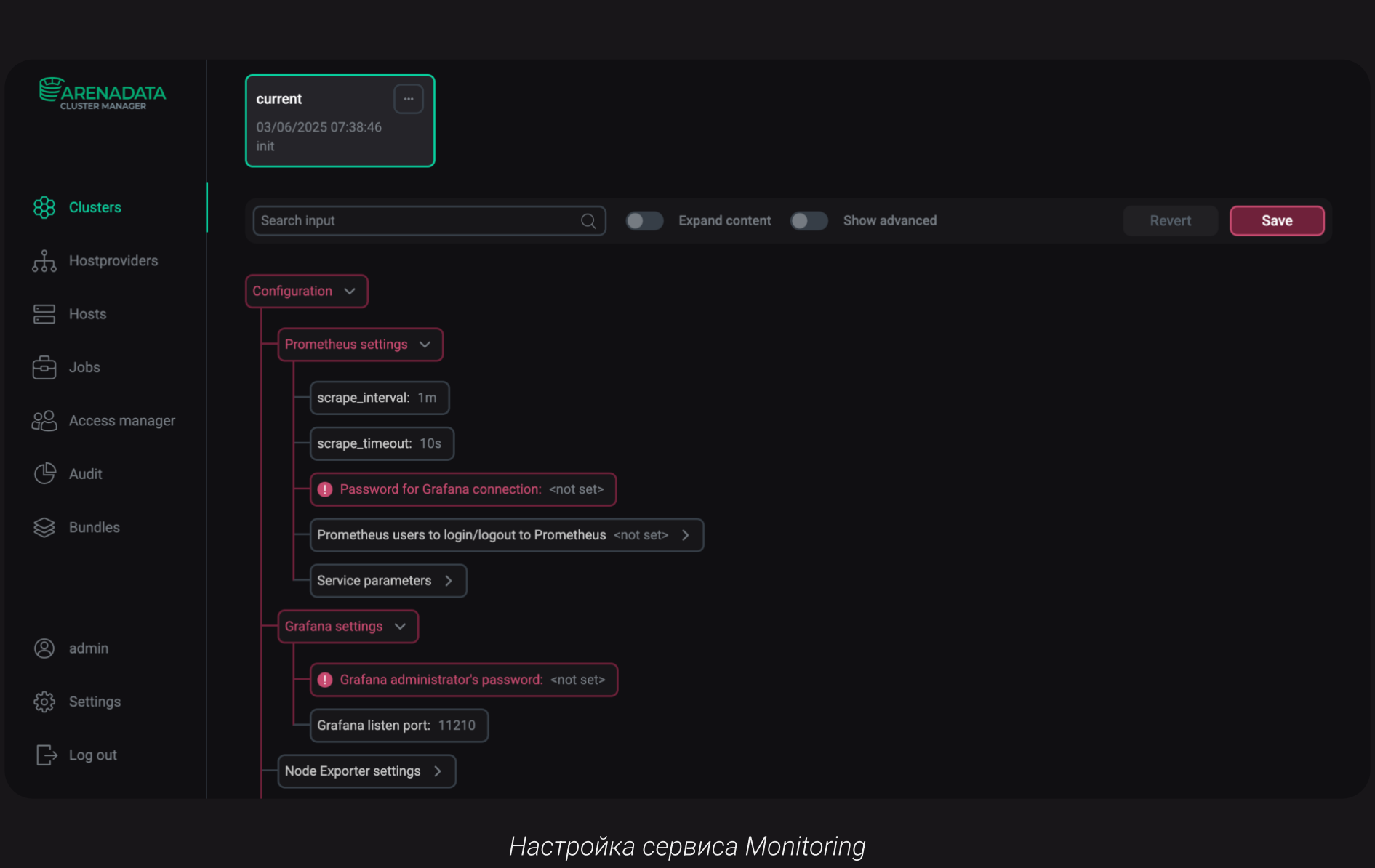


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH all services / Services
03 Jun 2025 07:39:44 UTC
ADH all services
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
Add services

Name	Version	State	Concerns	Actions
MariaDB	5.5.68	installed		
Monitoring	1.2.0	created		
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Solr	8.11.3	installed		
Spark3	3.5.4_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

Переход к настройке сервиса

2. В открывшемся окне заполните конфигурационные параметры сервиса — описания параметров приведены в разделе Monitoring. Поля, подсвеченные красным, обязательны для заполнения.



ARENADATA CLUSTER MANAGER
current
03/06/2025 07:38:46
init
Search input
Expand content
Show advanced
Revert
Save
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out

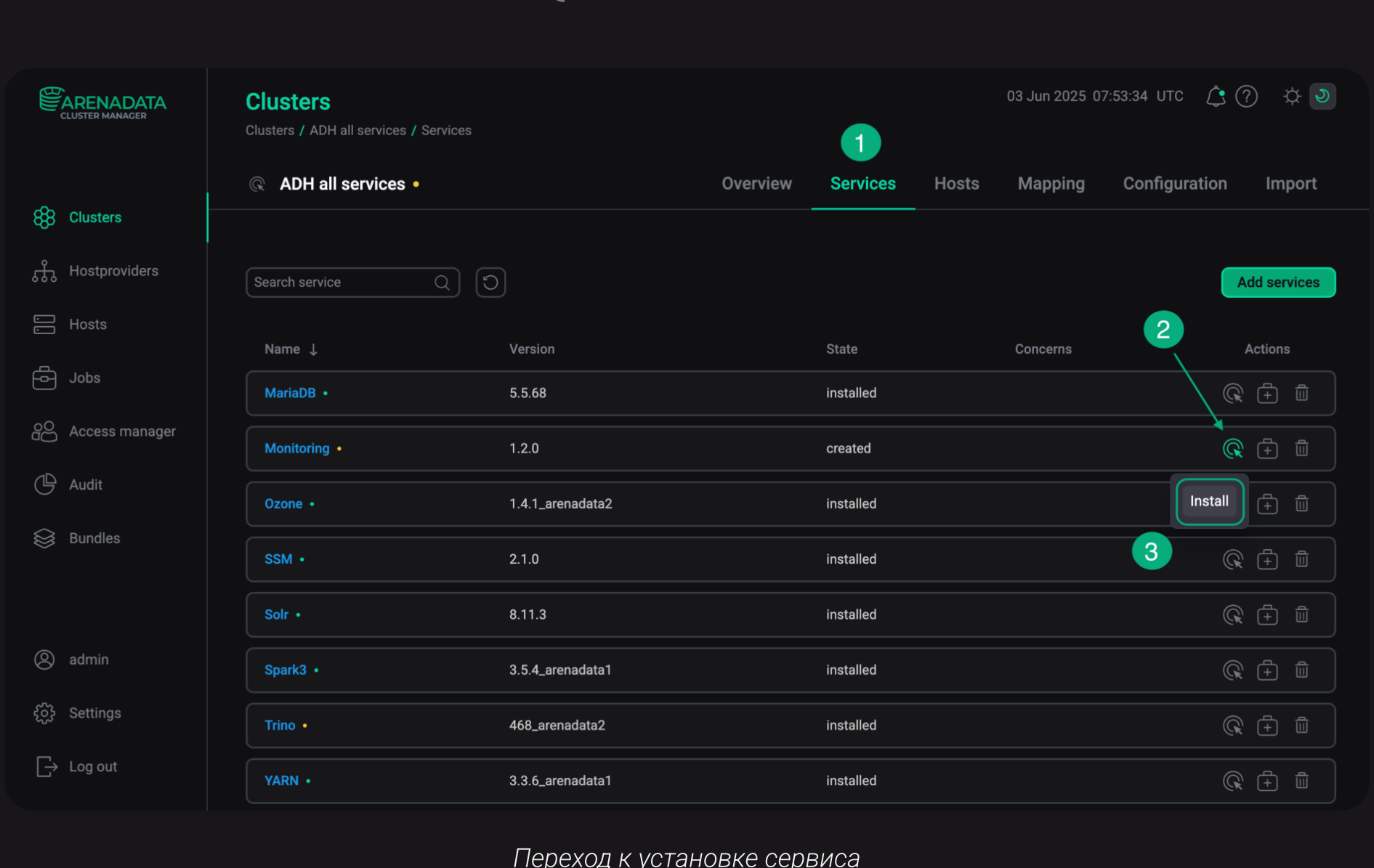
Configuration
- Prometheus settings
  - scrape_interval: 1m
  - scrape_timeout: 10s
  - Password for Grafana connection: <not set>
  - Prometheus users to login/logout to Prometheus <not set>
  - Service parameters
- Grafana settings
  - Grafana administrator's password: <not set>
  - Grafana listen port: 11210
- Node Exporter settings

Настройка сервиса Monitoring

После указания всех необходимых параметров нажмите ***Save***.

Шаг 4. Установка сервиса

1. На вкладке ***Services*** в строке сервиса Monitoring кликните по иконке в столбце ***Actions*** и запустите действие ***Install***.

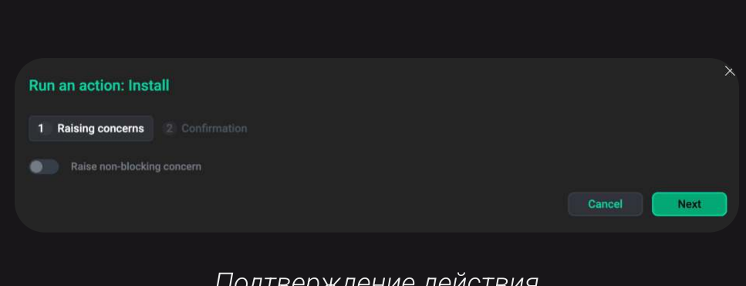


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH all services / Services
03 Jun 2025 07:53:34 UTC
ADH all services
Overview
1
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
Add services
2
Install
3

Name	Version	State	Concerns	Actions
MariaDB	5.5.68	installed		
Monitoring	1.2.0	created		
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Solr	8.11.3	installed		
Spark3	3.5.4_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

Переход к установке сервиса

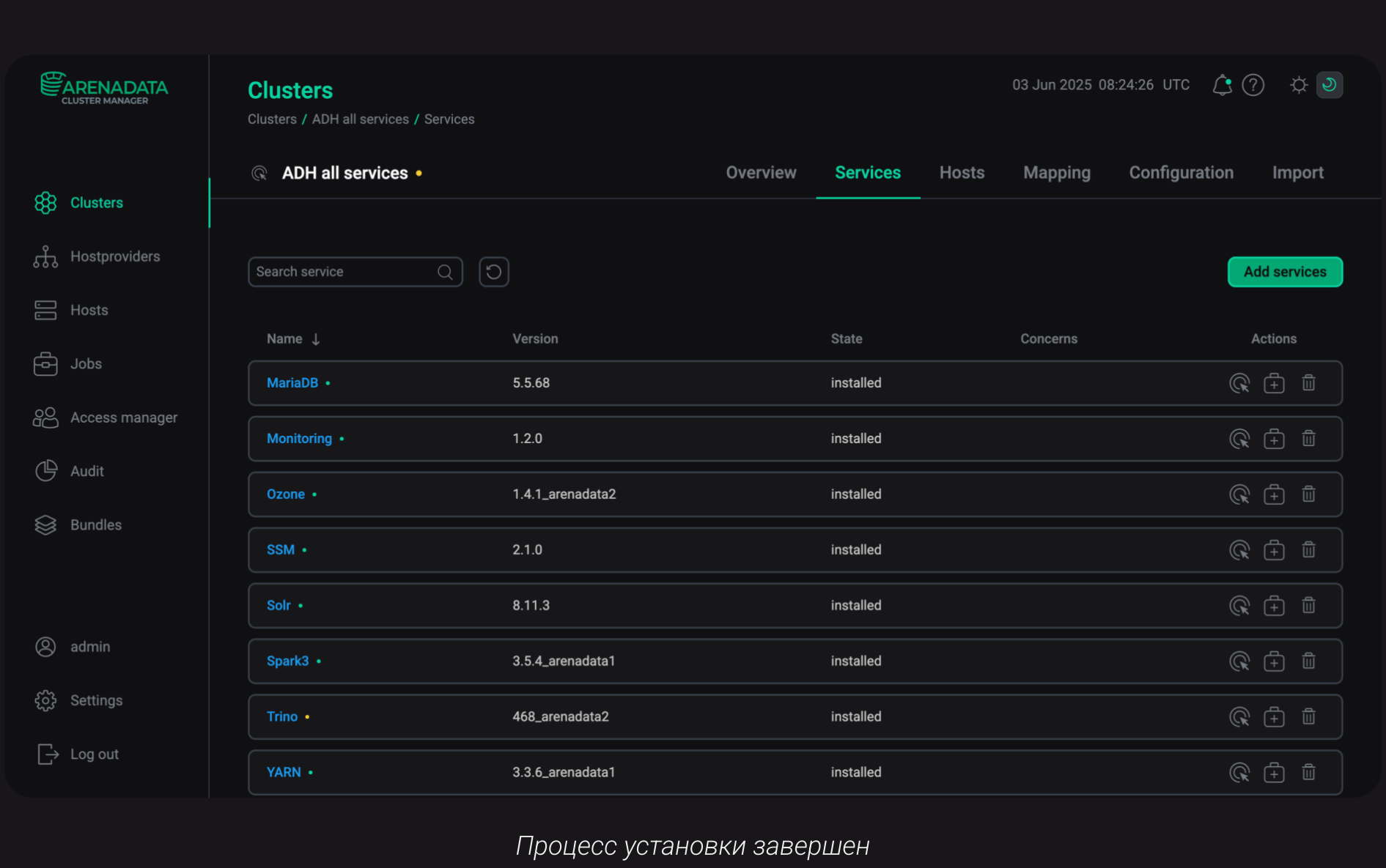
2. Подтвердите действие в открывшемся окне.



Run an action: Install
1 Raising concerns
2 Confirmation
Raise non-blocking concern
Cancel
Next

Подтверждение действия

3. Дождитесь завершения установки и убедитесь, что статус сервиса изменился с `created` на `installed`.

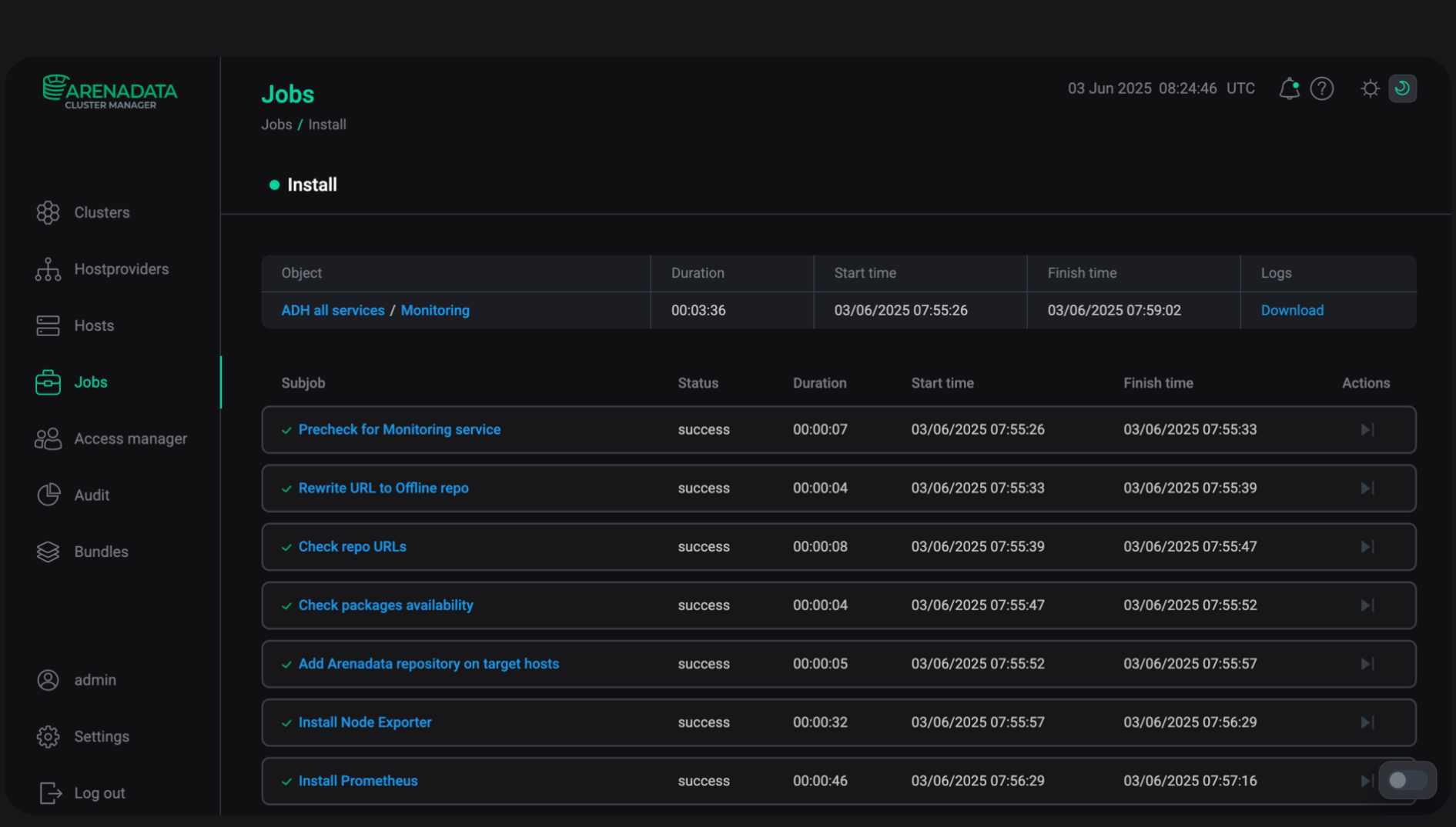


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH all services / Services
03 Jun 2025 08:24:26 UTC
ADH all services
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search service
Add services

Name	Version	State	Concerns	Actions
MariaDB	5.5.68	installed		
Monitoring	1.2.0	installed		
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Solr	8.11.3	installed		
Spark3	3.5.4_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

Процесс установки завершен

Чтобы посмотреть процесс выполнения установки сервиса и проанализировать ошибки в случае их возникновения, выберите пункт ***Jobs*** в левом навигационном меню и кликните по имени задачи ***Install*** в списке задач ***Jobs***.



ARENADATA CLUSTER MANAGER
Jobs
Jobs / Install
03 Jun 2025 08:24:46 UTC
Install
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out

Object	Duration	Start time	Finish time	Logs
ADH all services / Monitoring	00:03:36	03/06/2025 07:55:26	03/06/2025 07:59:02	Download

Subjob	Status	Duration	Start time	Finish time	Actions
Precheck for Monitoring service	success	00:00:07	03/06/2025 07:55:26	03/06/2025 07:55:33	
Rewrite URL to Offline repo	success	00:00:04	03/06/2025 07:55:33	03/06/2025 07:55:39	
Check repo URLs	success	00:00:08	03/06/2025 07:55:39	03/06/2025 07:55:47	
Check packages availability	success	00:00:04	03/06/2025 07:55:47	03/06/2025 07:55:52	
Add Arenadata repository on target hosts	success	00:00:05	03/06/2025 07:55:52	03/06/2025 07:55:57	
Install Node Exporter	success	00:00:32	03/06/2025 07:55:57	03/06/2025 07:56:29	
Install Prometheus	success	00:00:46	03/06/2025 07:56:29	03/06/2025 07:57:16	

Страница задачи Install

Шаг 5. Проверка результатов

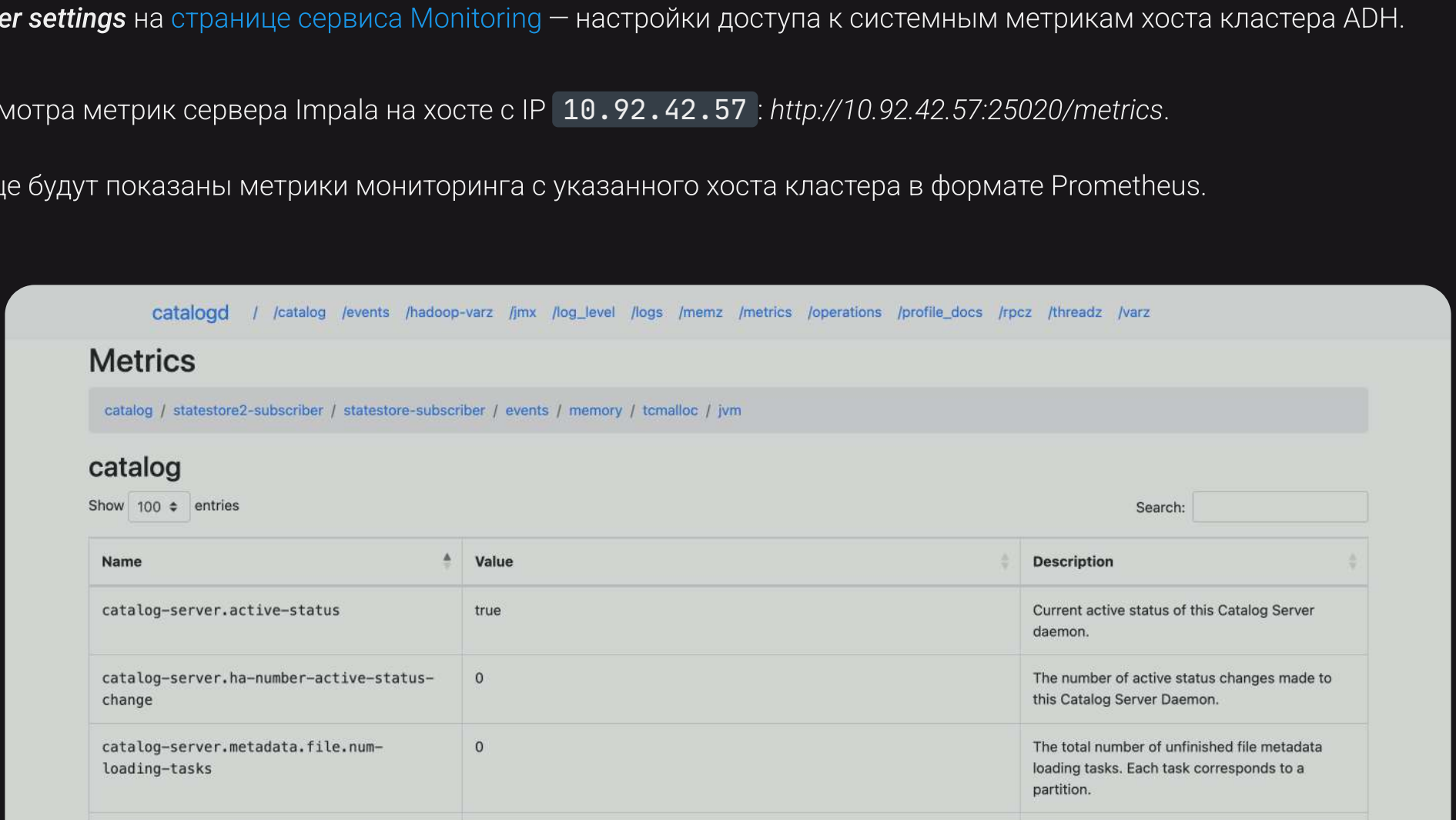
Сервис Monitoring запускается автоматически после установки. По завершении установки необходимо убедиться, что с хостов кластера поступают системные метрики и метрики сервисов ADH. Чтобы это проверить, можно в браузере посмотреть метрики в формате Prometheus, а также использовать веб-интерфейсы Prometheus и Grafana.

Метрики в формате Prometheus

1. В адресной строке браузера введите адрес хоста кластера ADH, а также укажите порт и конечную точку доступа (endpoint) к сервисным или системным метрикам, которые определяются параметрами:
   - в конфигурации сервисов ADH — настройки доступа к метрикам мониторинга этих сервисов;
   - в секции ***Node Exporter settings*** на странице сервиса Monitoring — настройки доступа к системным метрикам хоста кластера ADH.

   Пример адреса для просмотра метрик сервера Impala на хосте с IP `10.92.42.57`: *http://10.92.42.57:25020/metrics*.

2. На открывшейся странице будут показаны метрики мониторинга с указанного хоста кластера в формате Prometheus.



catalogd
/
/catalog
/events
/hadoop-varz
/jmx
/log_level
/logs
/memz
/metrics
/operations
/profile_docs
/rpcz
/threadz
/varz
catalog / statestore2-subscriber / statestore-subscriber / events / memory / tcmalloc / jvm
Show 100 entries
Search:

Metrics

catalog

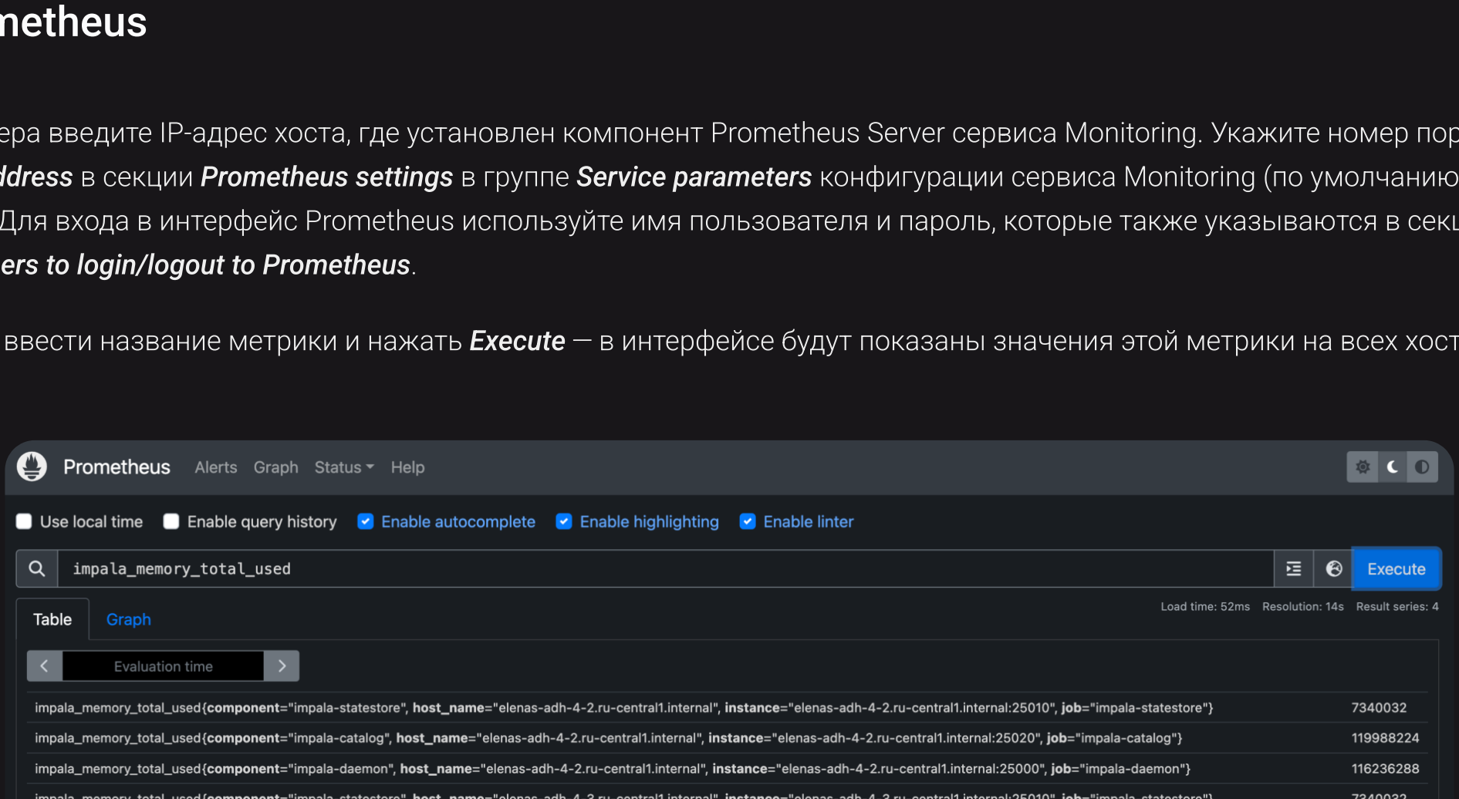
Name	Value	Description
catalog-server.active-status	true	Current active status of this Catalog Server daemon.
catalog-server.ha-number-active-status-change	0	The number of active status changes made to this Catalog Server Daemon.
catalog-server.metadata.file.num-loading-tasks	0	The total number of unfinished file metadata loading tasks. Each task corresponds to a partition.
catalog-server.metadata.file.num-loading-threads	0	The total size of all thread pools used in loading file metadata.
catalog-server.metadata.table.async-loading.num-in-progress	0	The total number of tables that are loading metadata asynchronously. E.g. initial metadata loading triggered by the first access on a table.
catalog-server.metadata.table.async-loading.queue-len	0	The total number of tables that are waiting for loading metadata asynchronously.

Метрики сервера Impala в формате Prometheus

Веб-интерфейс Prometheus

1. В адресной строке браузера введите IP-адрес хоста, где установлен компонент Prometheus Server сервиса Monitoring. Укажите номер порта, который определяется параметром ***web.listen-address*** в секции ***Prometheus settings*** в группе ***Service parameters*** конфигурации сервиса Monitoring (по умолчанию, `11200`). Пример адреса: *http://10.92.40.107:11200*. Для входа в интерфейс Prometheus используйте имя пользователя и пароль, которые также указываются в секции ***Prometheus settings*** — настройка ***Prometheus users to login/logout to Prometheus***.

2. В поле ***Expression*** можно ввести название метрики и нажать ***Execute*** — в интерфейсе будут показаны значения этой метрики на всех хостах кластера ADH.



Prometheus
Alerts
Graph
Status
Help
Use local time
Enable query history
Enable autocomplete
Enable highlighting
Enable linter
Execute
Load time: 52ms
Resolution: 14s
Result series: 4
Table
Graph
Evaluation time
Remove Panel
Add Panel

```
impala_memory_total_used
```

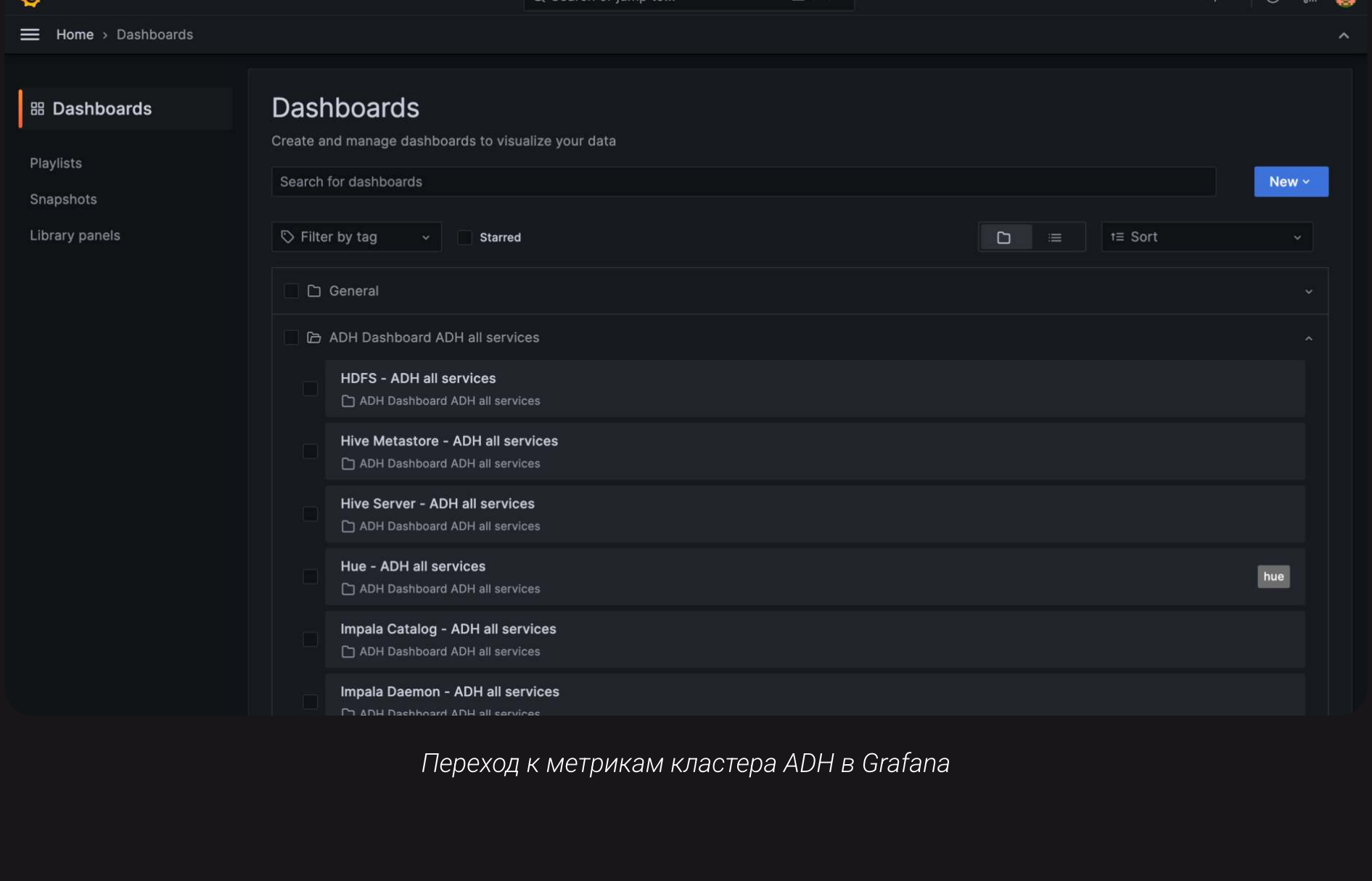
```
impala_memory_total_used{component="impala-statestore", host_name="elenas-adh-4-2.ru-central1.internal", instance="elenas-adh-4-2.ru-central1.internal:25010", job="impala-statestore"}   7340032
impala_memory_total_used{component="impala-catalog", host_name="elenas-adh-4-2.ru-central1.internal", instance="elenas-adh-4-2.ru-central1.internal:25020", job="impala-catalog"}   119988224
impala_memory_total_used{component="impala-daemon", host_name="elenas-adh-4-2.ru-central1.internal", instance="elenas-adh-4-2.ru-central1.internal:25000", job="impala-daemon"}   116236288
impala_memory_total_used{component="impala-statestore", host_name="elenas-adh-4-3.ru-central1.internal", instance="elenas-adh-4-3.ru-central1.internal:25010", job="impala-statestore"}   7340032
```

Веб-интерфейс Prometheus

Веб-интерфейс Grafana

1. В адресной строке браузера введите адрес хоста, на котором развернут компонент Grafana сервиса Monitoring. В качестве номера порта укажите значение параметра ***Grafana listen port*** из раздела ***Grafana settings*** в настройках конфигурации сервиса мониторинга (по умолчанию `11210`). Например, *http://10.92.40.107:11210*. Для входа используйте `admin` в качестве имени пользователя и значение параметра ***Grafana administrator's password*** (находится также в разделе настроек ***Grafana settings***) в качестве пароля.

2. В открывшемся окне выберите пункт меню ***Home → Dashboards*** и разверните секцию ***ADH Dashboard <ADH_cluster_name>***. В этой секции можно выбрать один из доступных дашбордов для просмотра сервисных или системных метрик, поступающих из вашего кластера ADH.



Search or jump to...
cmd+k
Home › Dashboards
Dashboards
Playlists
Snapshots
Library panels
Dashboards
Create and manage dashboards to visualize your data
Search for dashboards
New
Filter by tag
Starred
Sort
General
ADH Dashboard ADH all services
HDFS - ADH all services
ADH Dashboard ADH all services
Hive Metastore - ADH all services
ADH Dashboard ADH all services
Hive Server - ADH all services
ADH Dashboard ADH all services
Hue - ADH all services
hue
ADH Dashboard ADH all services
Impala Catalog - ADH all services
ADH Dashboard ADH all services
Impala Daemon - ADH all services

Переход к метрикам кластера ADH в Grafana

Кластер мониторинга

Елена Костюченко

Назад к содержанию

Чтобы настроить мониторинг для ADH при **offline**-установке, можно добавить сервисы мониторинга (Diamond, Graphite и Grafana) в кластер Enterprise Tools (необходимости в отдельном кластере мониторинга нет, так как его функции выполняет кластер Enterprise Tools) и интегрировать его с кластером ADH. Шаги для установки мониторинга в этом случае приведены ниже. Настройку кластера Enterprise Tools можно выполнить до и после установки кластера ADH, а шаг Интеграция с кластером ADH может быть выполнен только на базе добавленного кластера ADH.

- Настройка кластера Enterprise Tools
- Интеграция с кластером ADH

Настройка кластера Enterprise Tools

Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Добавление сервисов мониторинга в кластер ET

Шаг 2. Распределение компонентов мониторинга

Шаг 3. Настройка сервисов мониторинга

Шаг 4. Установка сервисов мониторинга

Шаг 5. Проверка результатов

- Graphite
- Grafana

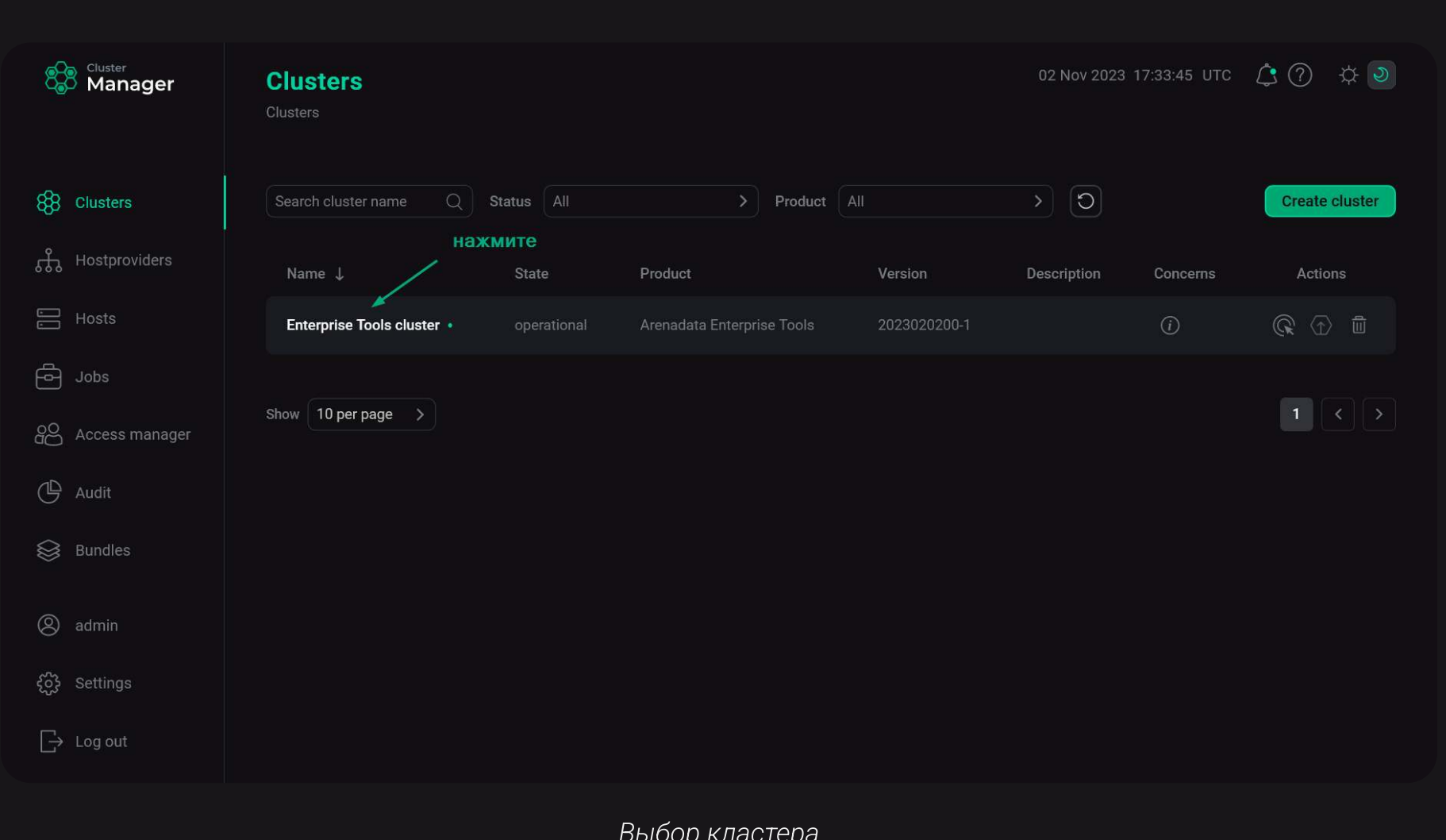
ВАЖНО

Перед настройкой мониторинга убедитесь, что кластер Enterprise Tools (ET) установлен в ADCM. Все последующие действия выполняются в нем.

Шаг 1. Добавление сервисов мониторинга в кластер ET

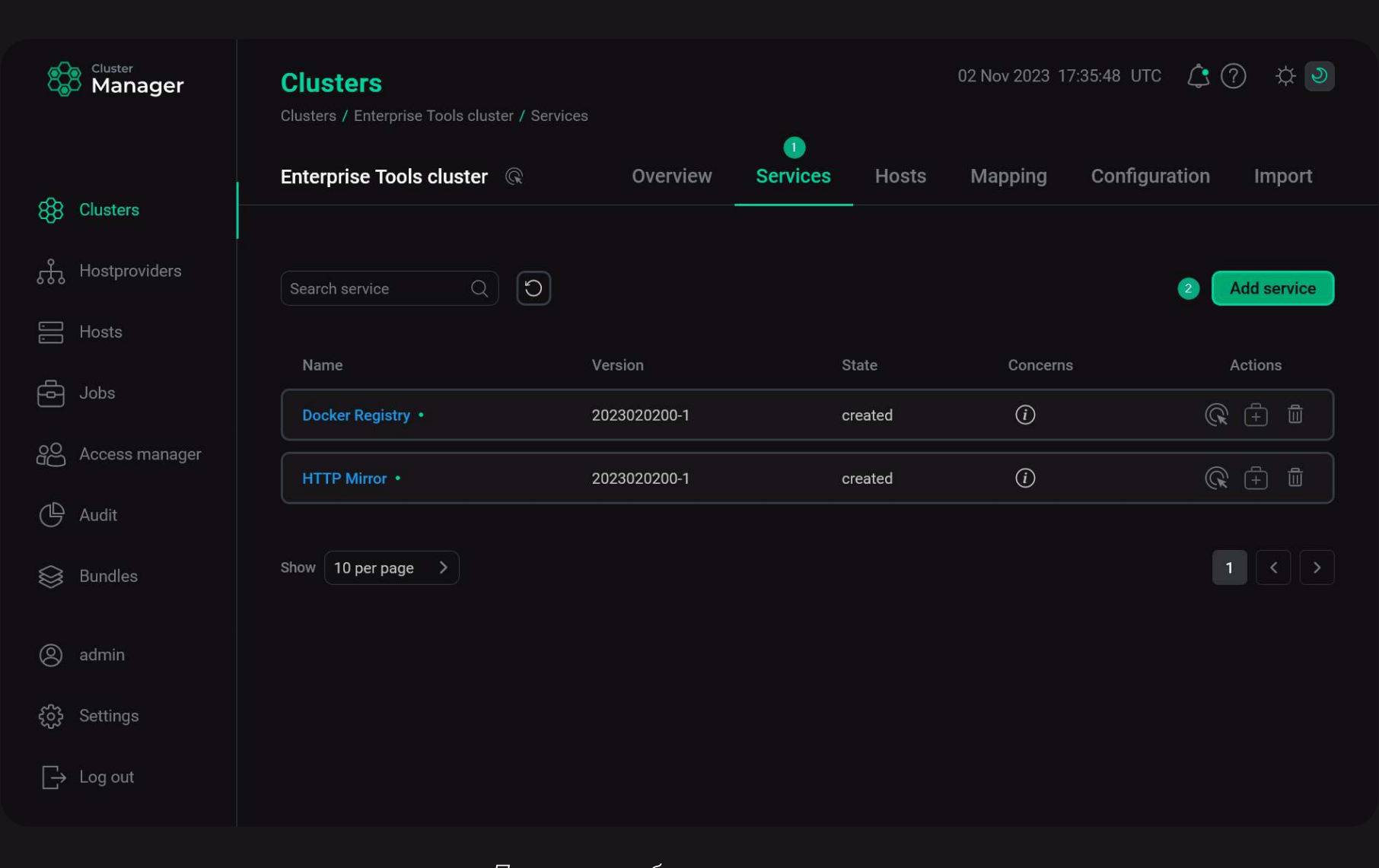
Для добавления сервисов в кластер:

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Cluster Manager
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
02 Nov 2023 17:33:45 UTC
Create cluster
нажмите
Name
State
Product
Version
Description
Concerns
Actions
Enterprise Tools cluster
operational
Arenadata Enterprise Tools
2023020200-1
10 per page

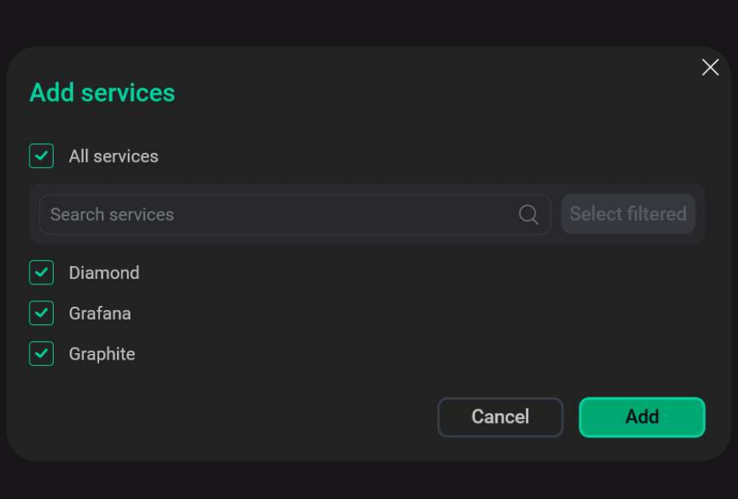
Выбор кластера

2. Откройте вкладку ***Services*** на странице кластера и нажмите ***Add service***.


Clusters
Clusters / Enterprise Tools cluster / Services
Enterprise Tools cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Add service
Docker Registry
HTTP Mirror
2023020200-1
created

Переход к добавлению сервисов

3. В открывшемся окне выберите сервисы для добавления в кластер и нажмите ***Add***.


Add services
All services
Search services
Select filtered
Diamond
Grafana
Graphite
Cancel
Add

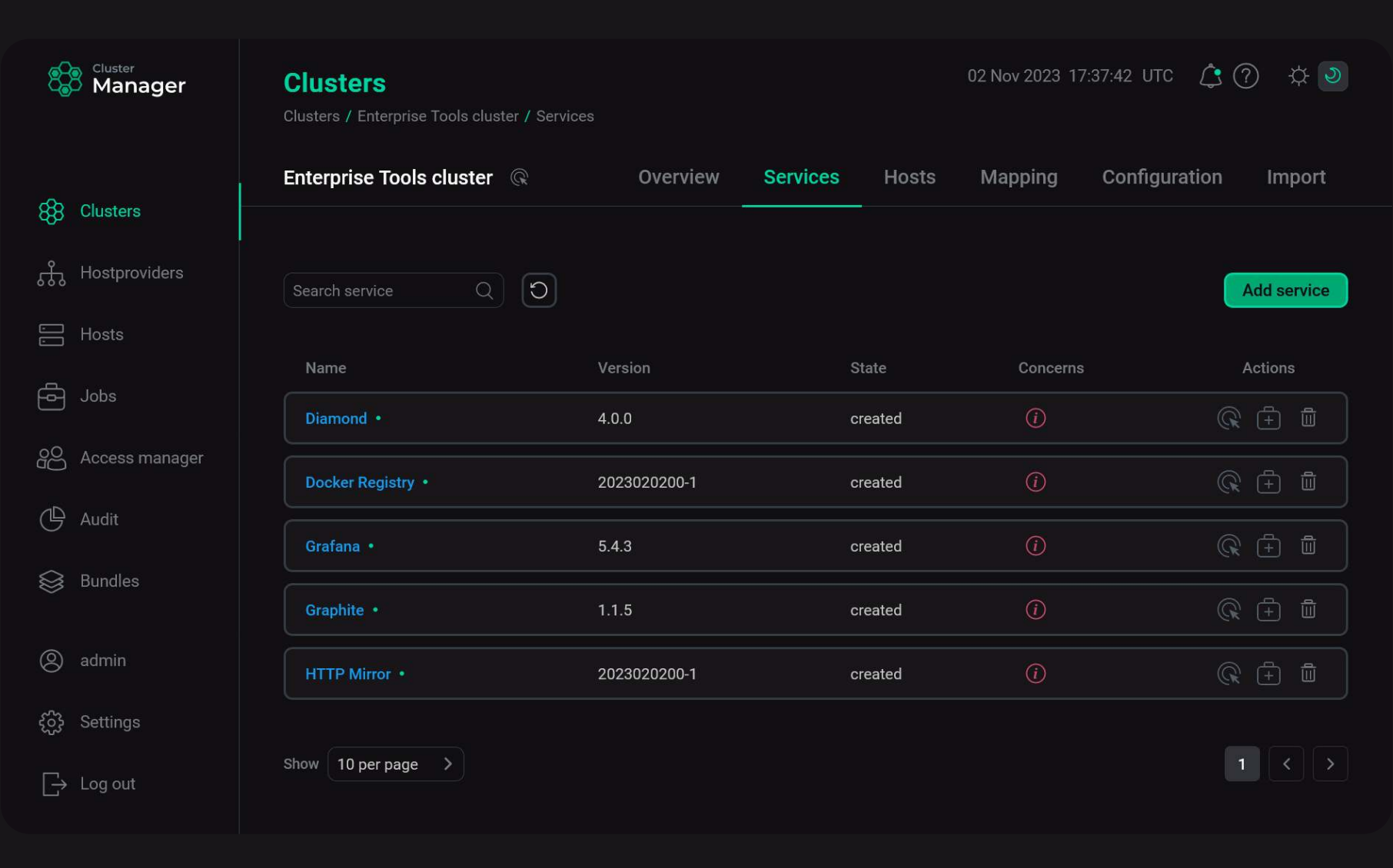
Выбор сервисов

Краткое описание доступных сервисов приведено ниже.

Сервисы мониторинга, которые необходимо добавить в кластер Enterprise Tools

Сервис	Назначение
Diamond	Python-сервис, используемый для сбора системных метрик с хостов, подключенных к системе мониторинга. Собранные метрики обрабатываются в Carbon (backend-сервисах Graphite)
Graphite	Состоит из нескольких компонентов: ▪ **Carbon** — набор сервисов для обработки метрик, поступающих из Diamond. ▪ **Whisper DB** — временная файловая база данных, где хранятся собранные метрики. ▪ **Web service** — web-сервис, позволяющий получать и визуализировать метрики из базы данных Whisper DB.
Grafana	Web-сервис, предназначенный для создания графиков и оповещений для визуализации метрик, полученных из Graphite. По умолчанию использует порт `3000`

4. В результате успешно добавленные сервисы отображаются на вкладке ***Services***.

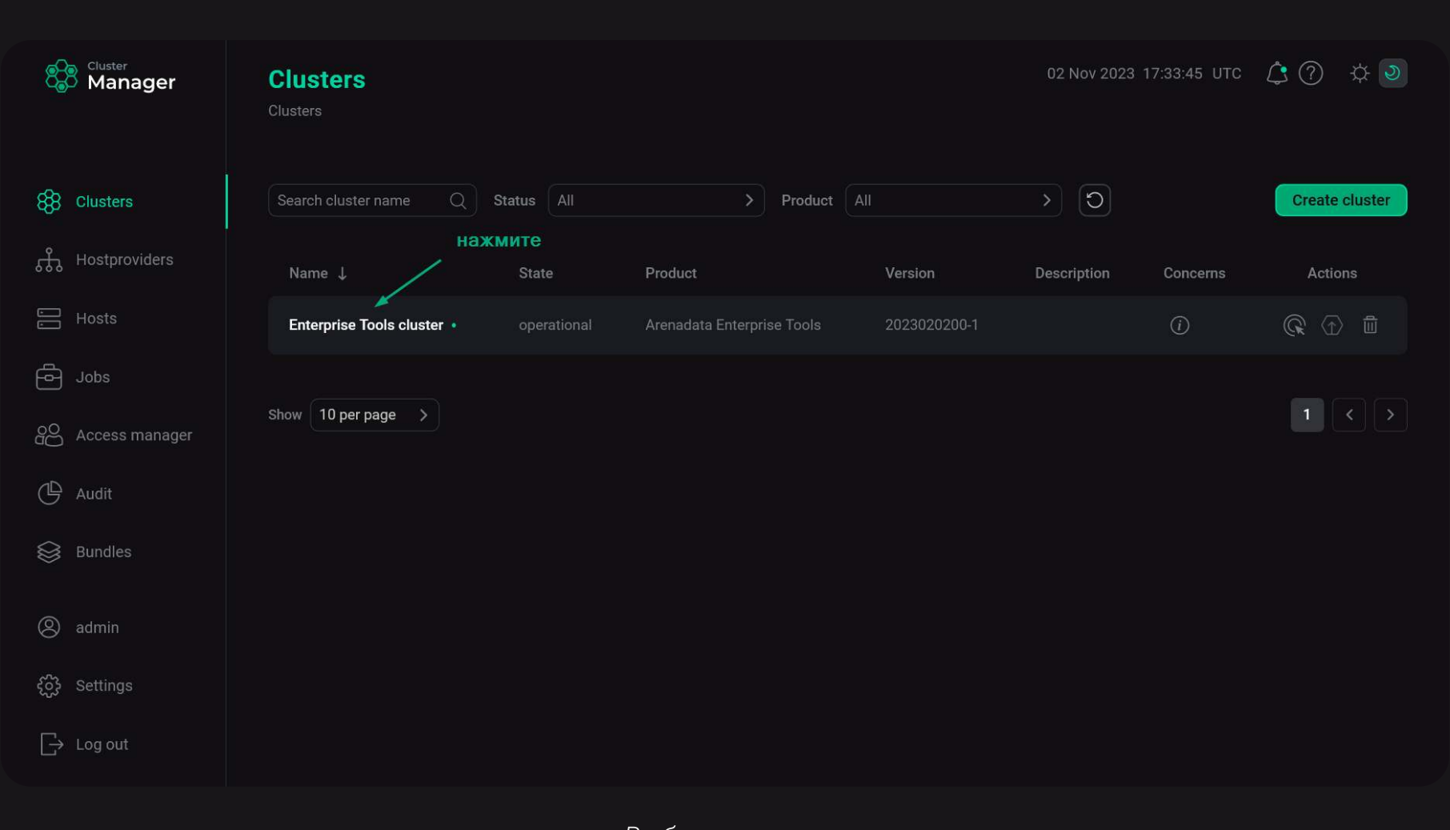

Enterprise Tools cluster
Services
Diamond
Docker Registry
Grafana
Graphite
HTTP Mirror
4.0.0
2023020200-1
5.4.3
1.1.5
created

Результат успешного добавления сервисов в кластер

Шаг 2. Распределение компонентов мониторинга

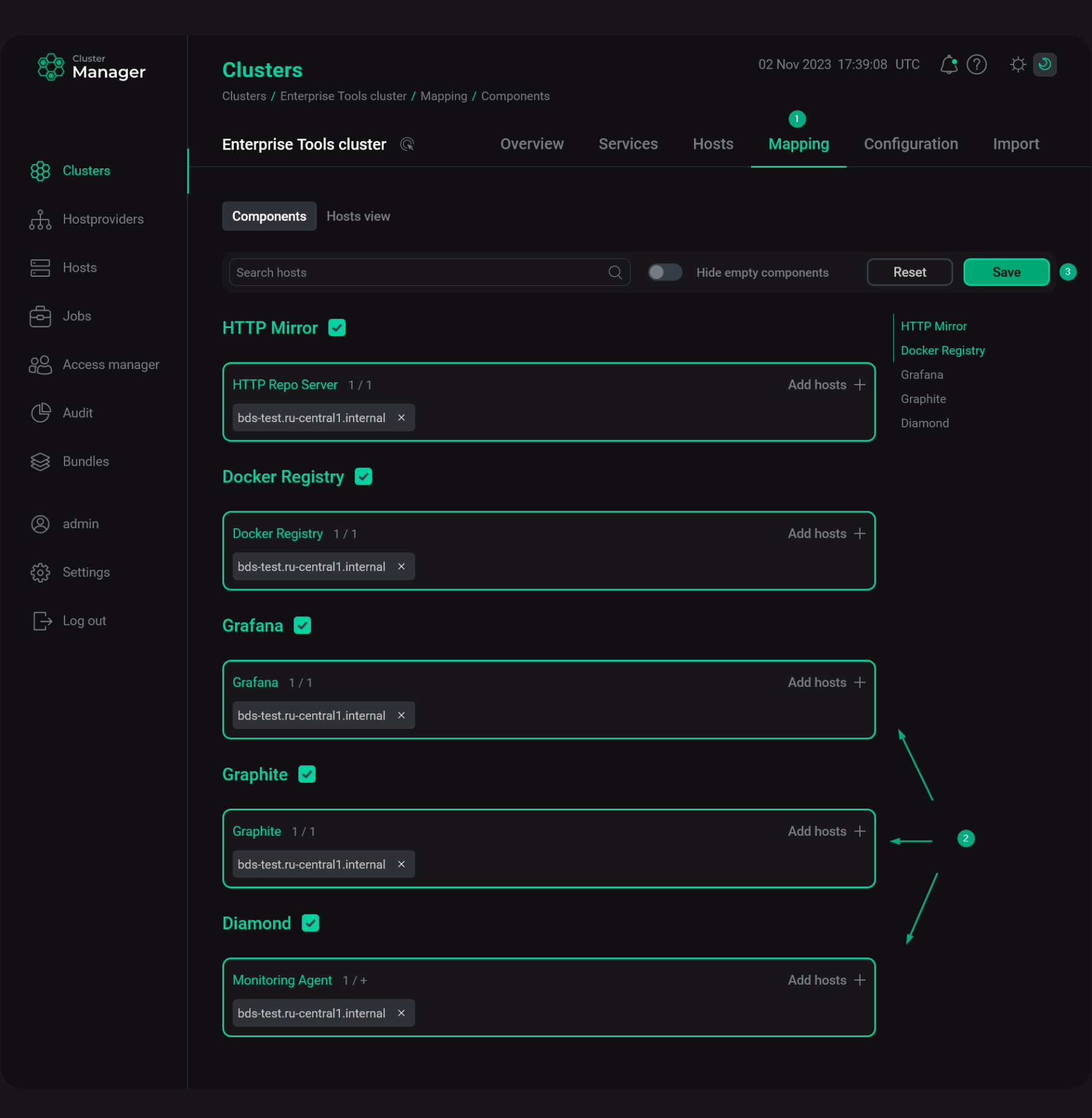
Для добавления компонентов на хосты кластера:

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Clusters
нажмите
Enterprise Tools cluster

Выбор кластера

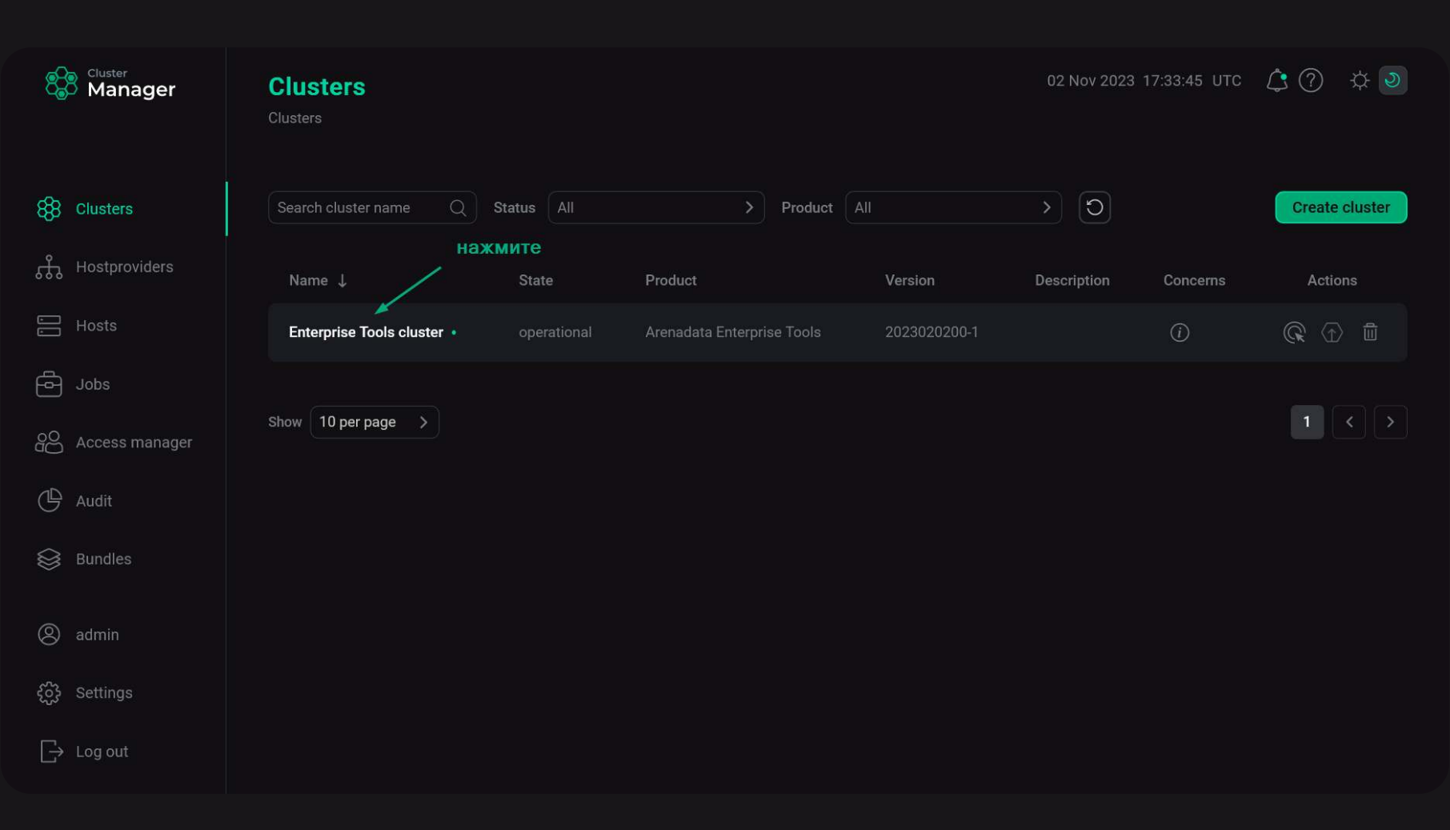
2. Откройте вкладку ***Mapping*** на странице кластера, добавьте компоненты мониторинга на все хосты кластера и нажмите ***Save***.


Clusters / Enterprise Tools cluster / Mapping / Components
Mapping
Components
Hosts view
Hide empty components
Reset
Save
HTTP Mirror
HTTP Repo Server 1 / 1
Docker Registry
Docker Registry 1 / 1
Grafana
Grafana 1 / 1
Graphite
Graphite 1 / 1
Diamond
Monitoring Agent 1 / +
Add hosts
bds-test.ru-central1.internal

Распределение компонентов между хостами

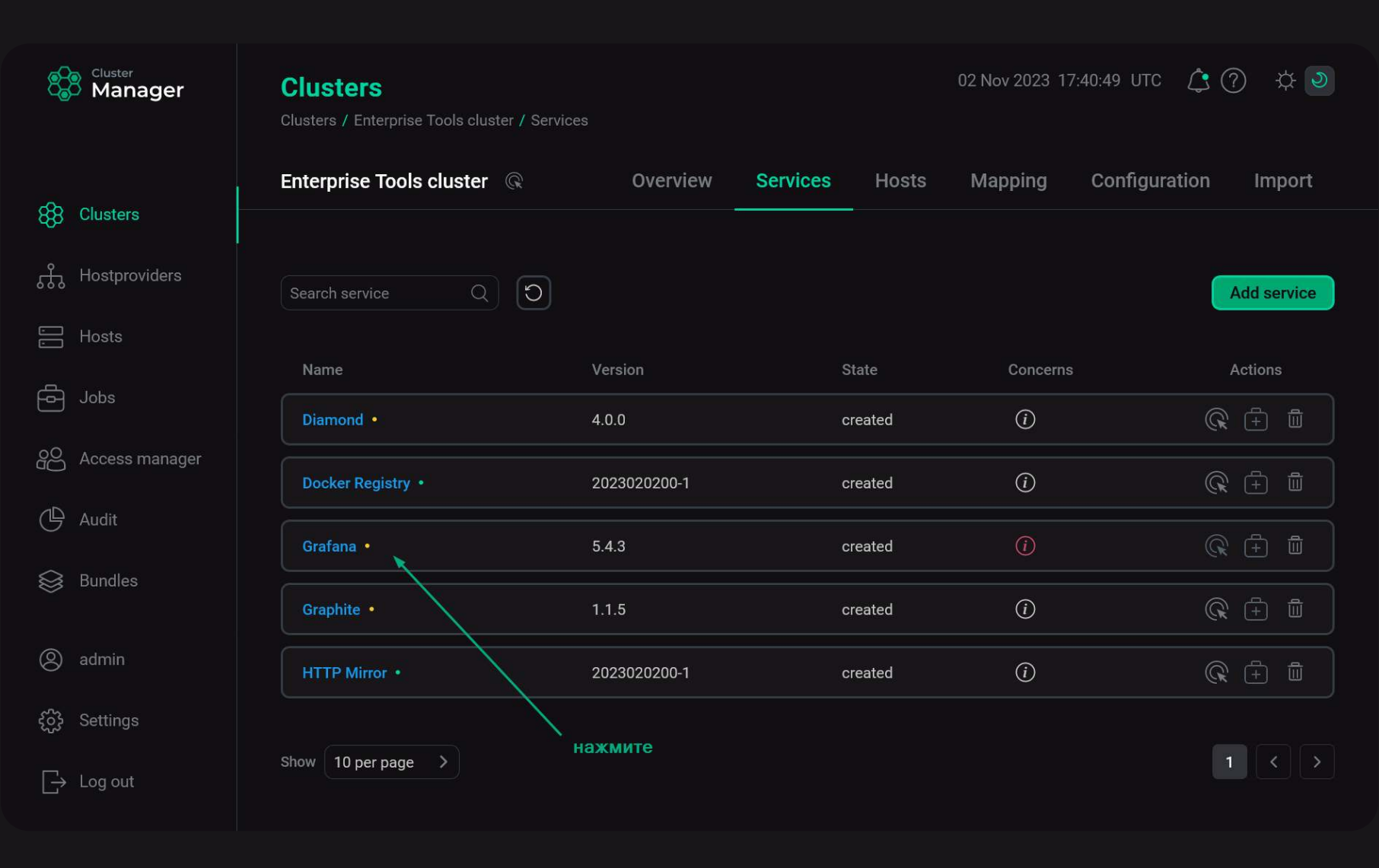
Шаг 3. Настройка сервисов мониторинга

1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.


Clusters
нажмите
Enterprise Tools cluster

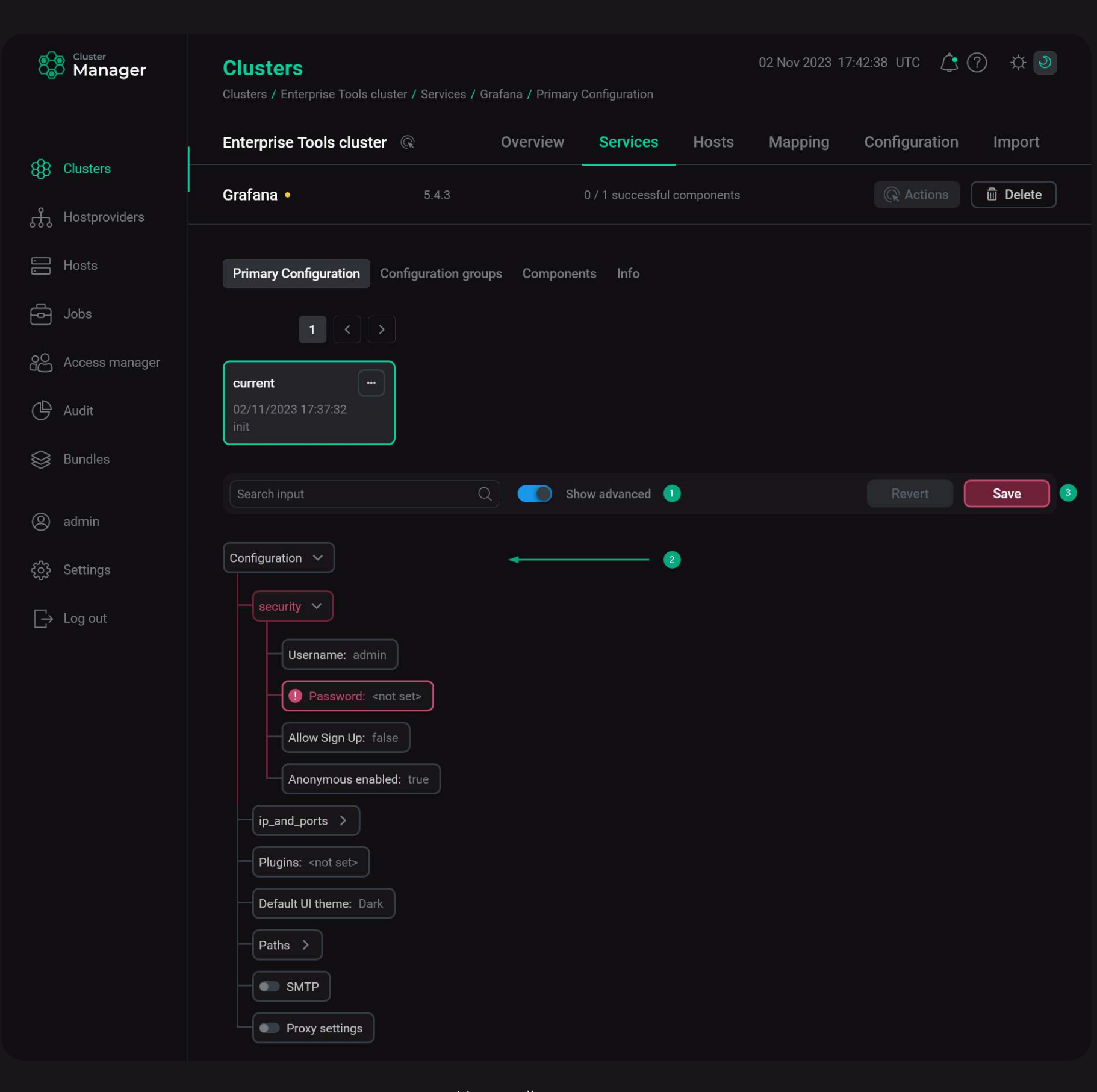
Выбор кластера

2. Откройте вкладку ***Services*** на странице кластера. Нажмите на имя сервиса в столбце ***Name***.


Services
Diamond
Docker Registry
Grafana
Graphite
HTTP Mirror
нажмите

Переход к настройке сервисов

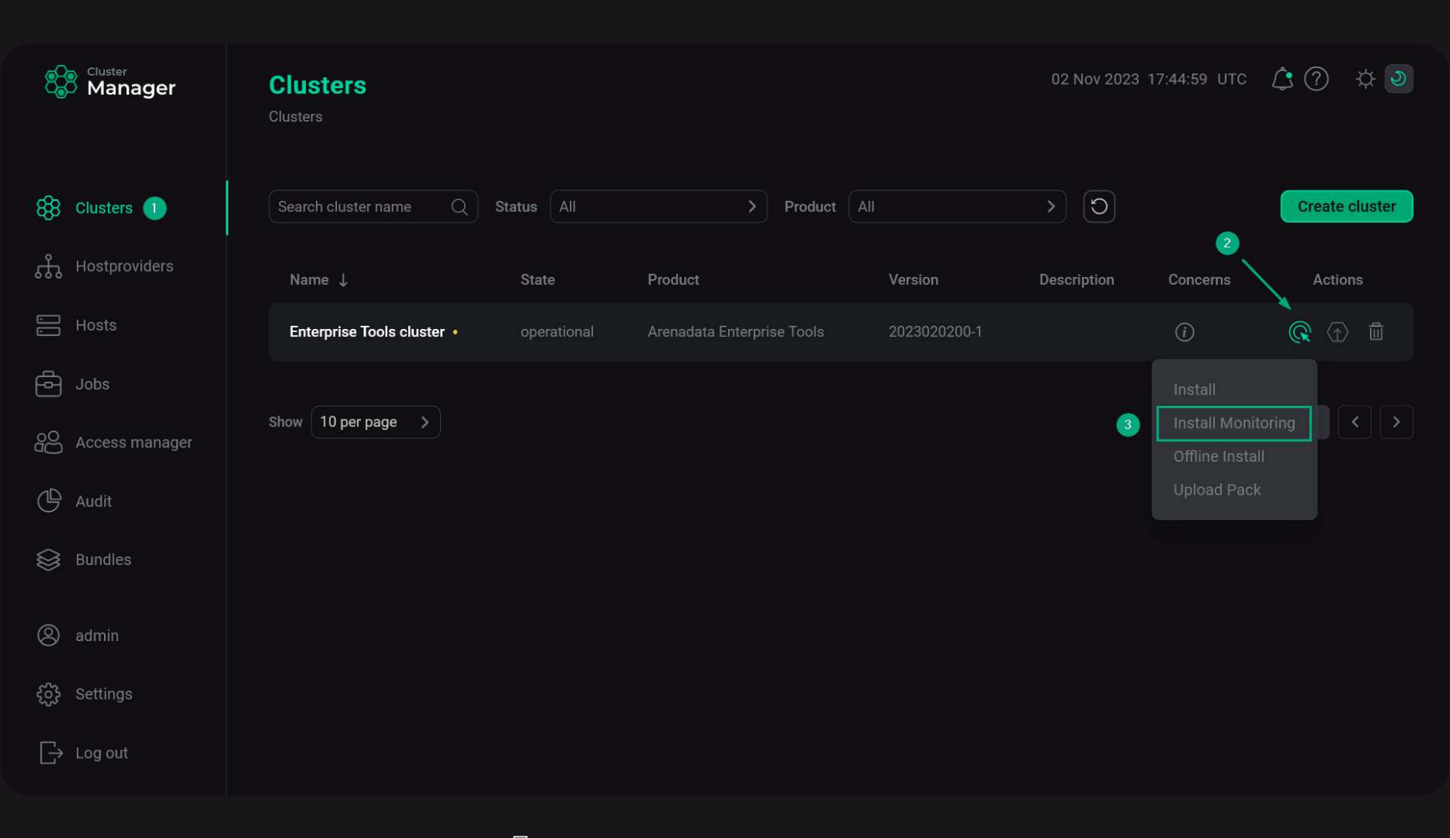
3. В открывшемся окне:
   - Откройте вкладку ***Primary Configuration***.
   - Переведите в активное состояние переключатель ***Show advanced***.
   - Заполните конфигурационные параметры выбранного сервиса. Поля, подсвеченные красным, обязательны для заполнения.
   - Нажмите ***Save***.


Clusters / Enterprise Tools cluster / Services / Grafana / Primary Configuration
Grafana
5.4.3
0 / 1 successful components
Actions
Delete
Primary Configuration
Configuration groups
Components
Info
current
02/11/2023 17:37:32
init
Show advanced
Revert
Save
Configuration
security
Username: admin
Password: <not set>
Allow Sign Up: false
Anonymous enabled: true
ip_and_ports
Plugins: <not set>
Default UI theme: Dark
Paths
SMTP
Proxy settings

Настройка сервиса

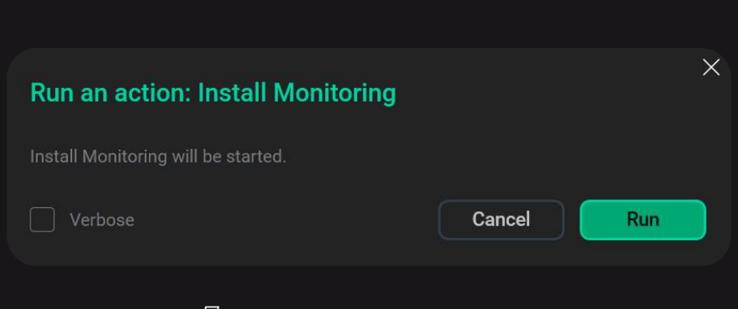
Шаг 4. Установка сервисов мониторинга

1. Запустите действие ***Install Monitoring*** для кластера, выбранного на странице ***Clusters***, нажав на иконку в столбце ***Actions***.


Clusters
Enterprise Tools cluster
Install
Install Monitoring
Offline Install
Upload Pack

Переход к установке мониторинга

2. Нажмите ***Run*** в открывшемся окне.


Run an action: Install Monitoring
Install Monitoring will be started.
Verbose
Cancel
Run

Подтверждение установки

3. Дождитесь завершения процесса установки.

ПРИМЕЧАНИЕ

В случае ошибок логи можно найти на странице ***Jobs***.

Шаг 5. Проверка результатов

Graphite

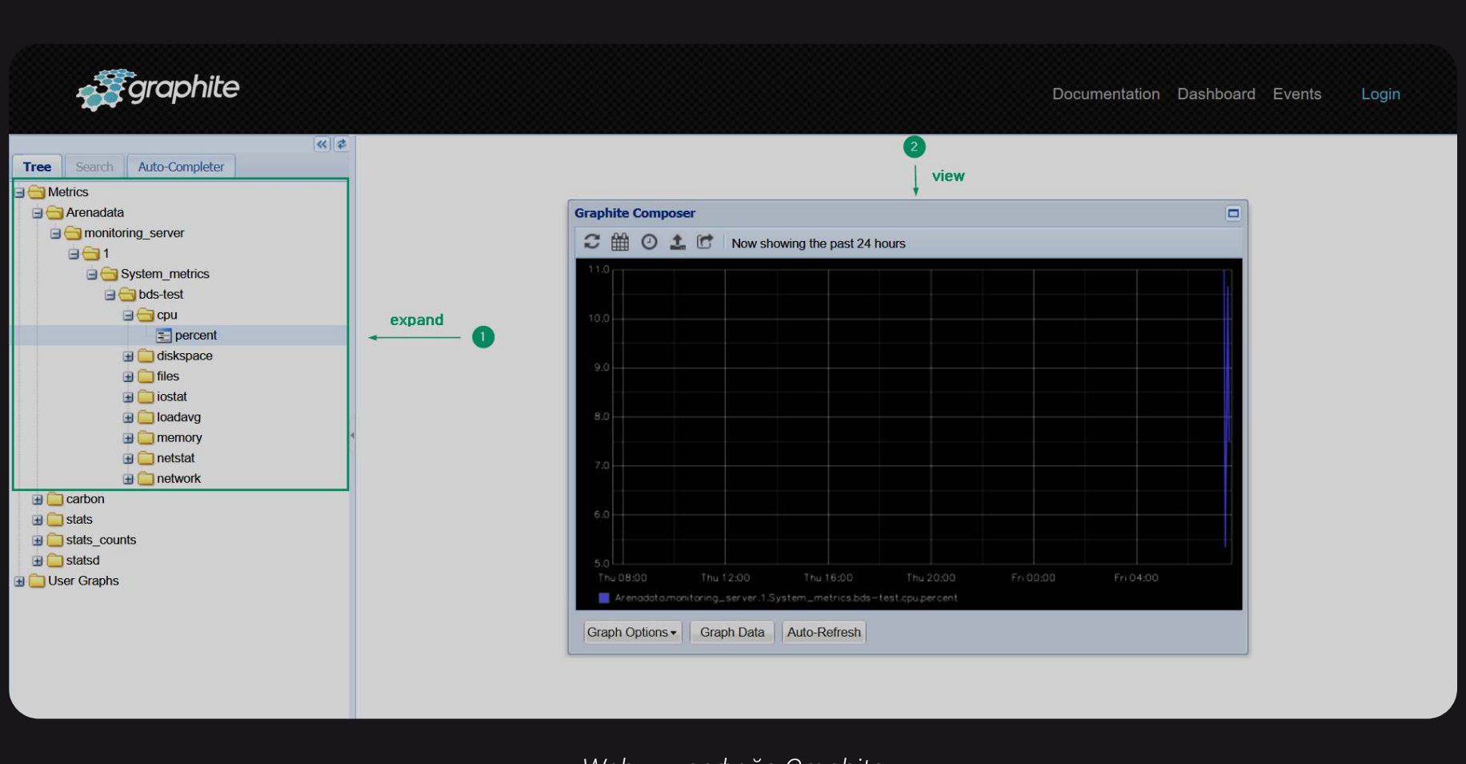
Чтобы проверить доступность web-интерфейса Graphite, необходимо выполнить шаги:

1. В адресной строке браузера введите адрес хоста, на котором развернут кластер мониторинга. В качестве номера порта укажите значение параметра ***Web-interface TCP port*** из раздела ***ip_and_ports***, определенное на этапе конфигурирования сервиса Graphite при установке кластера мониторинга. Значение `80` (используемое по умолчанию) можно не указывать.

ВАЖНО

В случае offline-установки этот адрес совпадает с адресом хоста, на котором развернут кластер Enterprise Tools.

2. Разверните дерево в левой части открывшейся формы. Здесь отображаются метрики System метрики с группировкой по хостам. Можно выбрать любую из метрик для просмотра динамики ее изменения в виде графика в правой части экрана.

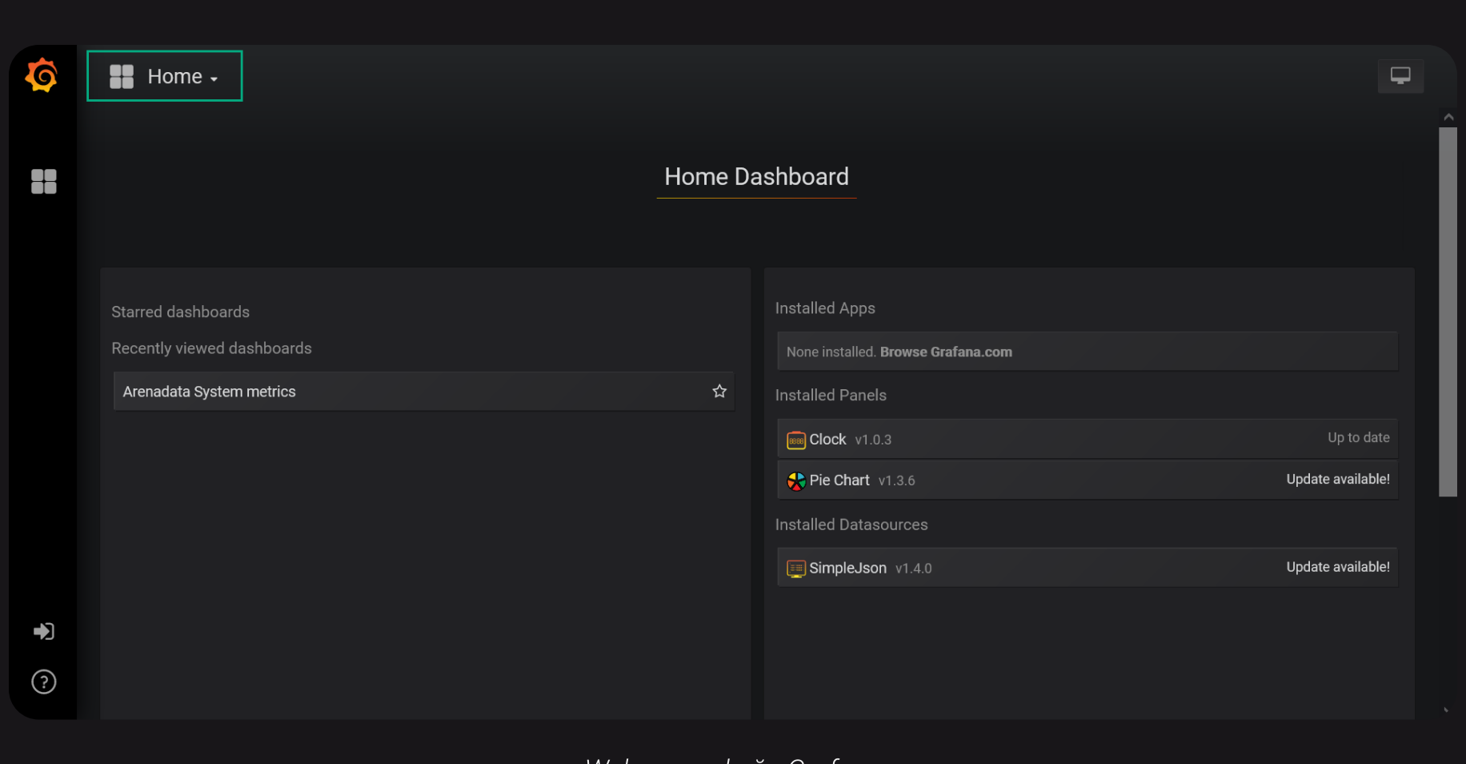

graphite
Documentation
Dashboard
Events
Login
Tree
Search
Auto-Completer
Metrics
Arenadata
monitoring_server
System_metrics
bds-test
cpu
percent
diskspace
files
iostat
loadavg
memory
netstat
network
carbon
stats
stats_counts
statsd
User Graphs
expand
view
Graphite Composer
Now showing the past 24 hours
Graph Options
Graph Data
Auto-Refresh

Web-интерфейс Graphite

Grafana

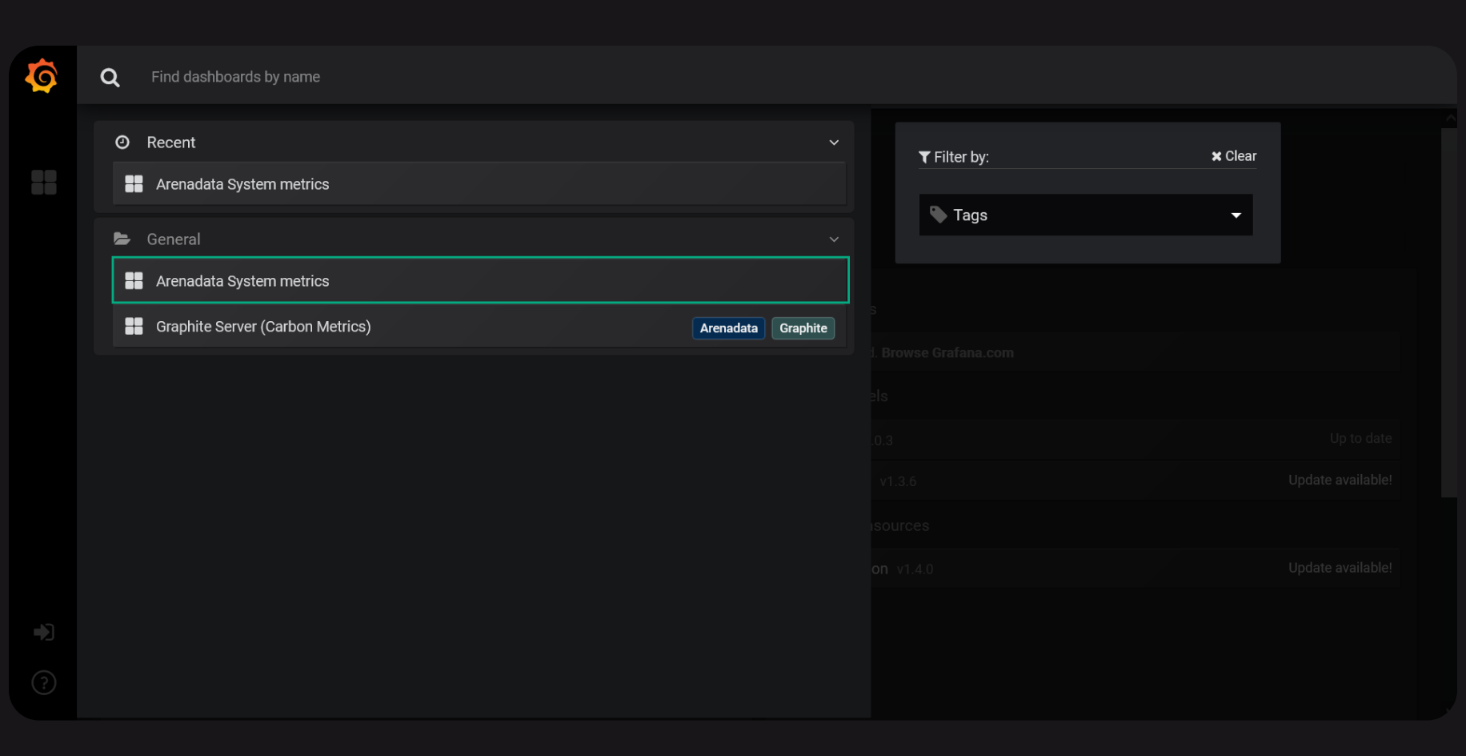
Чтобы проверить доступность web-интерфейса Grafana, необходимо выполнить шаги:

1. В адресной строке браузера введите адрес хоста, на котором развернут кластер мониторинга. В качестве номера порта укажите значение параметра ***Port*** из раздела ***ip_and_ports***, определенное на этапе конфигурирования сервиса Grafana при установке кластера мониторинга (по умолчанию `3000`).
2. Выберите пункт меню ***Home*** в открывшемся окне.


Home
Home Dashboard
Starred dashboards
Recently viewed dashboards
Arenadata System metrics
Installed Apps
None installed. Browse Grafana.com
Installed Panels
Clock v1.0.3
Up to date
Pie Chart v1.3.6
Update available!
Installed Datasources
SimpleJson v1.4.0
Update available!

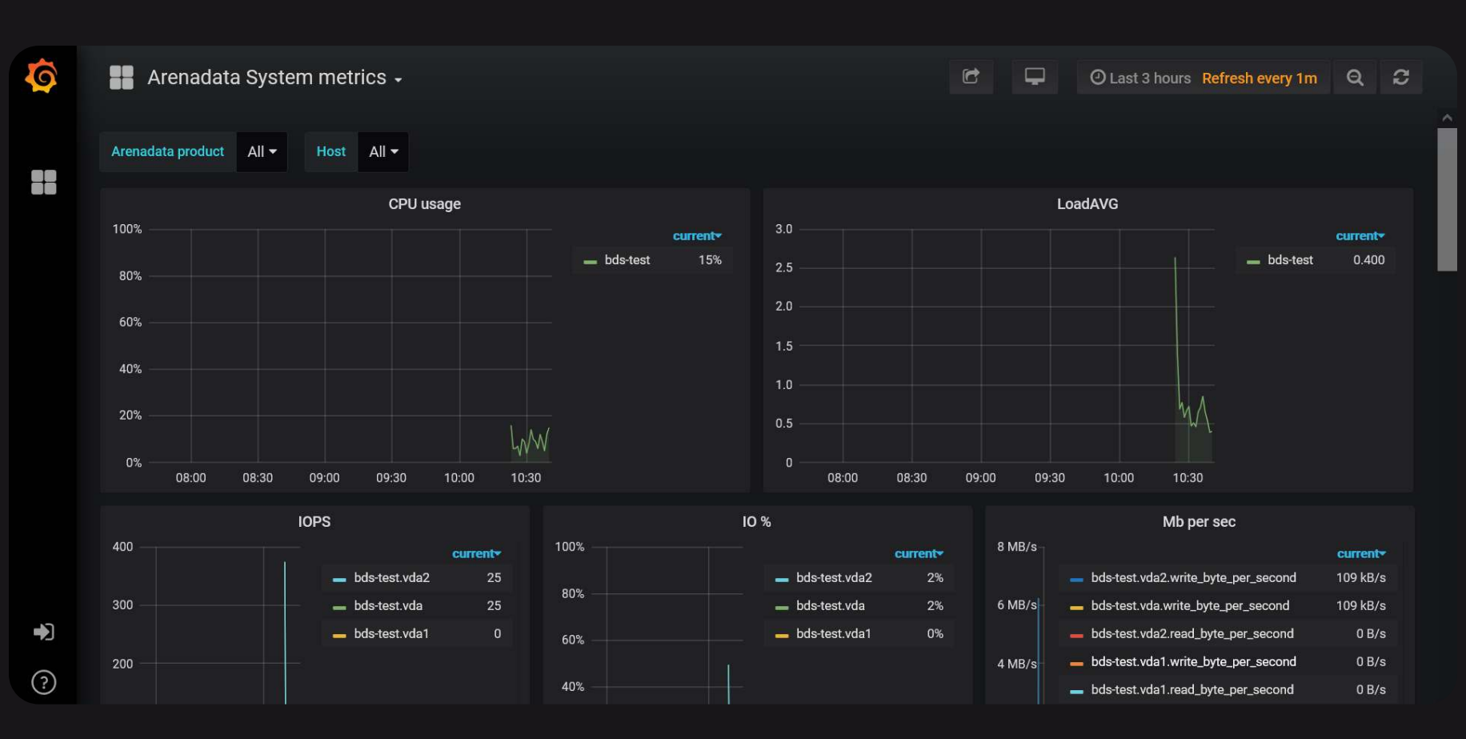
Web-интерфейс Grafana

3. Откройте любой из доступных дашбордов (dashboards). Например, ***Arenadata System metrics***.


Find dashboards by name
Recent
Arenadata System metrics
General
Arenadata System metrics
Graphite Server (Carbon Metrics)
Arenadata
Graphite
Filter by:
Clear
Tags

Переход к метрикам Arenadata System

4. Проанализируйте доступные графики.


Arenadata System metrics
Last 3 hours
Refresh every 1m
Arenadata product
All
Host
All
CPU usage
LoadAVG
IOPS
IO %
Mb per sec
current
bds-test
bds-test.vda2
bds-test.vda
bds-test.vda1

Просмотр метрик Arenadata System

Интеграция с кластером ADH

Евгения Кузина, Дарья Барышева

Содержание

Назад к содержанию

Шаг 1. Импорт настроек мониторинга

Шаг 2. Добавление сервиса мониторинга

Шаг 3. Распределение компонентов мониторинга

Шаг 4. Установка сервиса мониторинга

Шаг 5. Проверка результатов

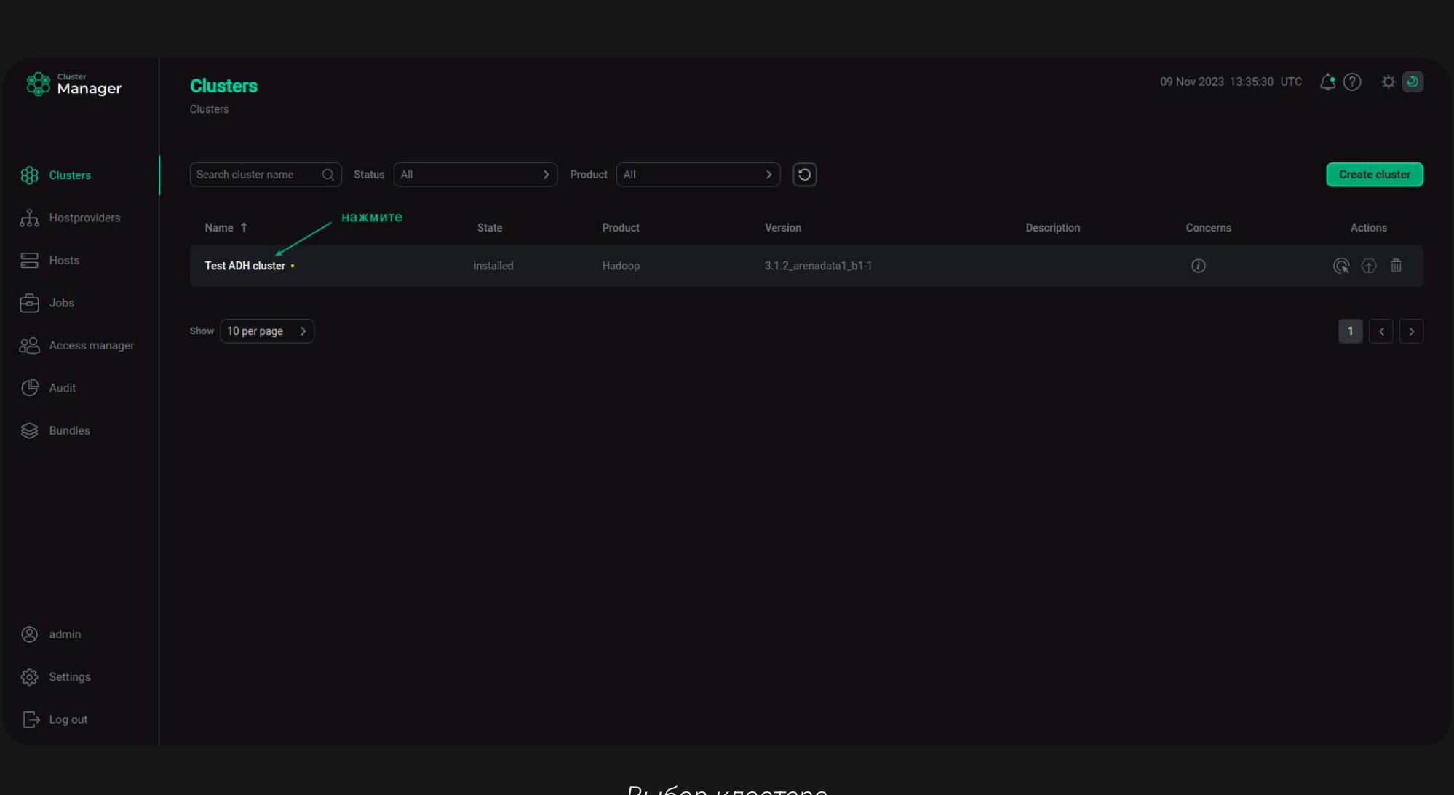
- Graphite
- Grafana

Для настройки мониторинга нового кластера необходимо интегрировать его с предварительно созданным кластером мониторинга и добавить отдельный сервис Monitoring .

Шаги для интеграции кластера мониторинга и кластера ADH описаны ниже. Все действия выполняются на базе кластера ADH.

Шаг 1. Импорт настроек мониторинга

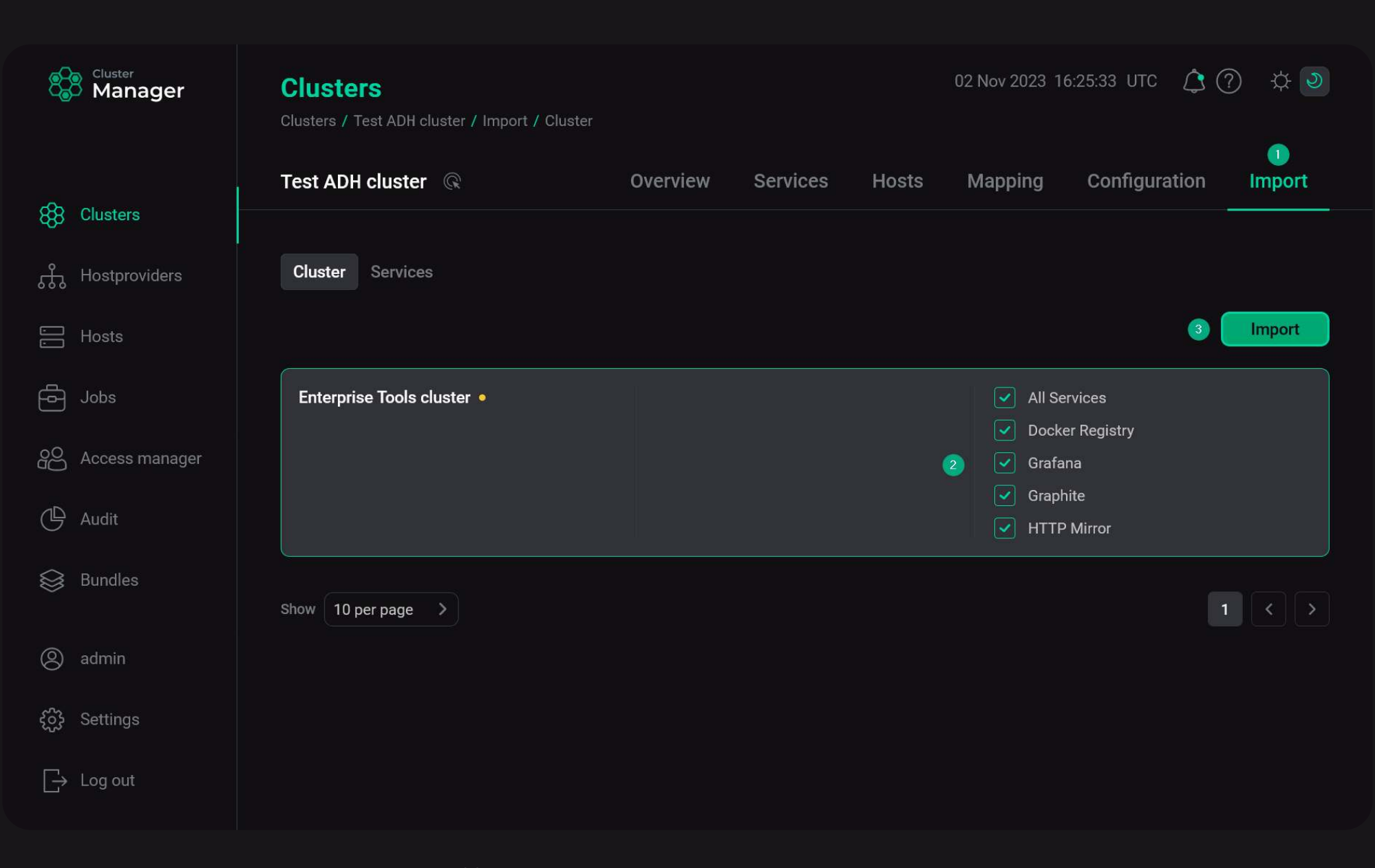
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.



Manager
Clusters
Create cluster
нажмите
Test ADH cluster
installed
Hadoop
3.1.2_arenadata1_b1-1

Выбор кластера

2. Откройте вкладку ***Import*** на странице кластера.
3. Выберите сервисы ***Graphite*** и ***Grafana*** ранее добавленного кластера мониторинга.
4. Нажмите ***Import***.



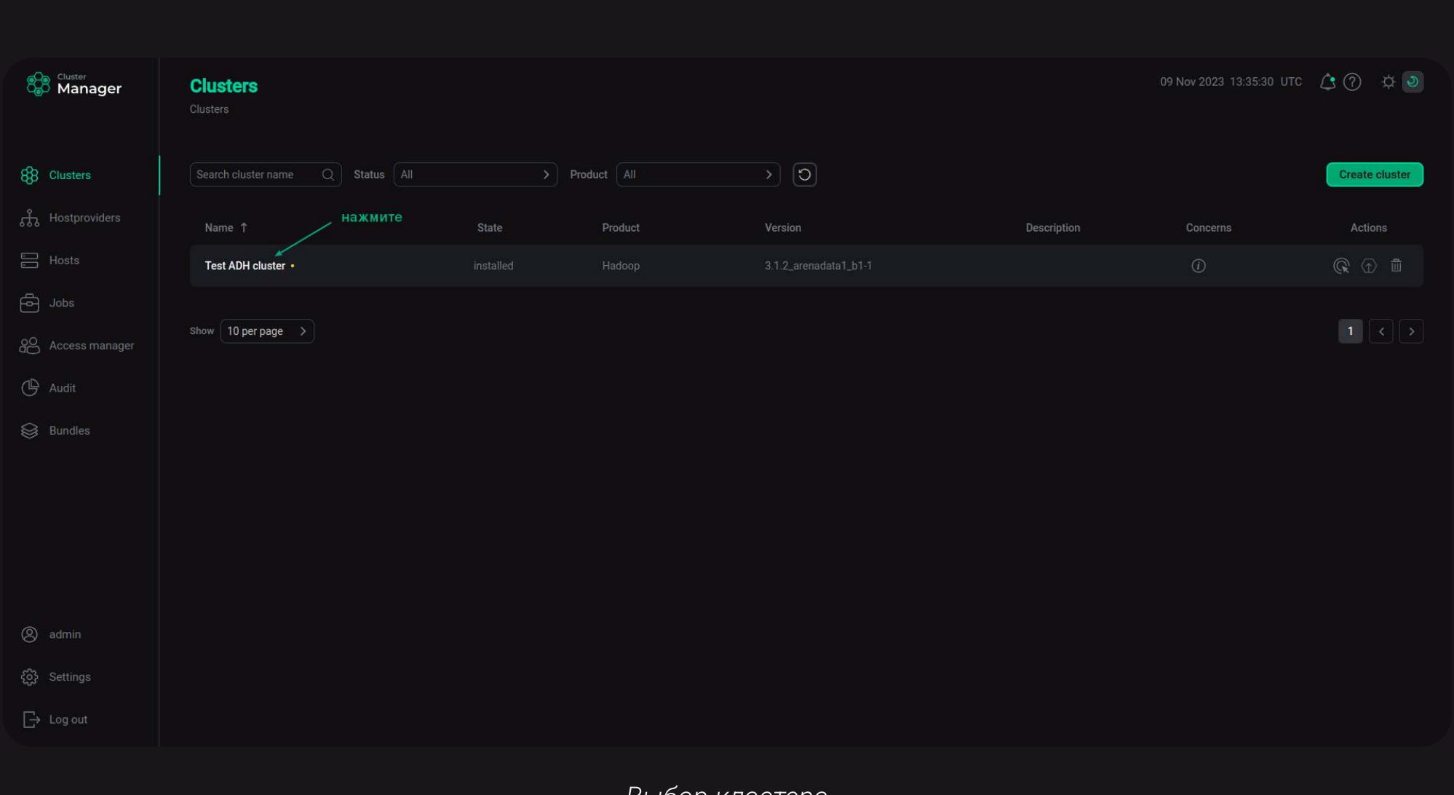
Clusters
Clusters / Test ADH cluster / Import / Cluster
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
1
Import
Cluster
Services
3
Import
Enterprise Tools cluster
2
All Services
Docker Registry
Grafana
Graphite
HTTP Mirror

Интеграция с кластером мониторинга

Шаг 2. Добавление сервиса мониторинга

Для добавления сервисов в кластер:

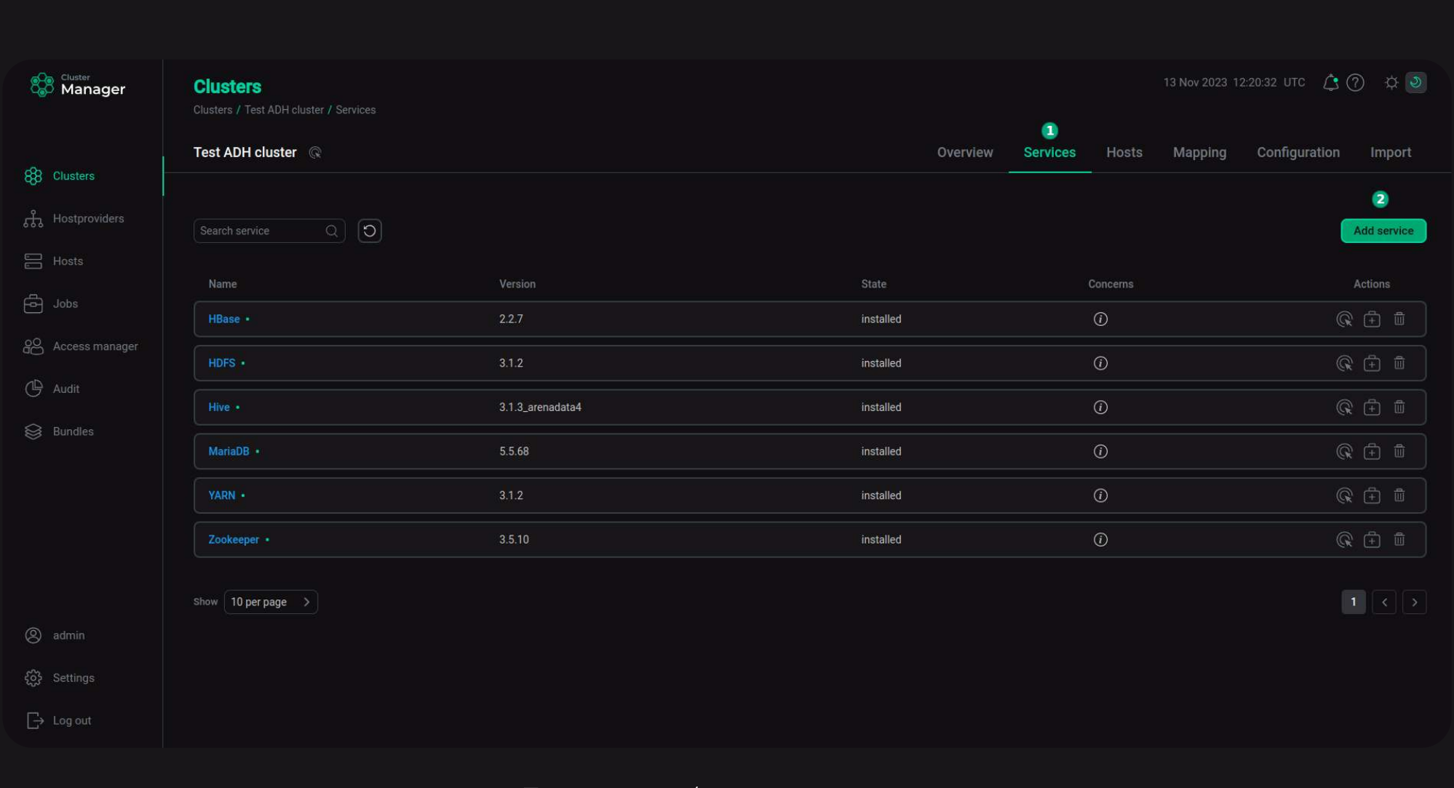
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.



Clusters
Create cluster
нажмите
Test ADH cluster

Выбор кластера

2. Откройте вкладку ***Services*** на странице кластера и нажмите ***Add service***.

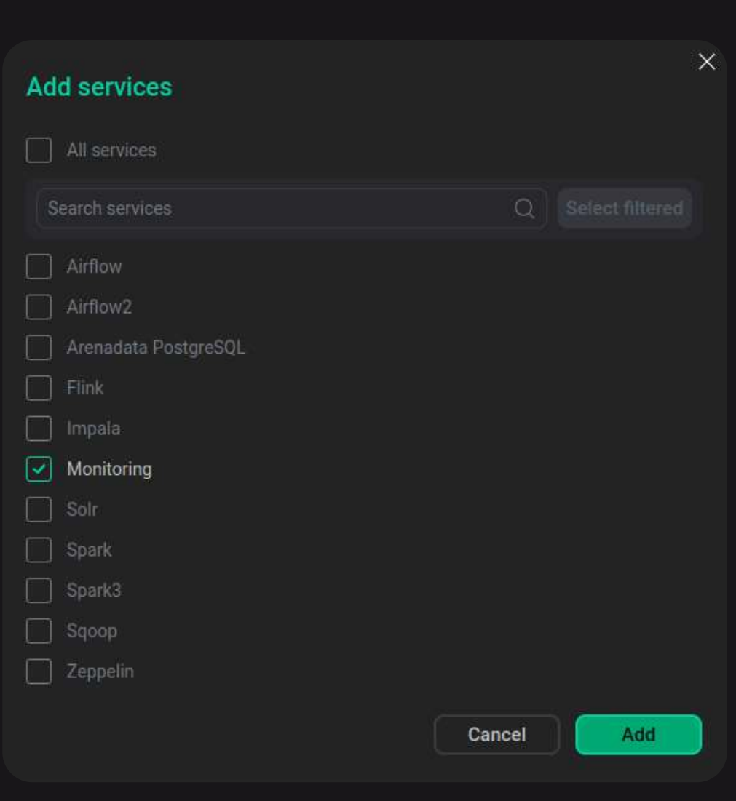


Test ADH cluster
1
Services
2
Add service

Name	Version	State
HBase	2.2.7	installed
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
MariaDB	5.5.68	installed
YARN	3.1.2	installed
Zookeeper	3.5.10	installed

Переход к добавлению сервисов

3. Выберите сервис Monitoring в открывшемся окне и нажмите ***Add***.



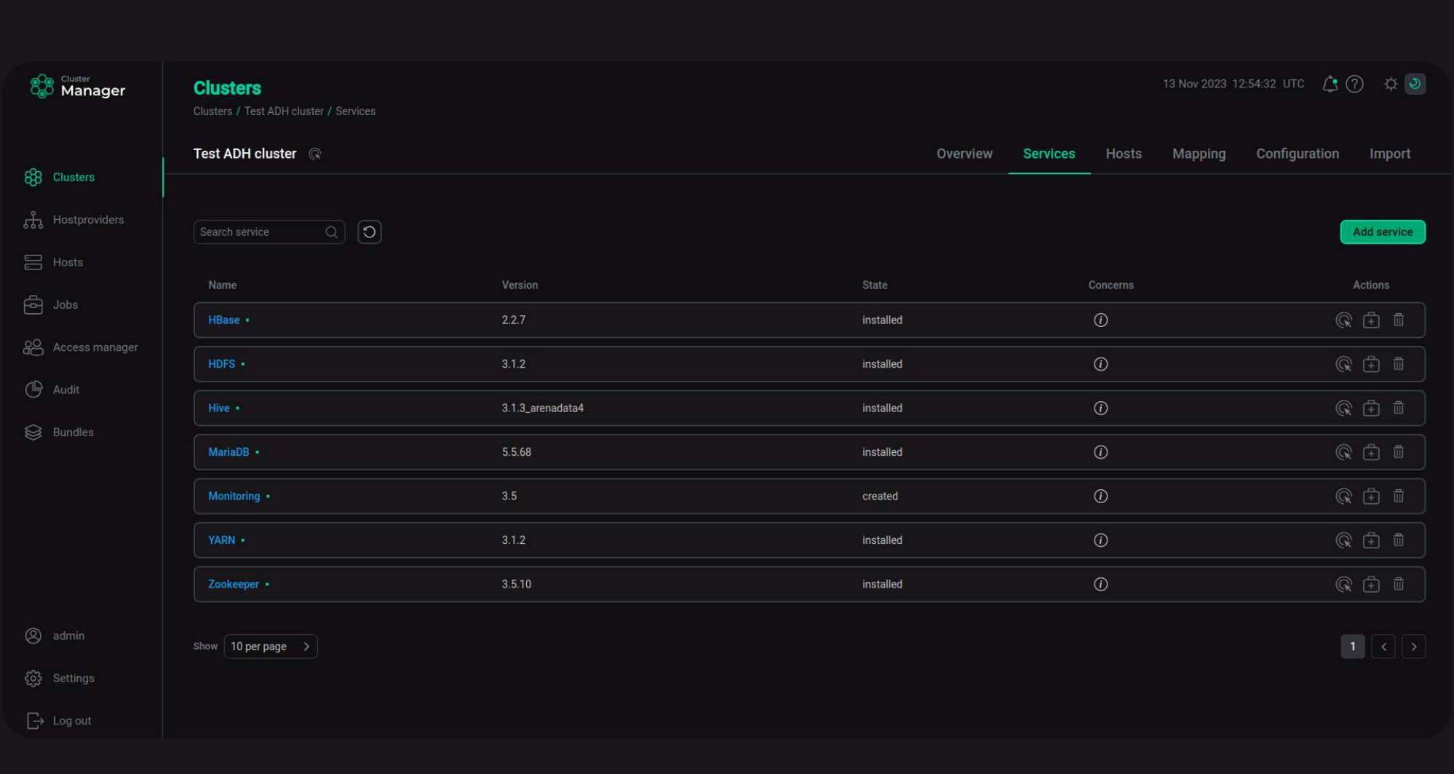
Add services
Search services
Select filtered
Cancel
Add

☐ All services

☐ Airflow
☐ Airflow2
☐ Arenadata PostgreSQL
☐ Flink
☐ Impala
☑ Monitoring
☐ Solr
☐ Spark
☐ Spark3
☐ Sqoop
☐ Zeppelin

Выбор сервисов

4. В результате успешно добавленные сервисы отображаются на вкладке ***Services***.



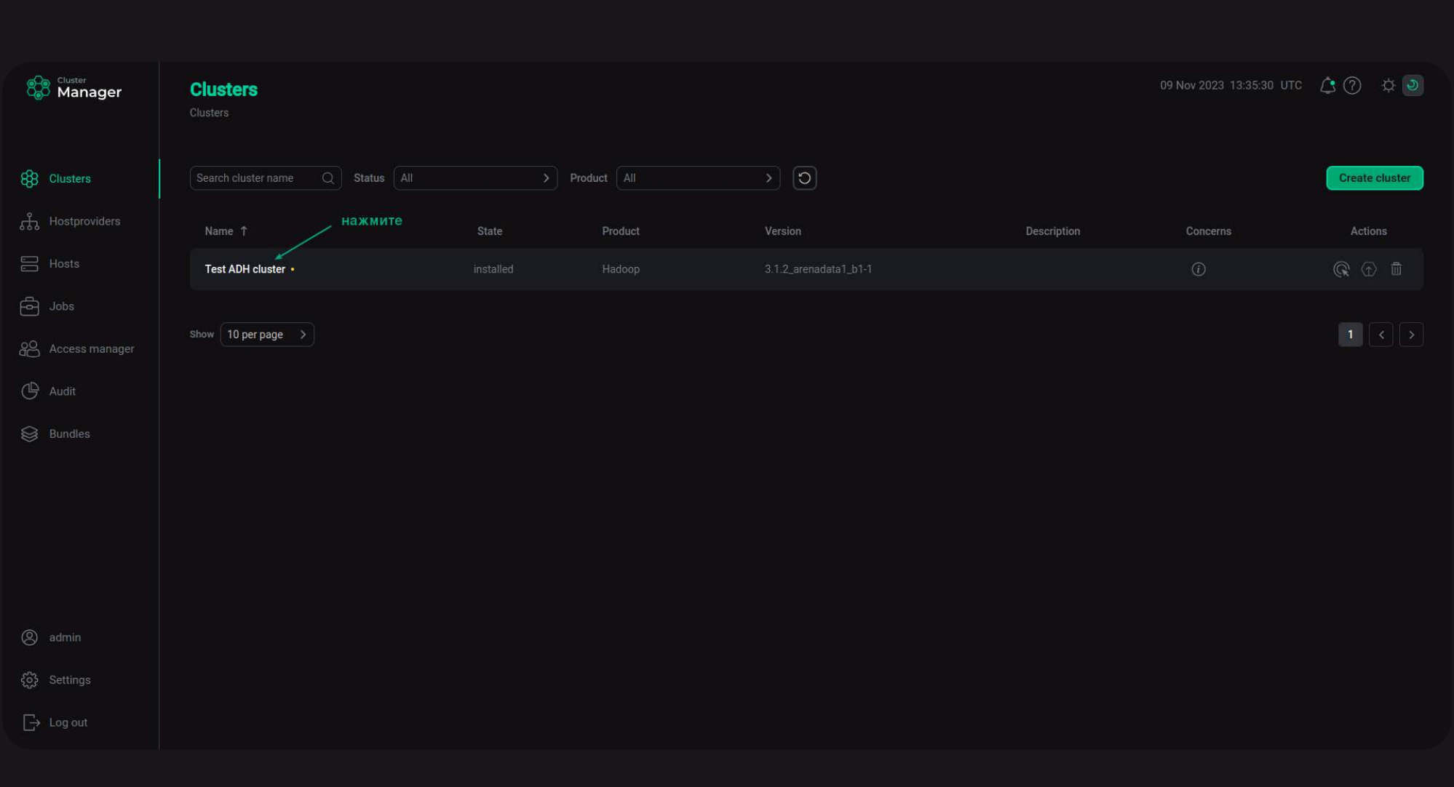
Name	Version	State
HBase	2.2.7	installed
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
MariaDB	5.5.68	installed
Monitoring	3.5	created
YARN	3.1.2	installed
Zookeeper	3.5.10	installed

Результат успешного добавления сервисов в кластер

Шаг 3. Распределение компонентов мониторинга

Для добавления компонентов на хосты кластера:

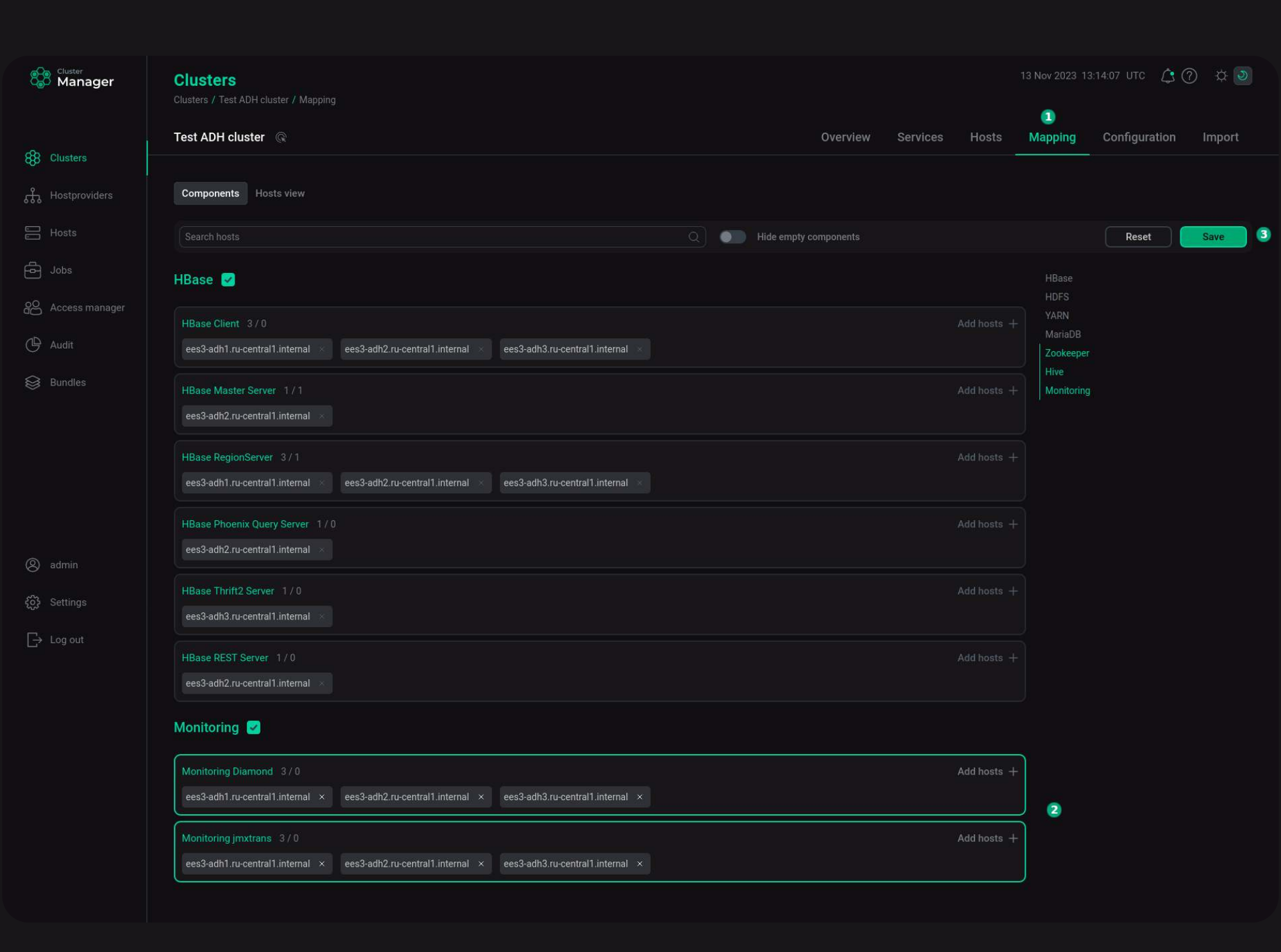
1. Выберите кластер на странице ***Clusters***. Для этого нажмите на имя кластера в столбце ***Name***.



Clusters
нажмите
Test ADH cluster

Выбор кластера

2. Откройте вкладку ***Mapping*** на странице кластера, добавьте компоненты мониторинга на все хосты кластера и нажмите ***Save***.

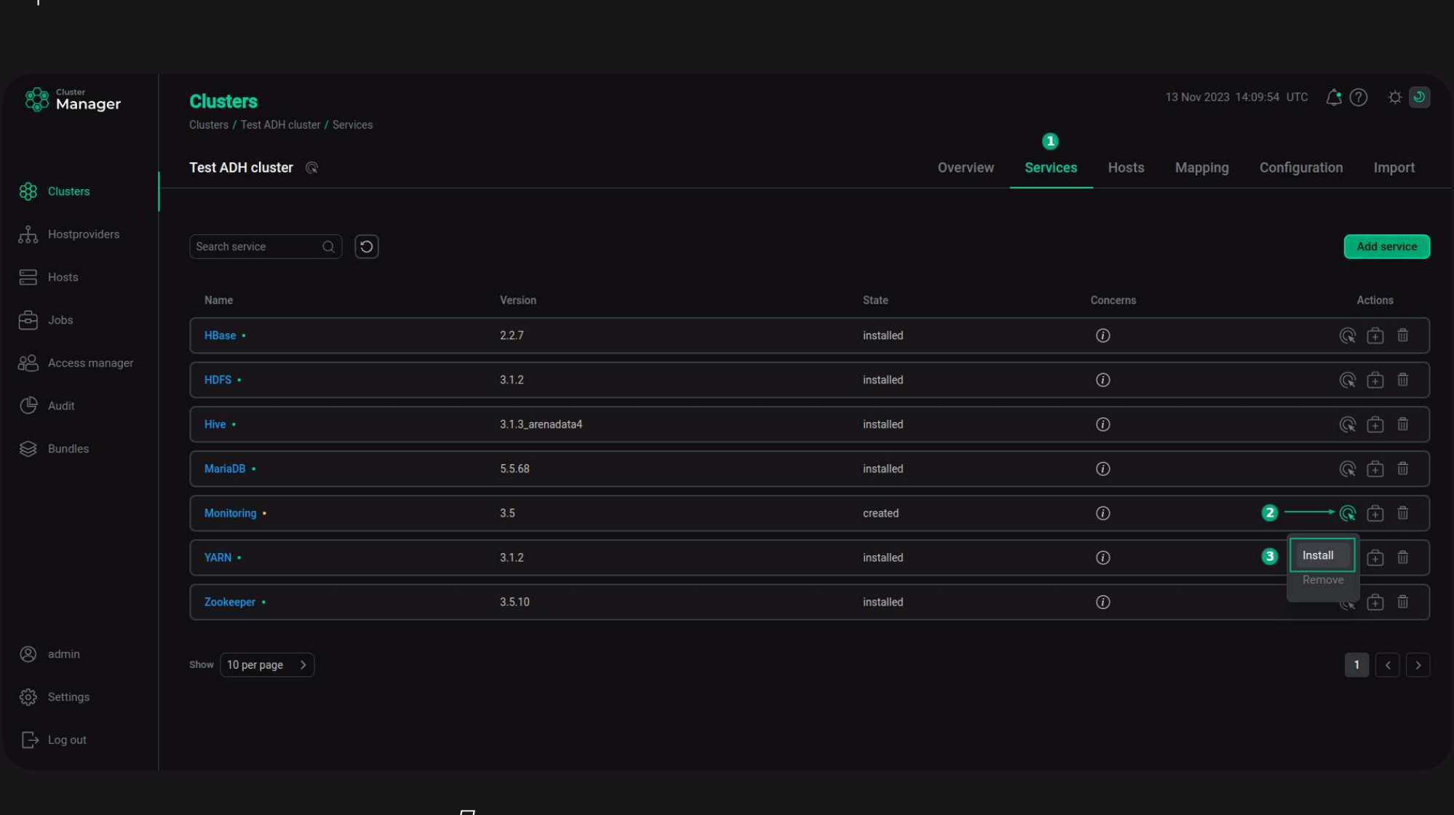


1
Mapping
Components
Hosts view
Reset
Save
3
HBase
Monitoring
Monitoring Diamond 3 / 0
Monitoring jmxtrans 3 / 0
ees3-adh1.ru-central1.internal
ees3-adh2.ru-central1.internal
ees3-adh3.ru-central1.internal
2

Распределение компонентов между хостами

Шаг 4. Установка сервиса мониторинга

1. Откройте вкладку ***Services*** на странице кластера. Нажмите на иконку в столбце ***Actions*** и выберите действие ***Install*** в строке, содержащей сервис Monitoring ; подтвердите действие в открывшемся окне.

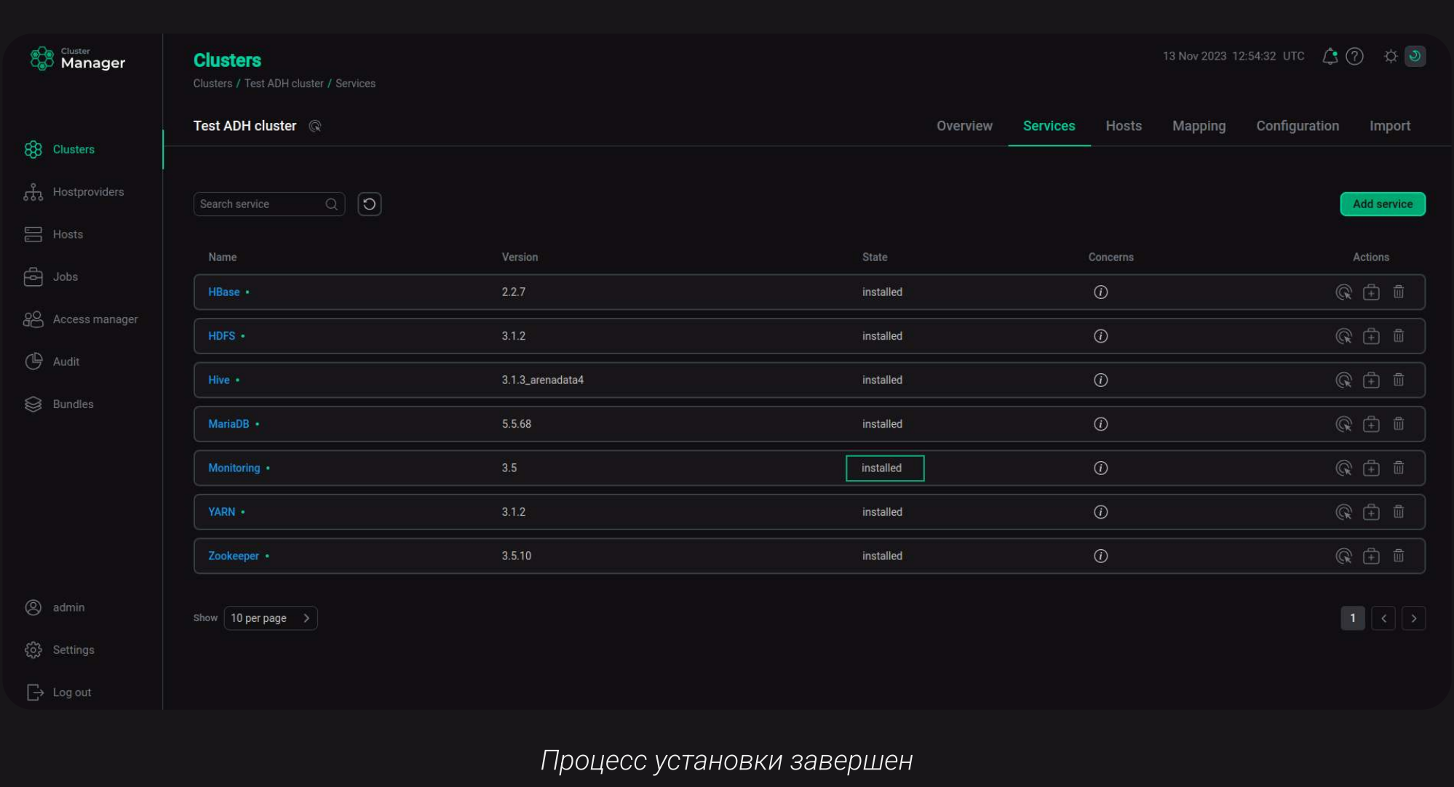


1
Services
2
3
Install
Remove

Переход к установке сервиса мониторинга

2. Дождитесь завершения процесса установки. В результате статус сервиса мониторинга изменяет свое значение с `created` на `installed` .

> **ПРИМЕЧАНИЕ**
>
> В случае ошибок логи можно найти на странице ***Jobs***.



Name	Version	State
HBase	2.2.7	installed
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
MariaDB	5.5.68	installed
Monitoring	3.5	installed
YARN	3.1.2	installed
Zookeeper	3.5.10	installed

Процесс установки завершен

> **ВНИМАНИЕ**
>
> После установки кластера мониторинга необходимо перезапустить кластер ADH, чтобы просматривать метрики.

Шаг 5. Проверка результатов

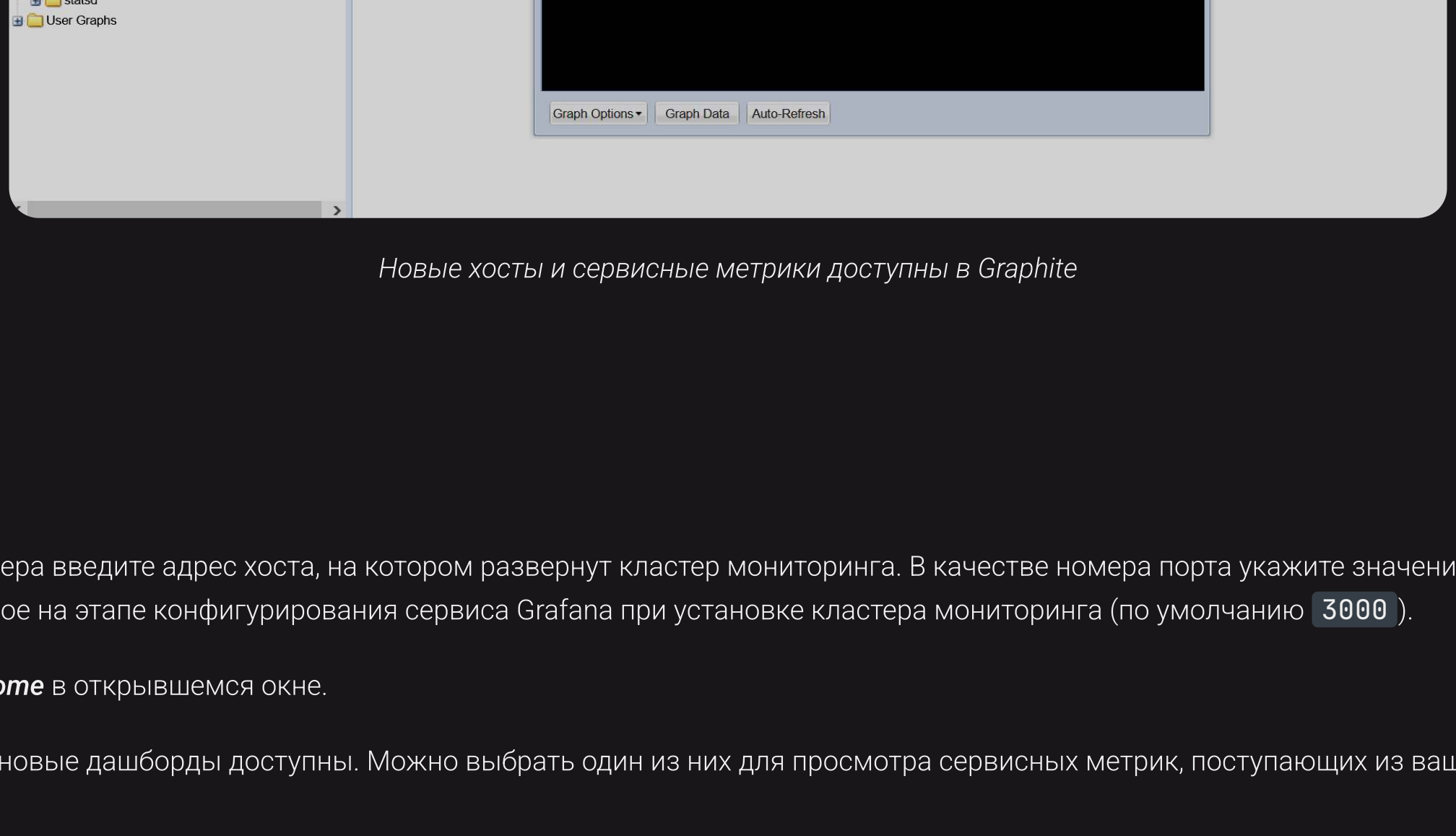
По завершении установки необходимо убедиться в следующем:

- Системные метрики собираются с хостов вашего кластера, а не только с хоста, на котором ранее был развернут кластер мониторинга.
- Наряду с системными метриками, с хостов кластера поступают сервисные метрики. Например, в кластерах ADH поддерживаются метрики для таких сервисов, как HDFS, YARN, HBase, Spark3.

Оба пункта можно проверить в web-интерфейсе Graphite и Grafana.

Graphite

1. В адресной строке браузера введите адрес хоста, на котором развернут кластер мониторинга. В качестве номера порта укажите значение параметра ***Web-interface TCP port*** из раздела ***ip_and_ports***, определенное на этапе конфигурирования сервиса Graphite при установке кластера мониторинга. Значение `80` (используемое по умолчанию) можно не указывать.
2. Разверните дерево в левой части открывшейся формы. Здесь отображаются метрики System и сервисные метрики с группировкой по хостам. Можно выбрать любую из метрик для просмотра динамики ее изменения в виде графика в правой части экрана.

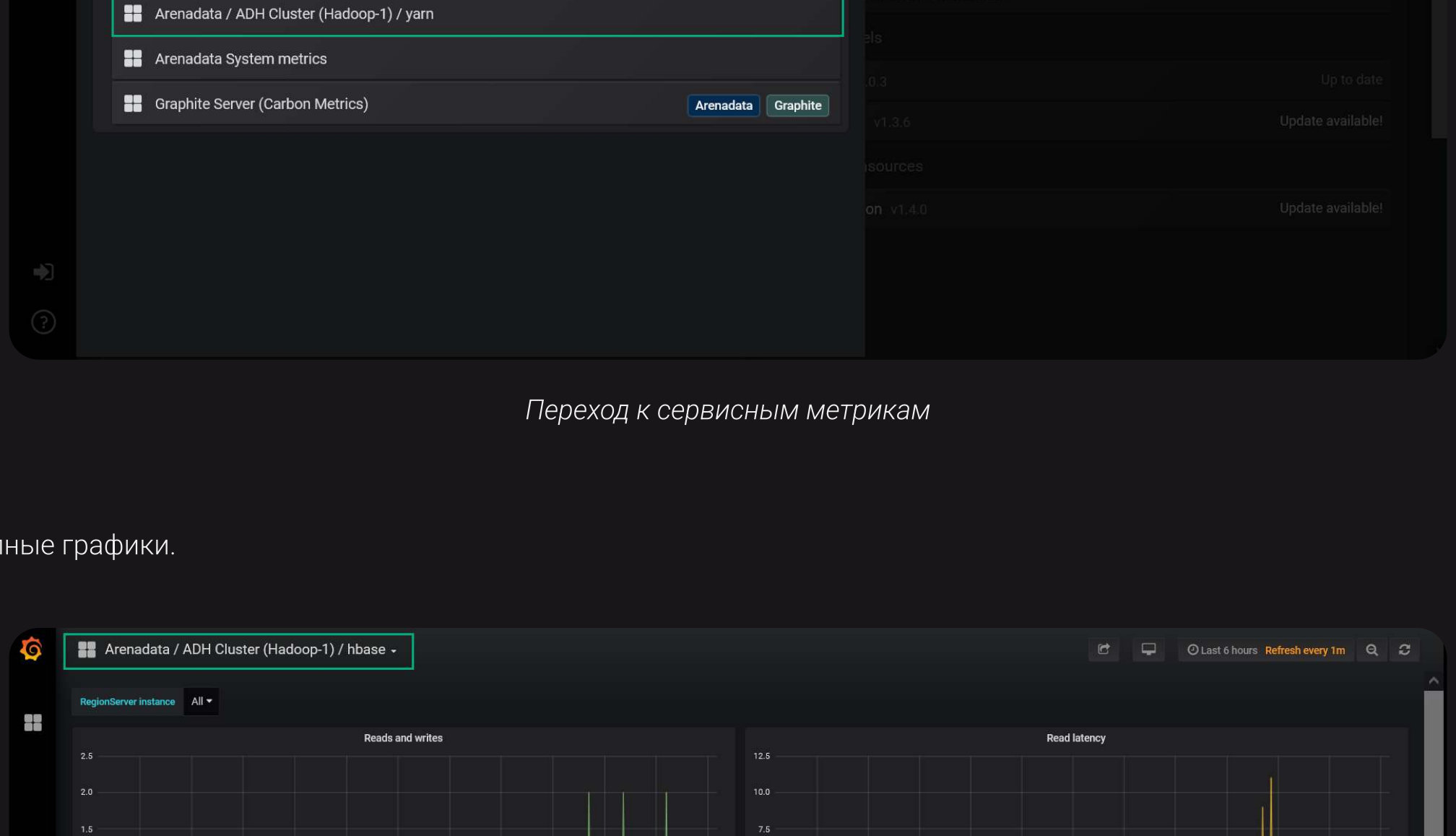


graphite
Documentation
Dashboard
Events
Login
Tree
Search
Auto-Completer
Metrics
Arenadata
Hadoop
Hadoop-1
hbase
regionserver
bds-adh-1_ru-central1_internal_10102
bds-adh-2_ru-central1_internal_10102
bds-adh-3_ru-central1_internal_10102
hdfs
yarn
monitoring_server
carbon
stats
stats_counts
statsd
User Graphs
services metrics are added
Graphite Composer
Now showing the past 24 hours
Graph Options
Graph Data
Auto-Refresh

Новые хосты и сервисные метрики доступны в Graphite

Grafana

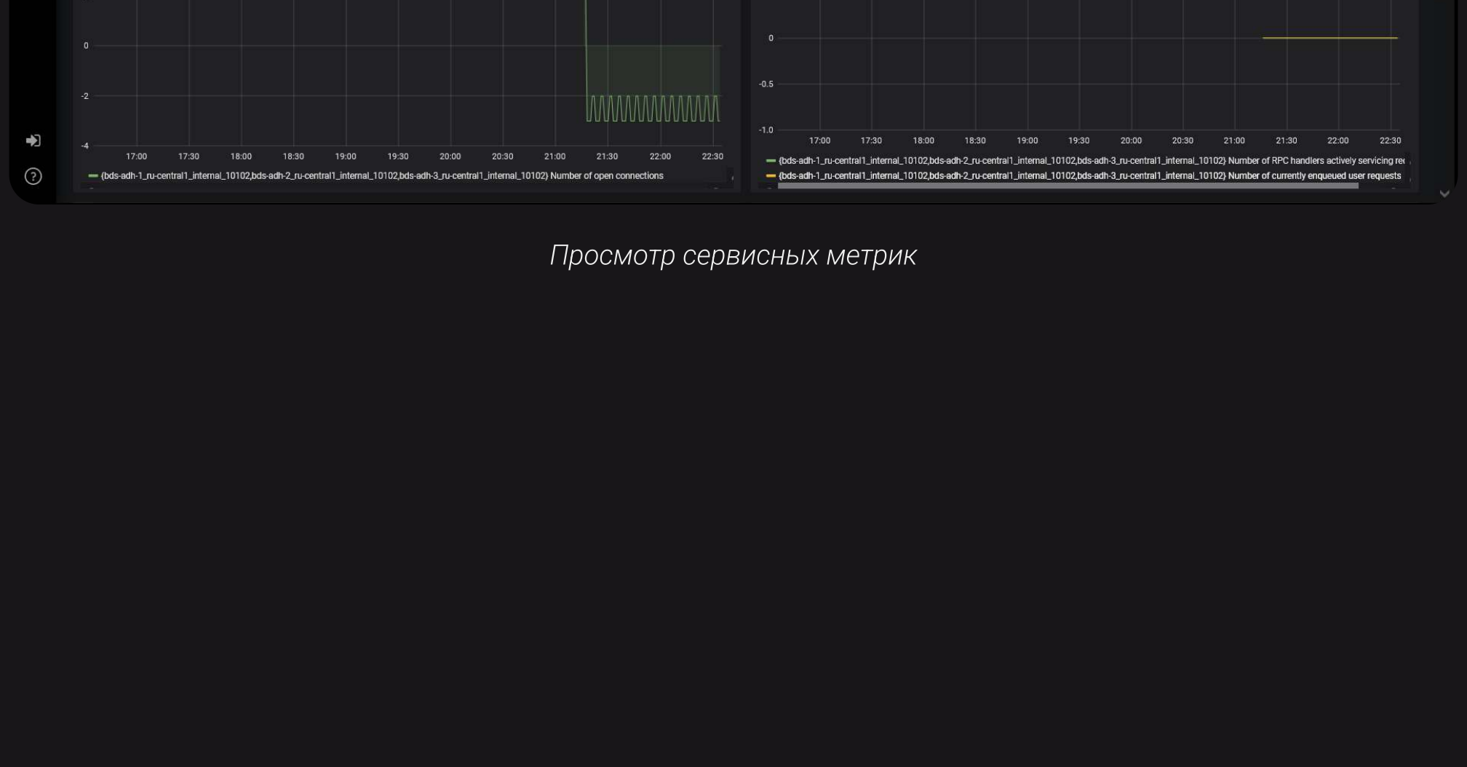
1. В адресной строке браузера введите адрес хоста, на котором развернут кластер мониторинга. В качестве номера порта укажите значение параметра ***Port*** из раздела ***ip_and_ports***, определенное на этапе конфигурирования сервиса Grafana при установке кластера мониторинга (по умолчанию `3000`).
2. Выберите пункт меню ***Home*** в открывшемся окне.
3. Обратите внимание, что новые дашборды доступны. Можно выбрать один из них для просмотра сервисных метрик, поступающих из вашего кластера.



Find dashboards by name
Recent
Arenadata System metrics
General
Arenadata / ADH Cluster (Hadoop-1) / hbase
Arenadata / ADH Cluster (Hadoop-1) / hdfs
Arenadata / ADH Cluster (Hadoop-1) / yarn
Arenadata System metrics
Graphite Server (Carbon Metrics)
Arenadata
Graphite
Filter by:
Clear
Tags

Переход к сервисным метрикам

4. Проанализируйте доступные графики.



Arenadata / ADH Cluster (Hadoop-1) / hbase
Last 6 hours
Refresh every 1m
RegionServer instance
All
Reads and writes
Read latency
Open connections
Request handlers

Просмотр сервисных метрик

Базовые операции

Назад к содержанию

- Airflow
- Flink
- HBase
- HDFS
- Hive
- HUE
- Impala
- Kyuubi
- MapReduce
- Ozone
- Phoenix
- Solr
- Spark
- SSM
- Trino
- YARN
- Zeppelin
- ZooKeeper

Подключение к Airflow через CLI

Елена Костюченко

Содержание

Назад к содержанию

Подключение

Примеры использования

Подключение

Airflow предоставляет интерфейс командной строки, реализованный в виде скрипта *bin/airflow*, который расположен в директории */opt/airflow/bin*.

Хотя рекомендуется управлять Airflow через ADCM или веб-интерфейс Airflow, CLI предоставляет доступ к большему количеству возможностей при работе с Airflow.

Чтобы взаимодействовать с Airflow через CLI, подключитесь к хосту кластера с компонентом Airflow через SSH и запустите скрипт, выполнив:

```
$ sudo /opt/airflow/bin/airflow
```

Пример вывода:

```
Usage: airflow [-h] GROUP_OR_COMMAND ...

Positional Arguments:
  GROUP_OR_COMMAND

    Groups:
      celery         Celery components
      config         View configuration
      connections    Manage connections
      dags           Manage DAGs
      db             Database operations
      jobs           Manage jobs
      kubernetes     Tools to help run the KubernetesExecutor
      pools          Manage pools
      providers      Display providers
      roles          Manage roles
      tasks          Manage tasks
      users          Manage users
      variables      Manage variables

    Commands:
      cheat-sheet    Display cheat sheet
      dag-processor  Start a standalone Dag Processor instance
      info           Show information about current Airflow and environment
      kerberos       Start a kerberos ticket renewer
      plugins        Dump information about loaded plugins
      rotate-fernet-key
                     Rotate encrypted connection credentials and variables
      scheduler      Start a scheduler instance
      standalone     Run an all-in-one copy of Airflow
      sync-perm      Update permissions for existing roles and optionally DAGs
      triggerer      Start a triggerer instance
      version        Show the version
      webserver      Start a Airflow webserver instance

Options:
  -h, --help         show this help message and exit
```

Подробное описание каждой команды доступно на странице Airflow CLI.

Чтобы протестировать работу CLI, запросите версию Airflow:

```
$ sudo /opt/airflow/bin/airflow version
```

Вывод:

```
2.6.3
```

Чтобы увидеть справку по определенной команде, используйте опцию `--help`:

```
$ sudo /opt/airflow/bin/airflow dags --help
```

Команды имеют следующий синтаксис:

```
$ airflow [GROUP_OR_COMMAND] [SUB_COMMAND] [OPTIONS]
```

Здесь:

- `GROUP_OR_COMMAND` — команда Airflow CLI или группа команд (например `dags`, `jobs` и т. д.).
- `SUB_COMMAND` — команда, относящаяся к группе команд, например, `dags list`.
- `OPTIONS` — параметры выбранной команды.

Примеры использования

Airflow предоставляет несколько инструментов, одним из которых является команда `dags`, которая позволяет управлять DAG с помощью CLI.

Например, чтобы получить список существующих DAG, выполните команду:

```
$ sudo /opt/airflow/bin/airflow dags list
```

Пример вывода:

```
dag_id                                                  | filepath
| owner   | paused
========================================================+========================================================================
============+=========+=======
dataset_consumes_1                                      | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_datasets.py | airflow | None
dataset_consumes_1_and_2                                | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_datasets.py | airflow | None
dataset_consumes_1_never_scheduled                      | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_datasets.py | airflow | None
dataset_consumes_unknown_never_scheduled                | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_datasets.py | airflow | None
dataset_produces_1                                      | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_datasets.py | airflow | None
dataset_produces_2                                      | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_datasets.py | airflow | None
example_bash_operator                                   | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_bash_operat | airflow | None
                                                        | or.py
|         |
example_branch_datetime_operator                        | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_branch_date | airflow | None
                                                        | time_operator.py
|         |
example_branch_datetime_operator_2                      | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_branch_date | airflow | None
                                                        | time_operator.py
|         |
example_branch_datetime_operator_3                      | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_branch_date | airflow | None
                                                        | time_operator.py
|         |
example_kubernetes_executor                             | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_kubernetes_ | airflow | None
                                                        | executor.py
|         |
example_local_kubernetes_executor                       | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_local_kuber | airflow | None
                                                        | netes_executor.py
|         |
example_nested_branch_dag                               | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_nested_bran | airflow | None
                                                        | ch_dag.py
|         |
example_params_trigger_ui                               | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/example_params_trig | airflow | None
                                                        | ger_ui.py
|         |
tutorial                                                | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/tutorial.py         | airflow | None
tutorial_dag                                            | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/tutorial_dag.py     | airflow | None
tutorial_taskflow_api                                   | /opt/airflow/lib/python3.10/site-
packages/airflow/example_dags/tutorial_taskflow_a | airflow | None
                                                        | pi.py
|         |
```

Чтобы выгрузить список DAG в файл, используйте опцию `--output` и укажите один из следующих форматов в качестве аргумента: `json`, `yaml`, `plain`, `table`.

Чтобы запустить DAG, используйте команду `trigger`:

```
$ sudo /opt/airflow/bin/airflow dags trigger tutorial
```

Здесь `tutorial` — это ID того DAG, который нужно запустить.

Подключение к Airflow через REST API

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
- Примеры запросов
  - Получение списка пулов
  - Проверка ошибок импорта
  - Получение информации о DAG
  - Запуск DAG
  - Получение списка задач

Обзор

Кроме веб-интерфейсов Airflow также предоставляет доступ к REST API. В этой статье содержится несколько примеров того, как работать с Airflow, используя `curl`. Полный список эндпойнтов доступен в спецификации REST API Airflow.

URL для запросов к API Airflow имеет следующий формат:

http://<server-address>:<port>/api/v1/<endpoint>

Здесь:

- `<server-address>` — IP-адрес или имя хоста сервера Airflow;
- `<port>` — порт сервера Airflow;
- `<endpoint>` — эндпойнт желаемого метода.

Большинство эндпойнтов принимают JSON в качестве входных данных и возвращают JSON в ответе. Добавьте следующий заголовок при выполнении запросов:

```
Content-Type: application/json
```

Пример запроса:

```
$ curl -L 'http://localhost:8080/api/v1/dags' -H 'Content-Type: application/json' --user "admin:admin"
```

Эта команда запрашивает информацию о существующих DAG в формате JSON.

Пример вывода:

```
{
      "dag_id": "example_params_ui_tutorial",
      "default_view": "grid",
      "description": "DAG demonstrating various options for a trigger form generated by DAG params",
      "file_token": ".
eJw9yjEOgCAMAMC_sEs1PohURWwE29AS5fdujJecAxYDpHpmfiHTBtLt4mf1ywxKFifB_cYUdaT4YZEcw4FJBwQrFg2NgjXjSpi9dPcDbOcjlA.
9dagyc8E47YZAC_P1pUK1AnuBgA",
      "fileloc": "/opt/airflow/lib/python3.10/site-packages/airflow/example_dags/example_params_ui_tutorial.py",
      "has_import_errors": false,
      "has_task_concurrency_limits": false,
      "is_active": true,
      "is_paused": true,
      "is_subdag": false,
      "last_expired": null,
      "last_parsed_time": "2024-08-29T13:43:21.591051+00:00",
      "last_pickled": null,
      "max_active_runs": 16,
      "max_active_tasks": 16,
      "next_dagrun": null,
      "next_dagrun_create_after": null,
      "next_dagrun_data_interval_end": null,
      "next_dagrun_data_interval_start": null,
      "owners": [
        "airflow"
      ],
      "pickle_id": null,
      "root_dag_id": null,
      "schedule_interval": null,
      "scheduler_lock": null,
      "tags": [
        {
          "name": "example_ui"
        }
      ],
      "timetable_description": "Never, external triggers only"
    },
```

Примеры запросов

Ниже приводятся примеры запросов к REST API Airflow. В запросах к URL, где присутствует параметр `<dag-id>`, требуется предоставить идентификатор DAG. Укажите идентификатор нужного DAG вместо этого параметра.

Для авторизации используются данные по умолчанию: пользователь `admin` с паролем `admin`.

Получение списка пулов

Чтобы получить список существующих пулов Airflow, выполните запрос `GET` к эндпойнту `pools`.

Пример запроса:

```
$ curl -X GET 'http://localhost:8080/api/v1/pools' \
    -H 'Content-Type: application/json' --user "admin:admin"
```

Ответ сервера:

```
{
  "pools": [
    {
      "description": "Default pool",
      "name": "default_pool",
      "occupied_slots": 0,
      "open_slots": 128,
      "queued_slots": 0,
      "running_slots": 0,
      "scheduled_slots": 7,
      "slots": 128
    }
  ],
  "total_entries": 1
}
```

Проверка ошибок импорта

Чтобы проверить наличие проблем с импортом DAG, отправьте запрос на эндпойнт `importErrors`.

Пример запроса:

```
$ curl --verbose 'http://localhost:8080/api/v1/importErrors' -H 'Content-Type: application/json' --user "admin:admin"
```

Ответ сервера:

```
{
  "import_errors": [
    {
      "filename": "/opt/airflow/dags/second_example.py",
      "import_error_id": 7,
      "stack_trace": "Traceback (most recent call last):\n  File \"<frozen importlib._bootstrap_external>\", line 947, in
source_to_code\n  File \"<frozen importlib._bootstrap>\", line 241, in _call_with_frames_removed\n  File
\"/opt/airflow/dags/second_example.py\", line 1\n    mport pandas as pd\n          ^^^^^^\nSyntaxError: invalid syntax\n",
      "timestamp": "2024-09-03T13:24:12.808983+00:00"
    }
  ],
  "total_entries": 1
}
```

Получение информации о DAG

Чтобы получить информацию о конкретном DAG, отправьте запрос `GET` с идентификатором нужного DAG на эндпойнт `details`.

Пример запроса:

```
$ curl -v -X GET http://localhost:8080/api/v1/dags/<dag-id>/details \
  -H 'Content-Type: application/json' \
  --user "admin:admin"
```

Ответ сервера:

```
{
  "catchup": false,
  "concurrency": 16,
  "dag_id": "tutorial_dag",
  "dag_run_timeout": null,
  "default_view": "grid",
  "description": "DAG tutorial",
  "doc_md": "\n### DAG Tutorial Documentation\nThis DAG is demonstrating an Extract -> Transform -> Load pipeline\n",
  "end_date": null,
  "file_token": ".eJw9ysEOQDAMANB_2d1KfJAUtTVKm63C_l5cHF_yAqg5IJdN9AbhGax51nOMQw-VnTrDZcdE9U_04GFC04qpgl-uhVE-RWvhBZv_HbQ.
xzpxxi3Ra3GD87EJmfl6rDYC8ME",
  "fileloc": "/opt/airflow/lib/python3.10/site-packages/airflow/example_dags/tutorial_dag.py",
  "is_active": true,
  "is_paused": true,
  "is_paused_upon_creation": null,
  "is_subdag": false,
  "last_parsed": "2024-09-03T13:20:29.979033+00:00",
  "max_active_runs": 16,
  "max_active_tasks": 16,
  "orientation": "LR",
  "owners": [
    "airflow"
  ],
  "params": {},
  "pickle_id": null,
  "render_template_as_native_obj": false,
  "schedule_interval": null,
  "start_date": "2021-01-01T00:00:00+00:00",
  "tags": [
    {
      "name": "example"
    }
  ],
  "timezone": "Timezone('UTC')"
}
```

Запуск DAG

Чтобы запустить DAG, отправьте запрос `POST`, содержащий параметры запуска DAG, на эндпойнт `dagRuns`.

Пример запроса:

```
$ curl -v -X POST http://localhost:8080/api/v1/dags/<dag-id>/dagRuns \
  -H 'Content-Type: application/json' \
  -d '<json>' \
  --user "admin:admin"
```

Здесь `<json>` — параметры запуска DAG в формате JSON.

Пример параметров:

```
{
  "dag_run_id": "string",
  "logical_date": "2019-08-24T14:15:22Z",
  "execution_date": "2019-08-24T14:15:22Z",
  "conf": {},
  "note": "string"
}
```

Чтобы установить конкретную дату выполнения (`logical_date`) для DAG, передайте временную метку в формате `YYYY-MM-DDTHH:mm:S`.

Ответ сервера:

```
{
  "conf": {},
  "dag_id": "tutorial_dag",
  "dag_run_id": "string",
  "data_interval_end": "2019-08-24T14:15:22+00:00",
  "data_interval_start": "2019-08-24T14:15:22+00:00",
  "end_date": null,
  "execution_date": "2019-08-24T14:15:22+00:00",
  "external_trigger": true,
  "last_scheduling_decision": null,
  "logical_date": "2019-08-24T14:15:22+00:00",
  "note": "string",
  "run_type": "manual",
  "start_date": null,
  "state": "queued"
}
```

Получение списка задач

Чтобы просмотреть список задач определенного DAG, отправьте запрос `GET` на эндпойнт `tasks` этого DAG.

Пример запроса:

```
$ curl -v -X GET http://localhost:8080/api/v1/dags/<dag-id>/tasks  \
      -H 'Content-Type: application/json'  \
      --user "admin:admin"
```

Ответ сервера

```
{
  "tasks": [
    {
      "class_ref": {
        "class_name": "PythonOperator",
        "module_path": "airflow.operators.python"
      },
      "depends_on_past": false,
      "downstream_task_ids": [
        "transform_people"
      ],
      "end_date": null,
      "execution_timeout": null,
      "extra_links": [],
      "is_mapped": false,
      "operator_name": "PythonOperator",
      "owner": "airflow",
      "params": {},
      "pool": "default_pool",
      "pool_slots": 1.0,
      "priority_weight": 1.0,
      "queue": "default",
      "retries": 0.0,
      "retry_delay": {
        "__type": "TimeDelta",
        "days": 0,
        "microseconds": 0,
        "seconds": 300
      },
      "retry_exponential_backoff": false,
      "start_date": "2021-12-17T00:00:00+00:00",
      "task_id": "extract_people",
      "template_fields": [
        "templates_dict",
        "op_args",
        "op_kwargs"
      ],
      "trigger_rule": "all_success",
      "ui_color": "#ffefeb",
      "ui_fgcolor": "#000",
      "wait_for_downstream": false,
      "weight_rule": "downstream"
    },
    {
      "class_ref": {
        "class_name": "PythonOperator",
        "module_path": "airflow.operators.python"
      },
      "depends_on_past": false,
      "downstream_task_ids": [],
      "end_date": null,
      "execution_timeout": null,
      "extra_links": [],
      "is_mapped": false,
      "operator_name": "PythonOperator",
      "owner": "airflow",
      "params": {},
      "pool": "default_pool",
      "pool_slots": 1.0,
      "priority_weight": 1.0,
      "queue": "default",
      "retries": 0.0,
      "retry_delay": {
        "__type": "TimeDelta",
        "days": 0,
        "microseconds": 0,
        "seconds": 300
      },
      "retry_exponential_backoff": false,
      "start_date": "2021-12-17T00:00:00+00:00",
      "task_id": "load_people",
      "template_fields": [
        "templates_dict",
        "op_args",
        "op_kwargs"
      ],
      "trigger_rule": "all_success",
      "ui_color": "#ffefeb",
      "ui_fgcolor": "#000",
      "wait_for_downstream": false,
      "weight_rule": "downstream"
    },
    {
      "class_ref": {
        "class_name": "PythonOperator",
        "module_path": "airflow.operators.python"
      },
      "depends_on_past": false,
      "downstream_task_ids": [
        "load_people"
      ],
      "end_date": null,
      "execution_timeout": null,
      "extra_links": [],
      "is_mapped": false,
      "operator_name": "PythonOperator",
      "owner": "airflow",
      "params": {},
      "pool": "default_pool",
      "pool_slots": 1.0,
      "priority_weight": 1.0,
      "queue": "default",
      "retries": 0.0,
      "retry_delay": {
        "__type": "TimeDelta",
        "days": 0,
        "microseconds": 0,
        "seconds": 300
      },
      "retry_exponential_backoff": false,
      "start_date": "2021-12-17T00:00:00+00:00",
      "task_id": "transform_people",
      "template_fields": [
        "templates_dict",
        "op_args",
        "op_kwargs"
      ],
      "trigger_rule": "all_success",
      "ui_color": "#ffefeb",
      "ui_fgcolor": "#000",
      "wait_for_downstream": false,
      "weight_rule": "downstream"
    }
  ],
  "total_entries": 3
}
```

Обзор UI в Airflow

Елена Костюченко

Содержание

Назад к содержанию

- Airflow UI
  - DAGs
  - Datasets
  - Security
  - Browse
  - Admin
  - Docs
- Flower UI

По умолчанию Airflow предоставляет два вида пользовательского Web-интерфейса:

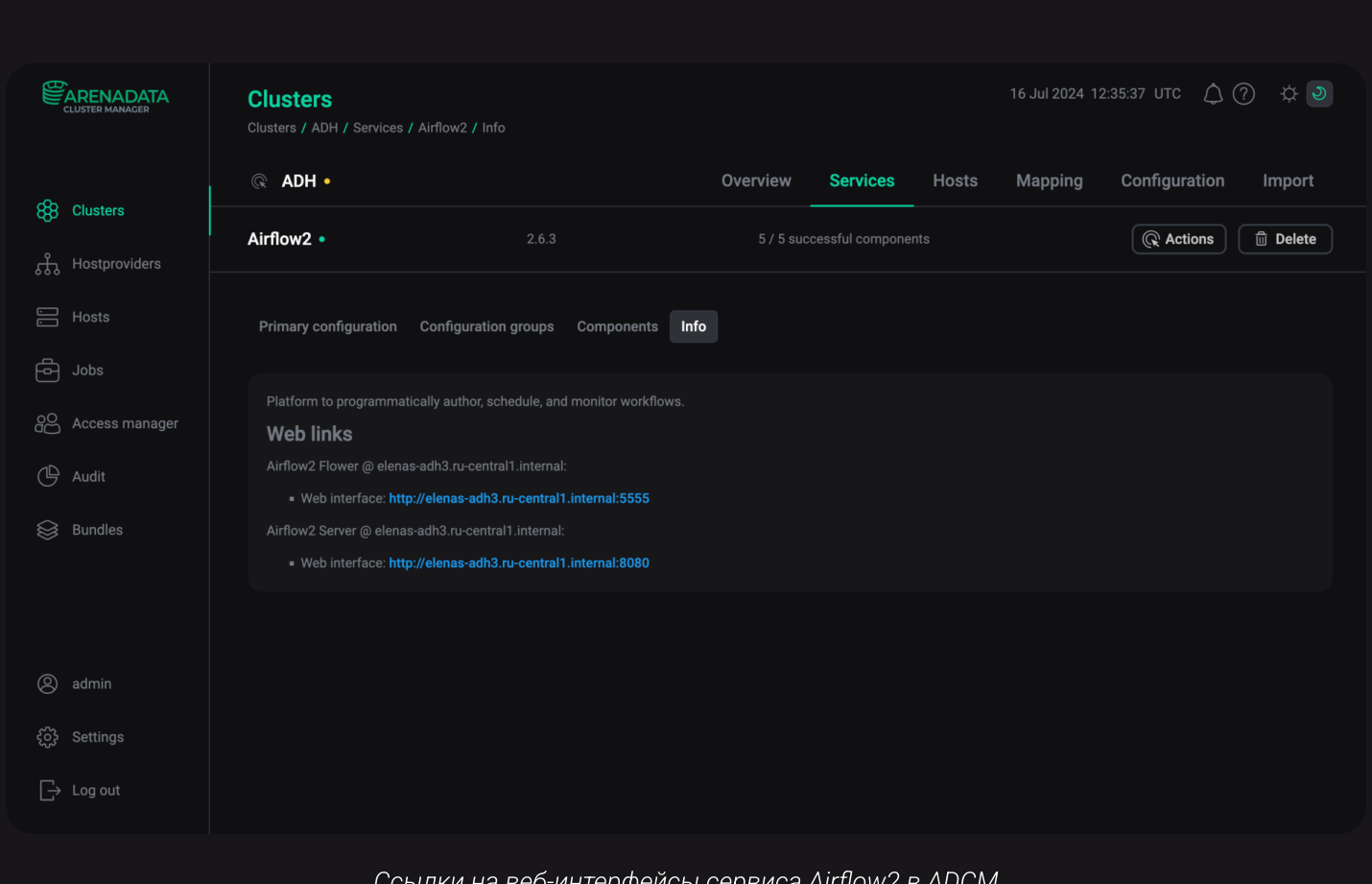
- **Airflow Web UI**. Данный интерфейс позволяет контролировать управляющие графы (DAGs) и задачи (tasks), отслеживать выполнение задач, настраивать параметры базы метаданных (например, соединение с внешними источниками) и выполнять другие операции.
- **Flower**. Это отдельный инструмент для наблюдения за кластерами Celery и управления ими.

Адреса веб-интерфейсов Airflow2

Интерфейс	URL	Документация
Airflow Web UI	http://<Airflow Server IP>:8080	Link
Flower	http://<Airflow Server IP>:5555	Link

При настроенном SSL-сертификате используйте `https://` вместо `http://`.

Ссылки на оба веб-интерфейса доступны на вкладке ***Info*** на странице сервиса в ADCM.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH / Services / Airflow2 / Info
16 Jul 2024 12:35:37 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
Airflow2
2.6.3
5 / 5 successful components
Actions
Delete
Primary configuration
Configuration groups
Components
Info

Platform to programmatically author, schedule, and monitor workflows.

Web links

Airflow2 Flower @ elenas-adh3.ru-central1.internal:
- Web interface: http://elenas-adh3.ru-central1.internal:5555

Airflow2 Server @ elenas-adh3.ru-central1.internal:
- Web interface: http://elenas-adh3.ru-central1.internal:8080

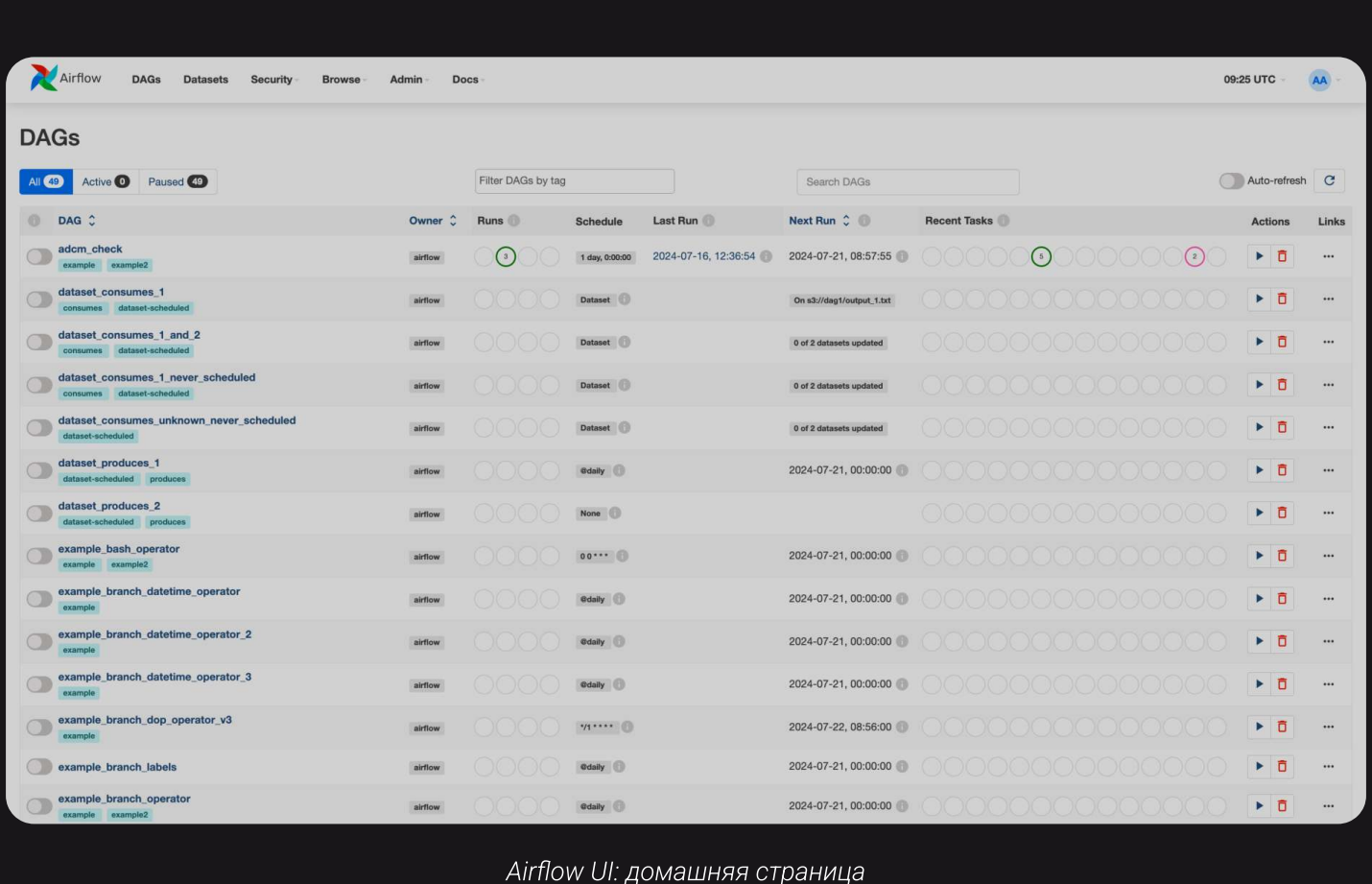
Ссылки на веб-интерфейсы сервиса Airflow2 в ADCM

Airflow UI

DAGs

Страница ***DAGs*** — это домашняя страница пользовательского интерфейса Airflow. На ней отображается список пайплайнов данных (data pipeline). В этом списке содержится информация о пайплайнах, элементы управления ими и ссылки на полезные страницы:

- ***DAG*** — уникальное имя идентификатора DAG. Чтобы приостановить работу DAG, используйте переключатель напротив его имени. При нажатии на имя, откроется страница этого DAG.
- ***Owner*** — владелец DAG, как указано в коде. При нажатии на имя владельца список будет отфильтрован так, чтобы показывались только те DAG, которые принадлежат этому владельцу.
- ***Runs*** — статусы всех предыдущих запусков этого DAG: поставлен в очередь (queued), успешен (success), выполняется (running), сбой (failed).
- ***Schedule*** — указывает, как часто запускается DAG. Он может запускаться вручную или по расписанию.
- ***Last Run*** — дата и время последнего запуска DAG. При нажатии на дату откроется страница этого DAG.
- ***Next Run*** — дата и время следующего запуска DAG. Для групп DAG, запущенных набором данных (dataset), отображается число обновлений с момента последнего запуска DAG.
- ***Recent Tasks*** — статусы текущего или последнего запуска DAG.
- ***Action*** — с помощью действий можно повторить последний запуск DAG, запустить его с определенной конфигурацией или удалить. Действие удаления не удаляет сам DAG, а только связанные с ним метаданные.
- ***Links*** — в меню доступны ссылки на определенные вкладки страницы DAG.

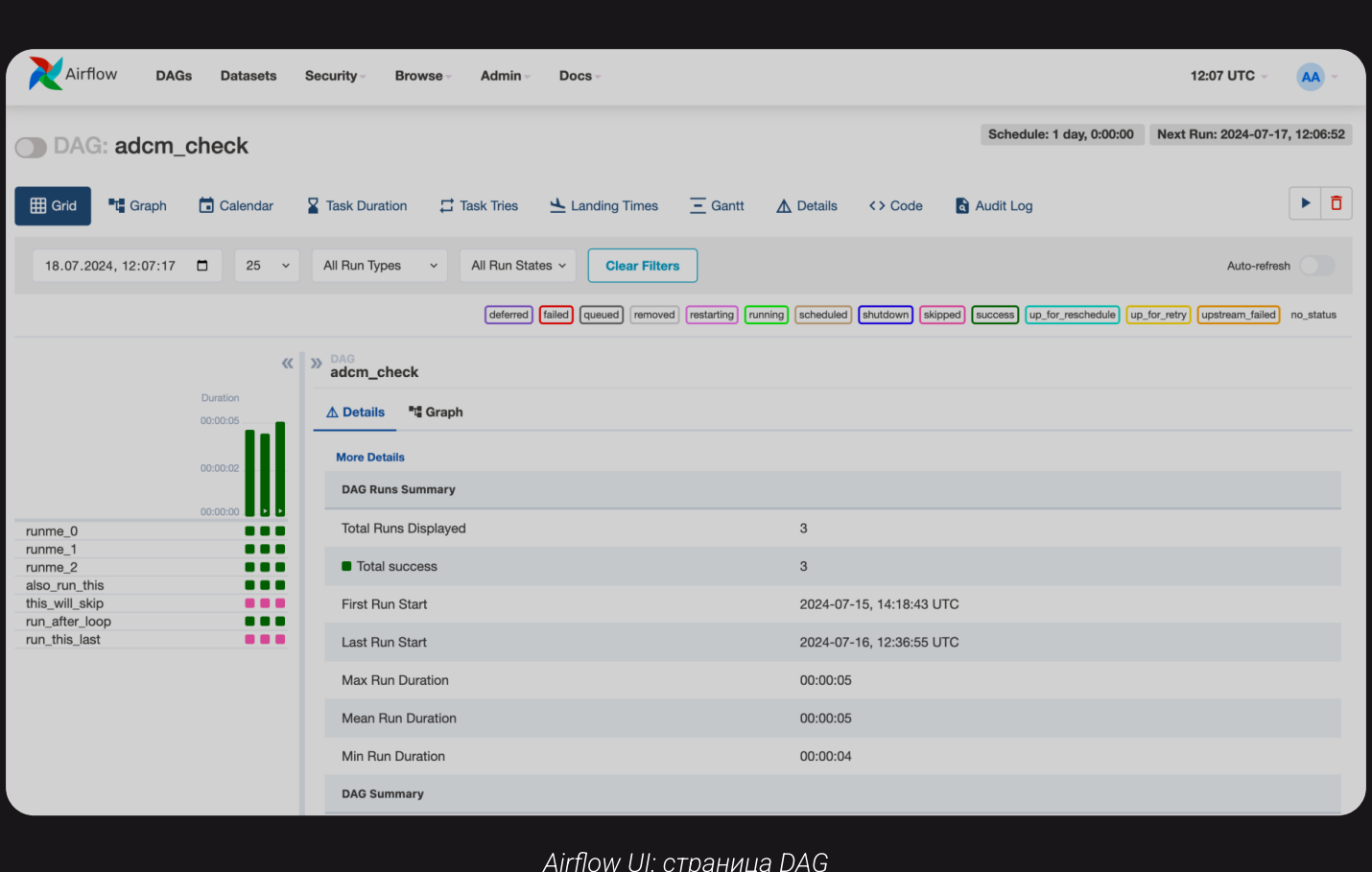

Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
09:25 UTC
DAGs
All
Active
Paused
Filter DAGs by tag
Search DAGs
Auto-refresh
DAG
Owner
Runs
Schedule
Last Run
Next Run
Recent Tasks
Actions
Links

Airflow UI: домашняя страница

Также вы можете фильтровать список DAG по тегу или использовать поиск.

На странице DAG доступно несколько представлений информации:

- ***Grid*** — диаграмма запусков DAG по продолжительности, статусу задач (task) и общая информация о запусках. Задачи представлены в виде квадратов, а DAG — в виде столбцов. Чтобы увидеть информацию по конкретной задаче, нажмите на соответствующий квадрат. Там же доступна информация о Xcom, логах и действиях, связанных с этой задачей.
- ***Graph*** — график задач с информацией о запусках.
- ***Calendar*** — календарь запущенных и запланированных DAG. Каждая дата функционирует как индикатор успешности запусков DAG в этот день.
- ***Task Duration*** — график выполнения каждой задачи.
- ***Task Tries*** — график попыток запуска каждой задачи.
- ***Landing Times*** — график фактического времени выполнения каждой задачи по сравнению с запланированным.
- ***Gantt*** — диаграмма Ганта для запланированного и фактического времени выполнения задач.
- ***Details*** — подробная информация о конкретном DAG.
- ***Code*** — код самого DAG.
- ***Audit Log*** — в представлении отображаются выбранные события и операции, выполненные в этот день. Вы можете выбрать, какие события включать в лог в конфигурации Airflow.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:07 UTC
Grid
Graph
Calendar
Task Duration
Task Tries
Landing Times
Gantt
Details
Code
Audit Log
All Run Types
All Run States
Clear Filters
Auto-refresh
deferred
failed
queued
removed
restarting
running
scheduled
shutdown
skipped
success
up_for_reschedule
up_for_retry
upstream_failed
no_status
Duration
runme_0
runme_1
runme_2
also_run_this
this_will_skip
run_after_loop
run_this_last
Details
Graph
More Details

DAG: adcm_check

Schedule: 1 day, 0:00:00

Next Run: 2024-07-17, 12:06:52

18.07.2024, 12:07:17

25

DAG Runs Summary

Total Runs Displayed	3
Total success	3
First Run Start	2024-07-15, 14:18:43 UTC
Last Run Start	2024-07-16, 12:36:55 UTC
Max Run Duration	00:00:05
Mean Run Duration	00:00:05
Min Run Duration	00:00:04

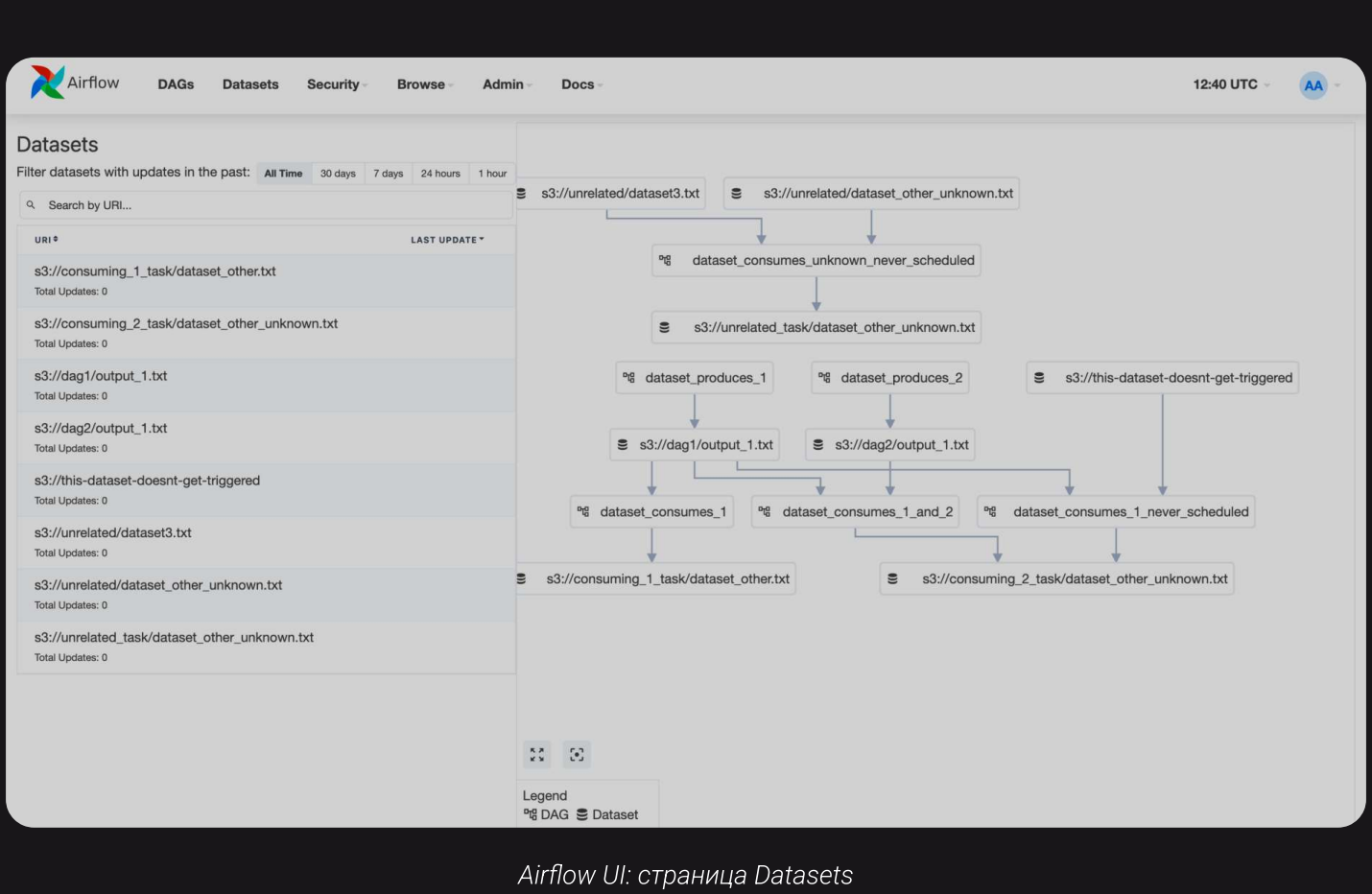
DAG Summary

Airflow UI: страница DAG

Datasets

На странице ***Datasets*** отображается список существующих наборов данных и графическое представление связей между наборами данных и группами DAG. При нажатии на набор данных в списке или на графике будут подсвечены все его связи.

На странице предусмотрена возможность фильтровать список для отображения недавней истории и поиска определенного набора данных.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:40 UTC
All Time
30 days
7 days
24 hours
1 hour
Search by URI...
s3://unrelated/dataset3.txt
s3://unrelated/dataset_other_unknown.txt
dataset_consumes_unknown_never_scheduled
s3://unrelated_task/dataset_other_unknown.txt
dataset_produces_1
dataset_produces_2
s3://this-dataset-doesnt-get-triggered
s3://dag1/output_1.txt
s3://dag2/output_1.txt
dataset_consumes_1
dataset_consumes_1_and_2
dataset_consumes_1_never_scheduled
s3://consuming_1_task/dataset_other.txt
s3://consuming_2_task/dataset_other_unknown.txt
Legend
DAG
Dataset

Datasets

Filter datasets with updates in the past:

URI	LAST UPDATE
s3://consuming_1_task/dataset_other.txt Total Updates: 0	
s3://consuming_2_task/dataset_other_unknown.txt Total Updates: 0	
s3://dag1/output_1.txt Total Updates: 0	
s3://dag2/output_1.txt Total Updates: 0	
s3://this-dataset-doesnt-get-triggered Total Updates: 0	
s3://unrelated/dataset3.txt Total Updates: 0	
s3://unrelated/dataset_other_unknown.txt Total Updates: 0	
s3://unrelated_task/dataset_other_unknown.txt Total Updates: 0	

Airflow UI: страница Datasets

Security

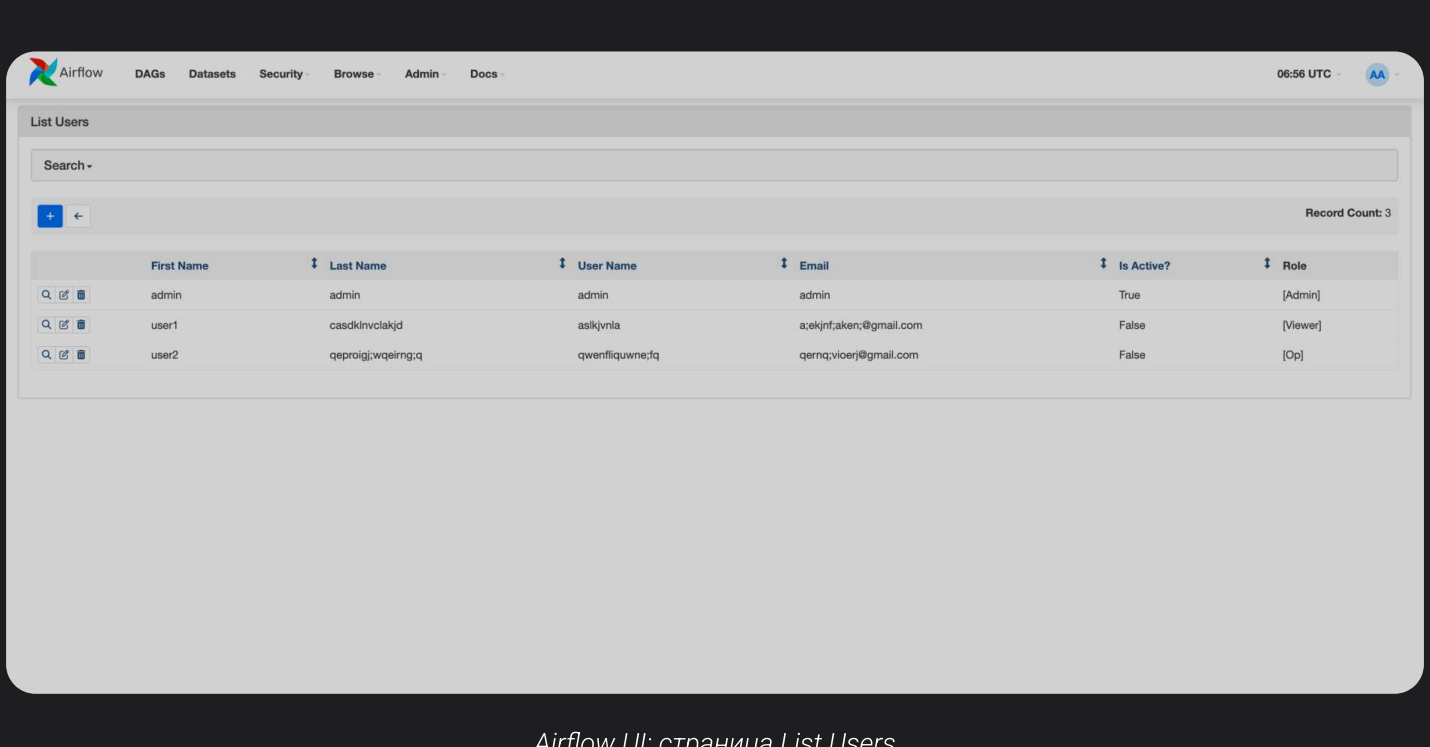
Меню ***Security*** обеспечивает доступ к информации о пользователе, ролях, настройках безопасности и предоставляет возможность управлять ими.

List Users

На странице ***List Users*** отображается список пользователей Airflow. Каждый пользователь имеет следующие характеристики: имя, фамилия, имя пользователя, адрес электронной почты, статус и роль.

Вы можете просмотреть, отредактировать или удалить конкретного пользователя, выбрав соответствующее действие напротив его имени.

Чтобы добавить пользователя, нажмите **+** и введите информацию о пользователе.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
06:56 UTC
Search

List Users

Record Count: 3

First Name	Last Name	User Name	Email	Is Active?	Role
admin	admin	admin	admin	True	[Admin]
user1	casdklnvclakjd	aslkjvnla	a;ekjnf;aken;@gmail.com	False	[Viewer]
user2	qeproigj;wqeirng;q	qwenfliquwne;fq	qerng;vioerj@gmail.com	False	[Op]

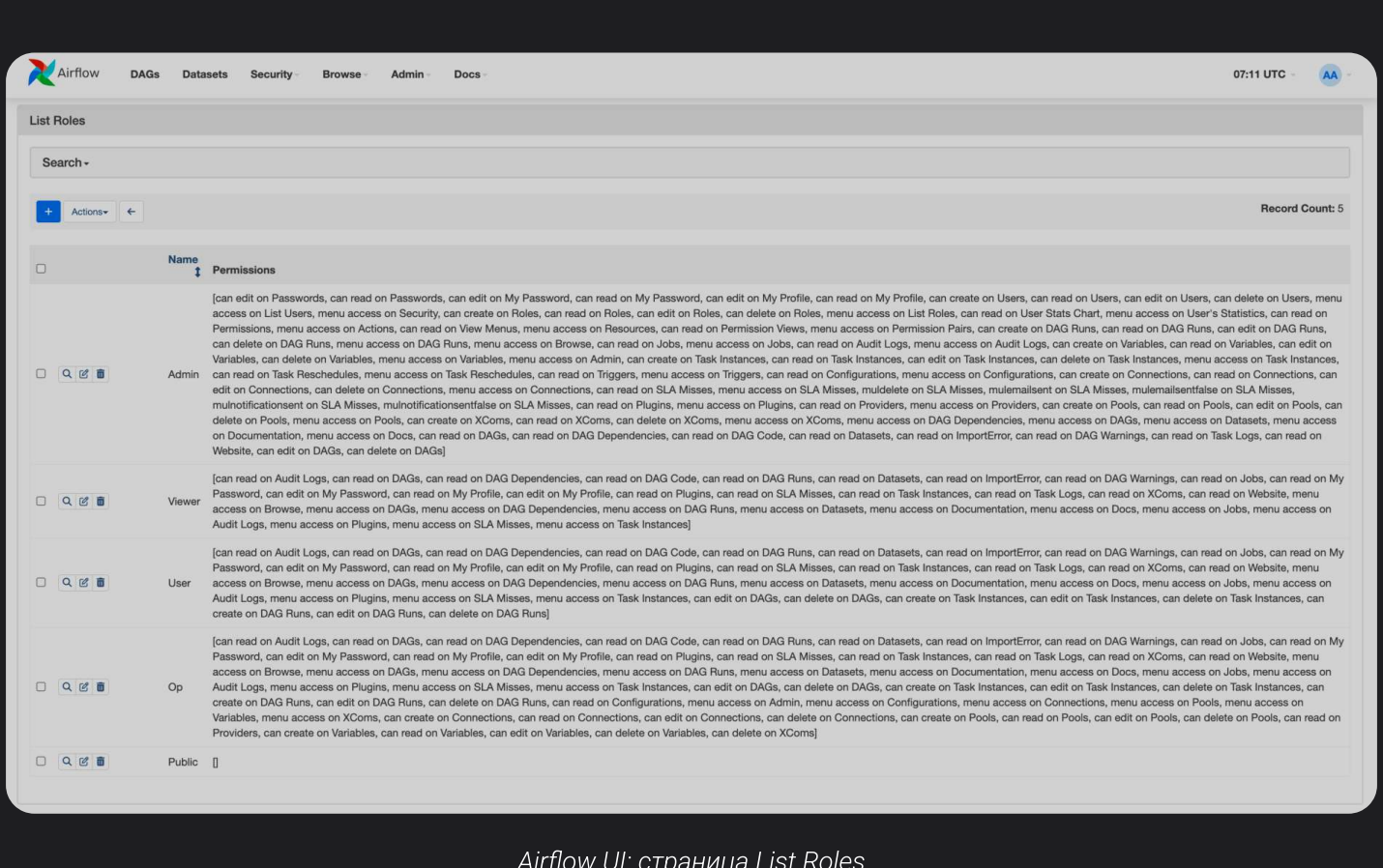
Airflow UI: страница List Users

List Roles

На странице ***List Roles*** отображается список ролей пользователя. Каждая роль имеет название и список разрешений, привязанных к этой роли.

Вы можете просмотреть, отредактировать или удалить роль, выбрав соответствующее действие напротив ее имени.

Чтобы добавить роль, нажмите **+** и введите имя роли и ее разрешения.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
07:11 UTC
List Roles
Search
Actions
Record Count: 5
Name
Permissions
Admin
Viewer
User
Op
Public

Airflow UI: страница List Roles

User's statistics

На странице ***User's statistics*** отображается диаграмма количества входов пользователей. Чтобы показать количество неудачных попыток входа в систему, выберите фильтр ***Failed Login Count***.

Actions, Resources, Permissions

- ***Actions*** — на странице отображается список действий, которые можно предоставить пользователю.
- ***Resources*** — на странице отображается список ресурсов, к которым может быть предоставлен доступ.
- ***Permissions*** — на странице отображается список разрешений, которые можно предоставить для определенного ресурса.

Browse

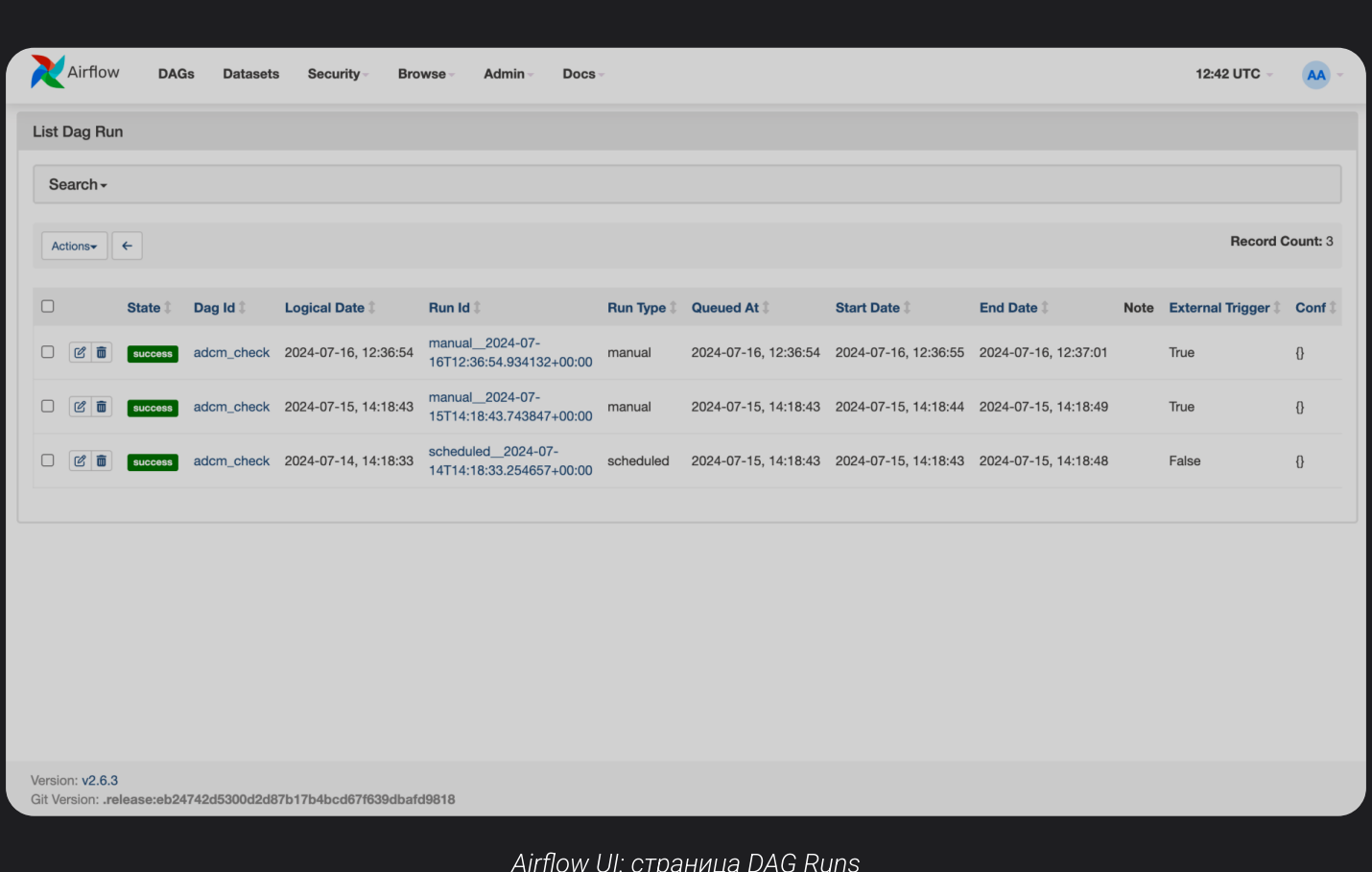
Меню ***Browse*** содержит набор страниц со списками, которые позволяют просматривать статистику по нескольким объектам одновременно и управлять ими.

DAG Runs

На странице ***DAG Runs*** отображается список всех запусков DAG.

Вы можете управлять несколькими DAG одновременно, отметив нужные запуски DAG и выбрав для них общее действие.

Ссылки в столбцах ***Dag Id*** и ***Run Id*** открывают страницу этого DAG.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:42 UTC
Search
Actions

List Dag Run

Record Count: 3

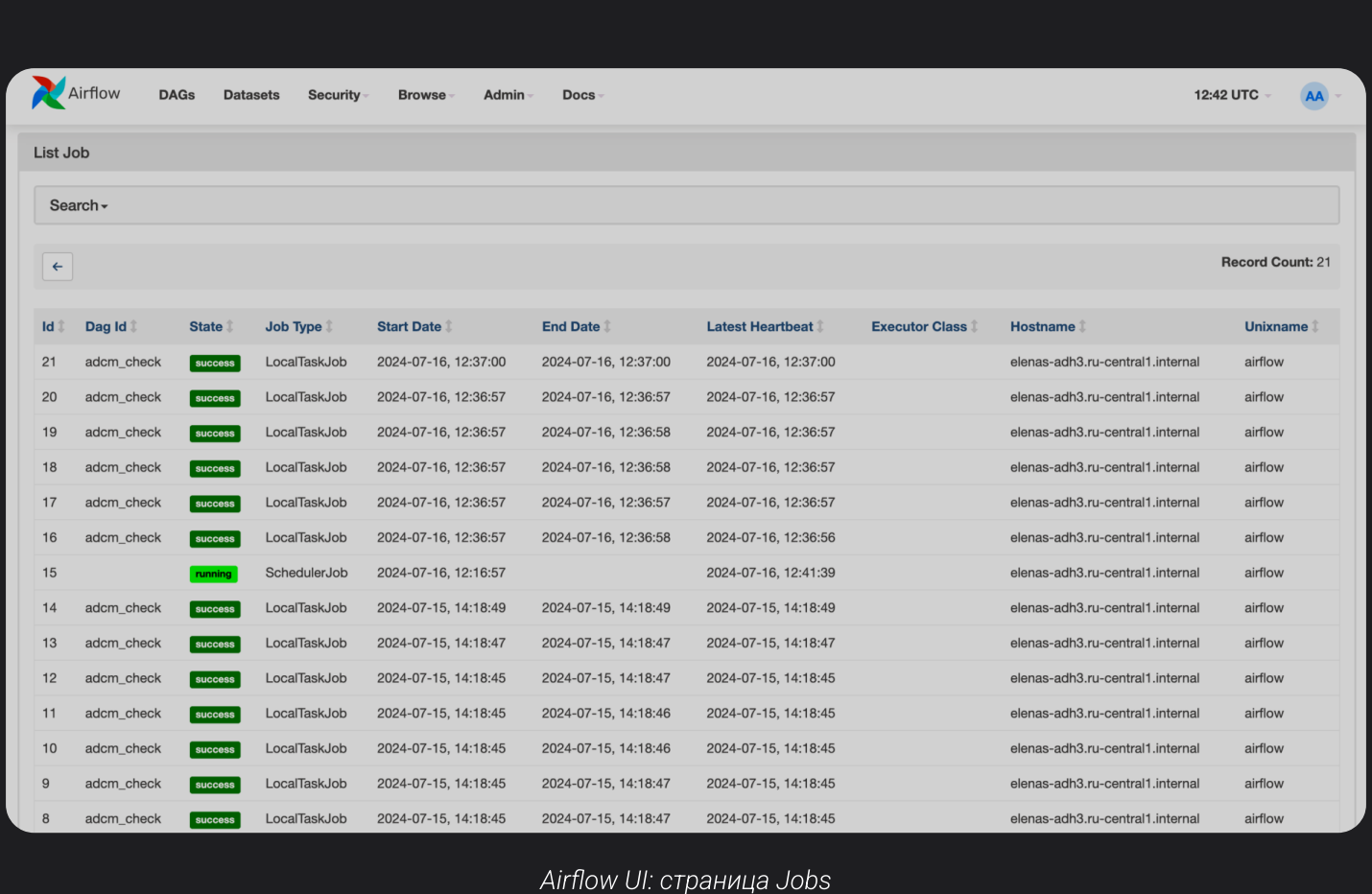
State	Dag Id	Logical Date	Run Id	Run Type	Queued At	Start Date	End Date	Note	External Trigger	Conf
success	adcm_check	2024-07-16, 12:36:54	manual__2024-07-16T12:36:54.934132+00:00	manual	2024-07-16, 12:36:54	2024-07-16, 12:36:55	2024-07-16, 12:37:01		True	{}
success	adcm_check	2024-07-15, 14:18:43	manual__2024-07-15T14:18:43.743847+00:00	manual	2024-07-15, 14:18:43	2024-07-15, 14:18:44	2024-07-15, 14:18:49		True	{}
success	adcm_check	2024-07-14, 14:18:33	scheduled__2024-07-14T14:18:33.254657+00:00	scheduled	2024-07-15, 14:18:43	2024-07-15, 14:18:43	2024-07-15, 14:18:48		False	{}

Version: v2.6.3
Git Version: .release:eb24742d5300d2d87b17b4bcd67f639dbafd9818

Airflow UI: страница DAG Runs

Jobs

На странице ***Jobs*** отображается список всех заданий с информацией об их DAG, текущем статусе, датах начала и окончания, последнем heartbeat, executor class, имени хоста и unixname.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:42 UTC
Search

List Job

Record Count: 21

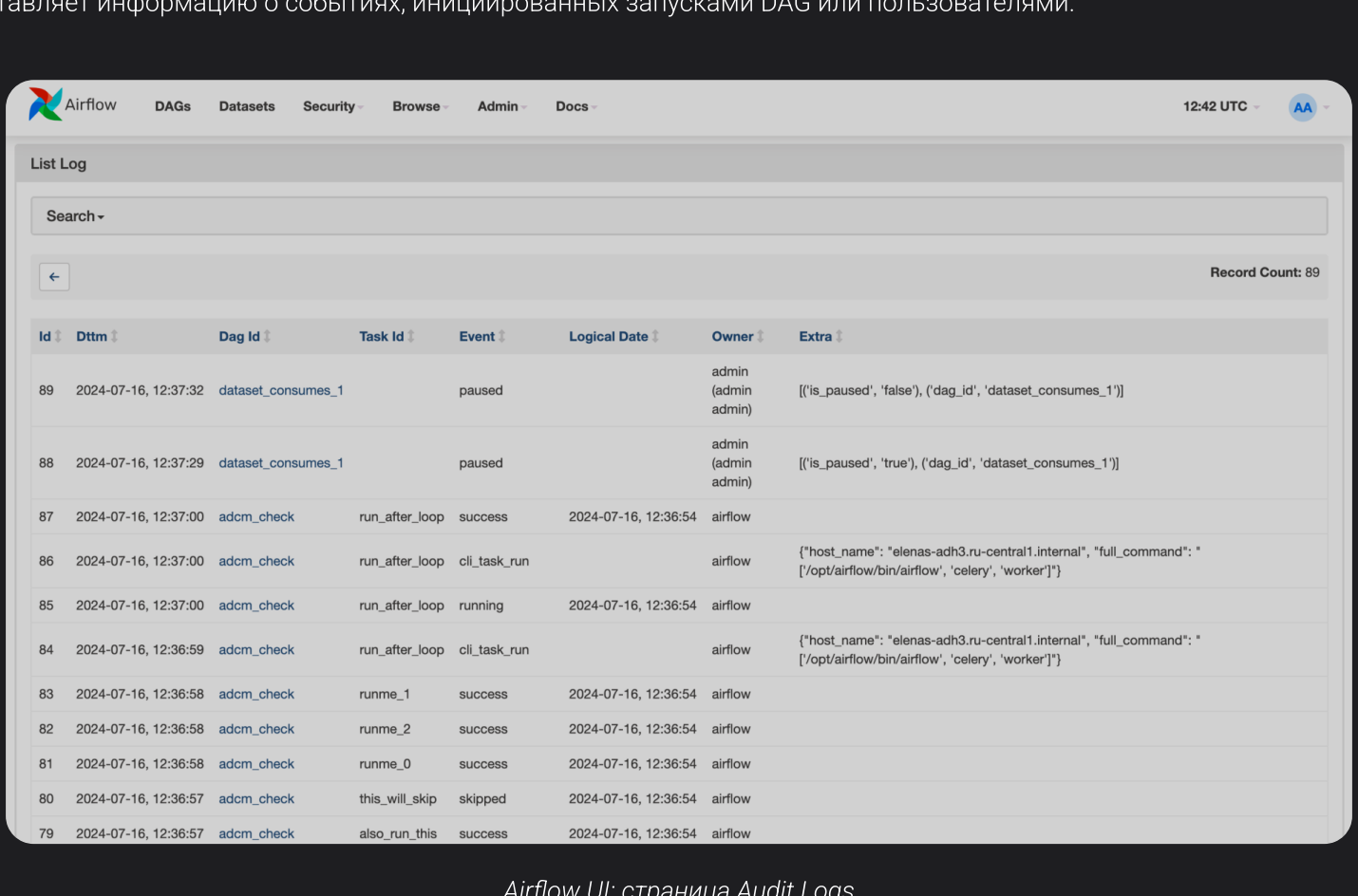
Id	Dag Id	State	Job Type	Start Date	End Date	Latest Heartbeat	Executor Class	Hostname	Unixname
21	adcm_check	success	LocalTaskJob	2024-07-16, 12:37:00	2024-07-16, 12:37:00	2024-07-16, 12:37:00		elenas-adh3.ru-central1.internal	airflow
20	adcm_check	success	LocalTaskJob	2024-07-16, 12:36:57	2024-07-16, 12:36:57	2024-07-16, 12:36:57		elenas-adh3.ru-central1.internal	airflow
19	adcm_check	success	LocalTaskJob	2024-07-16, 12:36:57	2024-07-16, 12:36:58	2024-07-16, 12:36:57		elenas-adh3.ru-central1.internal	airflow
18	adcm_check	success	LocalTaskJob	2024-07-16, 12:36:57	2024-07-16, 12:36:58	2024-07-16, 12:36:57		elenas-adh3.ru-central1.internal	airflow
17	adcm_check	success	LocalTaskJob	2024-07-16, 12:36:57	2024-07-16, 12:36:57	2024-07-16, 12:36:57		elenas-adh3.ru-central1.internal	airflow
16	adcm_check	success	LocalTaskJob	2024-07-16, 12:36:57	2024-07-16, 12:36:58	2024-07-16, 12:36:56		elenas-adh3.ru-central1.internal	airflow
15		running	SchedulerJob	2024-07-16, 12:16:57		2024-07-16, 12:41:39		elenas-adh3.ru-central1.internal	airflow
14	adcm_check	success	LocalTaskJob	2024-07-15, 14:18:49	2024-07-15, 14:18:49	2024-07-15, 14:18:49		elenas-adh3.ru-central1.internal	airflow
13	adcm_check	success	LocalTaskJob	2024-07-15, 14:18:47	2024-07-15, 14:18:47	2024-07-15, 14:18:47		elenas-adh3.ru-central1.internal	airflow
12	adcm_check	success	LocalTaskJob	2024-07-15, 14:18:45	2024-07-15, 14:18:47	2024-07-15, 14:18:45		elenas-adh3.ru-central1.internal	airflow
11	adcm_check	success	LocalTaskJob	2024-07-15, 14:18:45	2024-07-15, 14:18:46	2024-07-15, 14:18:45		elenas-adh3.ru-central1.internal	airflow
10	adcm_check	success	LocalTaskJob	2024-07-15, 14:18:45	2024-07-15, 14:18:46	2024-07-15, 14:18:45		elenas-adh3.ru-central1.internal	airflow
9	adcm_check	success	LocalTaskJob	2024-07-15, 14:18:45	2024-07-15, 14:18:47	2024-07-15, 14:18:45		elenas-adh3.ru-central1.internal	airflow
8	adcm_check	success	LocalTaskJob	2024-07-15, 14:18:45	2024-07-15, 14:18:47	2024-07-15, 14:18:45		elenas-adh3.ru-central1.internal	airflow

Airflow UI: страница Jobs

Audit Logs

В Airflow логи аудита используются для отслеживания действий пользователей и системных событий, происходящих во время выполнения DAG.

Страница ***Audit Logs*** предоставляет информацию о событиях, инициированных запусками DAG или пользователями.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:42 UTC
Search

List Log

Record Count: 89

Id	Dttm	Dag Id	Task Id	Event	Logical Date	Owner	Extra
89	2024-07-16, 12:37:32	dataset_consumes_1		paused		admin (admin admin)	[('is_paused', 'false'), ('dag_id', 'dataset_consumes_1')]
88	2024-07-16, 12:37:29	dataset_consumes_1		paused		admin (admin admin)	[('is_paused', 'true'), ('dag_id', 'dataset_consumes_1')]
87	2024-07-16, 12:37:00	adcm_check	run_after_loop	success	2024-07-16, 12:36:54	airflow	
86	2024-07-16, 12:37:00	adcm_check	run_after_loop	cli_task_run		airflow	{"host_name": "elenas-adh3.ru-central1.internal", "full_command": "['/opt/airflow/bin/airflow', 'celery', 'worker']"}
85	2024-07-16, 12:37:00	adcm_check	run_after_loop	running	2024-07-16, 12:36:54	airflow	
84	2024-07-16, 12:36:59	adcm_check	run_after_loop	cli_task_run		airflow	{"host_name": "elenas-adh3.ru-central1.internal", "full_command": "['/opt/airflow/bin/airflow', 'celery', 'worker']"}
83	2024-07-16, 12:36:58	adcm_check	runme_1	success	2024-07-16, 12:36:54	airflow	
82	2024-07-16, 12:36:58	adcm_check	runme_2	success	2024-07-16, 12:36:54	airflow	
81	2024-07-16, 12:36:58	adcm_check	runme_0	success	2024-07-16, 12:36:54	airflow	
80	2024-07-16, 12:36:57	adcm_check	this_will_skip	skipped	2024-07-16, 12:36:54	airflow	
79	2024-07-16, 12:36:57	adcm_check	also_run_this	success	2024-07-16, 12:36:54	airflow	

Airflow UI: страница Audit Logs

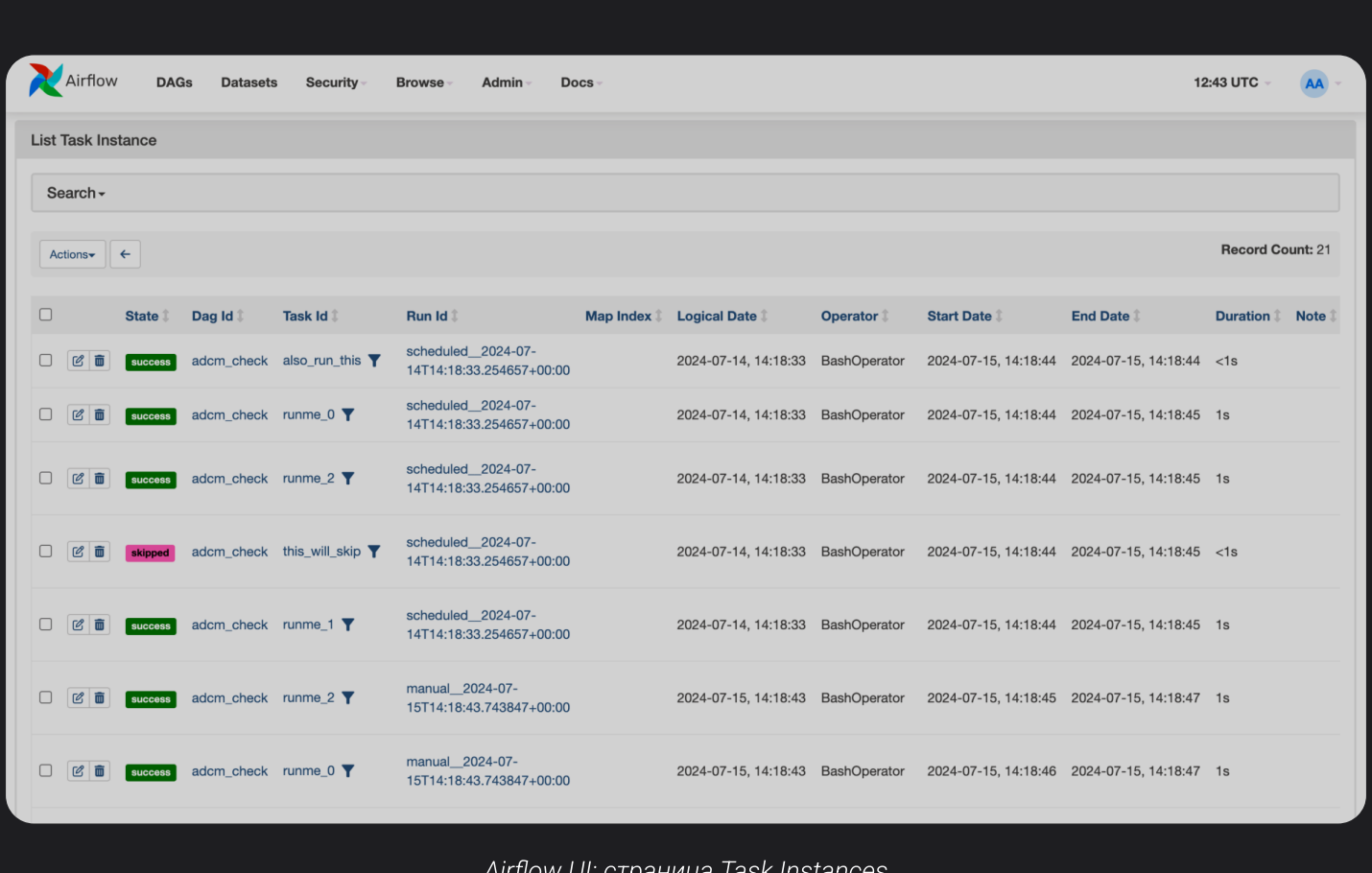
Task Instances и Task Reschedules

Страница ***Task Instances*** предоставляет информацию о каждой задаче: DAG, к которому они принадлежат, время начала выполнения и окончания и многое другое.

Вы можете управлять несколькими задачами одновременно, выделив нужные задачи и выбрав для них общее действие.

Нажав ***Dag Id***, ***Task Id*** или ***Run Id*** конкретной задачи, можно перейти на страницу DAG, к которому принадлежит задача.

На странице ***Task Reschedules*** отображается список задач, время выполнения которых было перенесено, и информация о них.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:43 UTC
Search
Actions

List Task Instance

Record Count: 21

State	Dag Id	Task Id	Run Id	Map Index	Logical Date	Operator	Start Date	End Date	Duration	Note
success	adcm_check	also_run_this	scheduled__2024-07-14T14:18:33.254657+00:00		2024-07-14, 14:18:33	BashOperator	2024-07-15, 14:18:44	2024-07-15, 14:18:44	<1s	
success	adcm_check	runme_0	scheduled__2024-07-14T14:18:33.254657+00:00		2024-07-14, 14:18:33	BashOperator	2024-07-15, 14:18:44	2024-07-15, 14:18:45	1s	
success	adcm_check	runme_2	scheduled__2024-07-14T14:18:33.254657+00:00		2024-07-14, 14:18:33	BashOperator	2024-07-15, 14:18:44	2024-07-15, 14:18:45	1s	
skipped	adcm_check	this_will_skip	scheduled__2024-07-14T14:18:33.254657+00:00		2024-07-14, 14:18:33	BashOperator	2024-07-15, 14:18:44	2024-07-15, 14:18:45	<1s	
success	adcm_check	runme_1	scheduled__2024-07-14T14:18:33.254657+00:00		2024-07-14, 14:18:33	BashOperator	2024-07-15, 14:18:44	2024-07-15, 14:18:45	1s	
success	adcm_check	runme_2	manual__2024-07-15T14:18:43.743847+00:00		2024-07-15, 14:18:43	BashOperator	2024-07-15, 14:18:45	2024-07-15, 14:18:47	1s	
success	adcm_check	runme_0	manual__2024-07-15T14:18:43.743847+00:00		2024-07-15, 14:18:43	BashOperator	2024-07-15, 14:18:46	2024-07-15, 14:18:47	1s	

Airflow UI: страница Task Instances

Triggers

На странице ***Triggers*** отображается список существующих триггеров.

Более подробно о триггерах читайте в документации Airflow.

SLA Misses

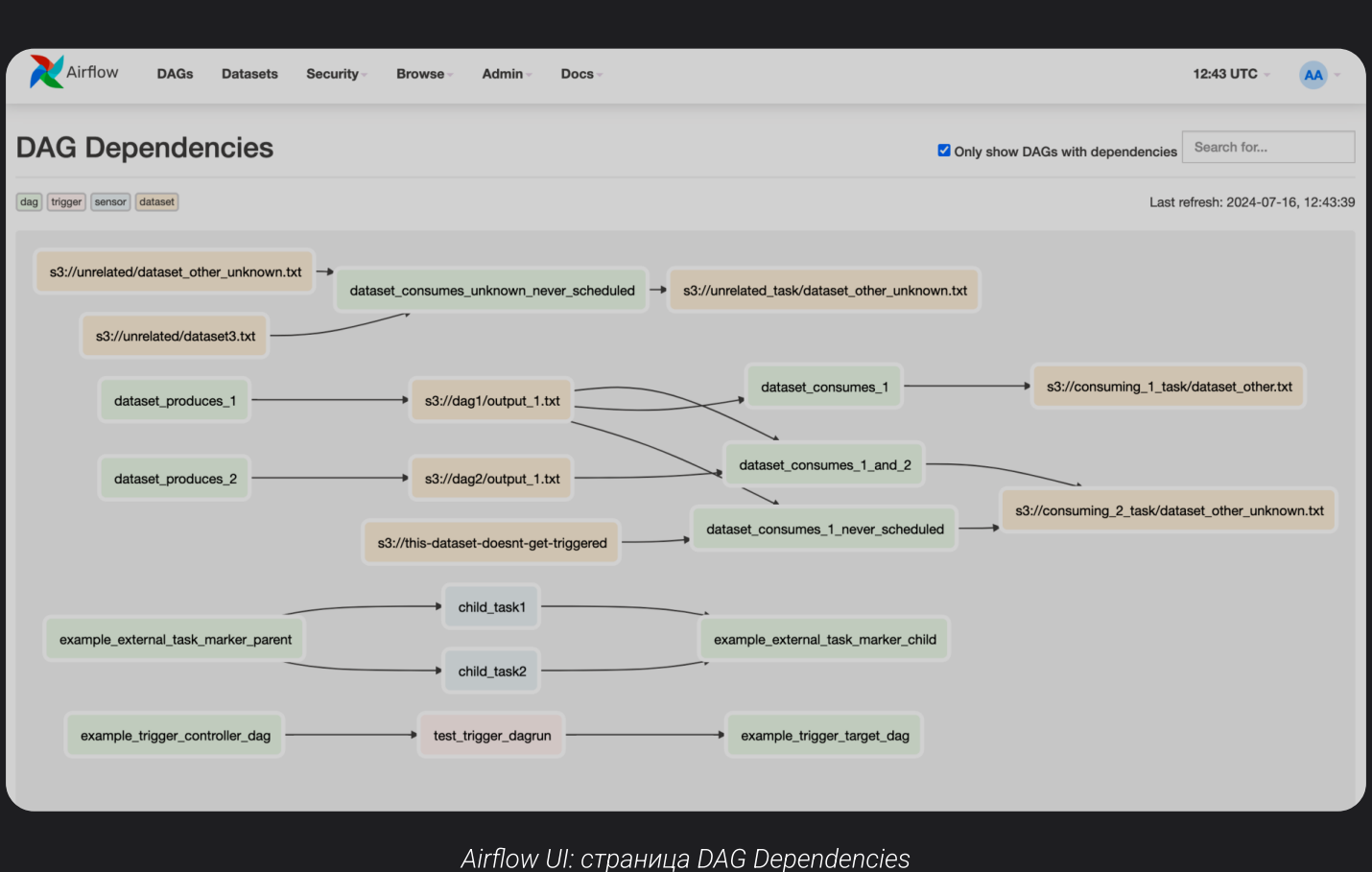
Соглашение об уровне обслуживания или SLA — это ожидаемое максимальное время, в течение которого задача должна быть выполнена относительно времени старта DAG.

Если выполнение задачи занимает больше времени, это отображается на странице ***SLA Misses***.

Более подробно об SLA читайте в документации Airflow.

DAG Dependencies

На странице ***DAG Dependencies*** показан график связей между разными DAG, триггерами, сенсорами (sensors) и наборами данных.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:43 UTC
DAG Dependencies
Only show DAGs with dependencies
Search for...
dag
trigger
sensor
dataset
Last refresh: 2024-07-16, 12:43:39
s3://unrelated/dataset_other_unknown.txt
dataset_consumes_unknown_never_scheduled
s3://unrelated_task/dataset_other_unknown.txt
s3://unrelated/dataset3.txt
dataset_produces_1
s3://dag1/output_1.txt
dataset_consumes_1
s3://consuming_1_task/dataset_other.txt
dataset_produces_2
s3://dag2/output_1.txt
dataset_consumes_1_and_2
s3://this-dataset-doesnt-get-triggered
dataset_consumes_1_never_scheduled
s3://consuming_2_task/dataset_other_unknown.txt
child_task1
example_external_task_marker_parent
example_external_task_marker_child
child_task2
example_trigger_controller_dag
test_trigger_dagrun
example_trigger_target_dag

Airflow UI: страница DAG Dependencies

Admin

Меню ***Admin*** предоставляет действия по управлению ресурсами и информацию о конфигурации Airflow.

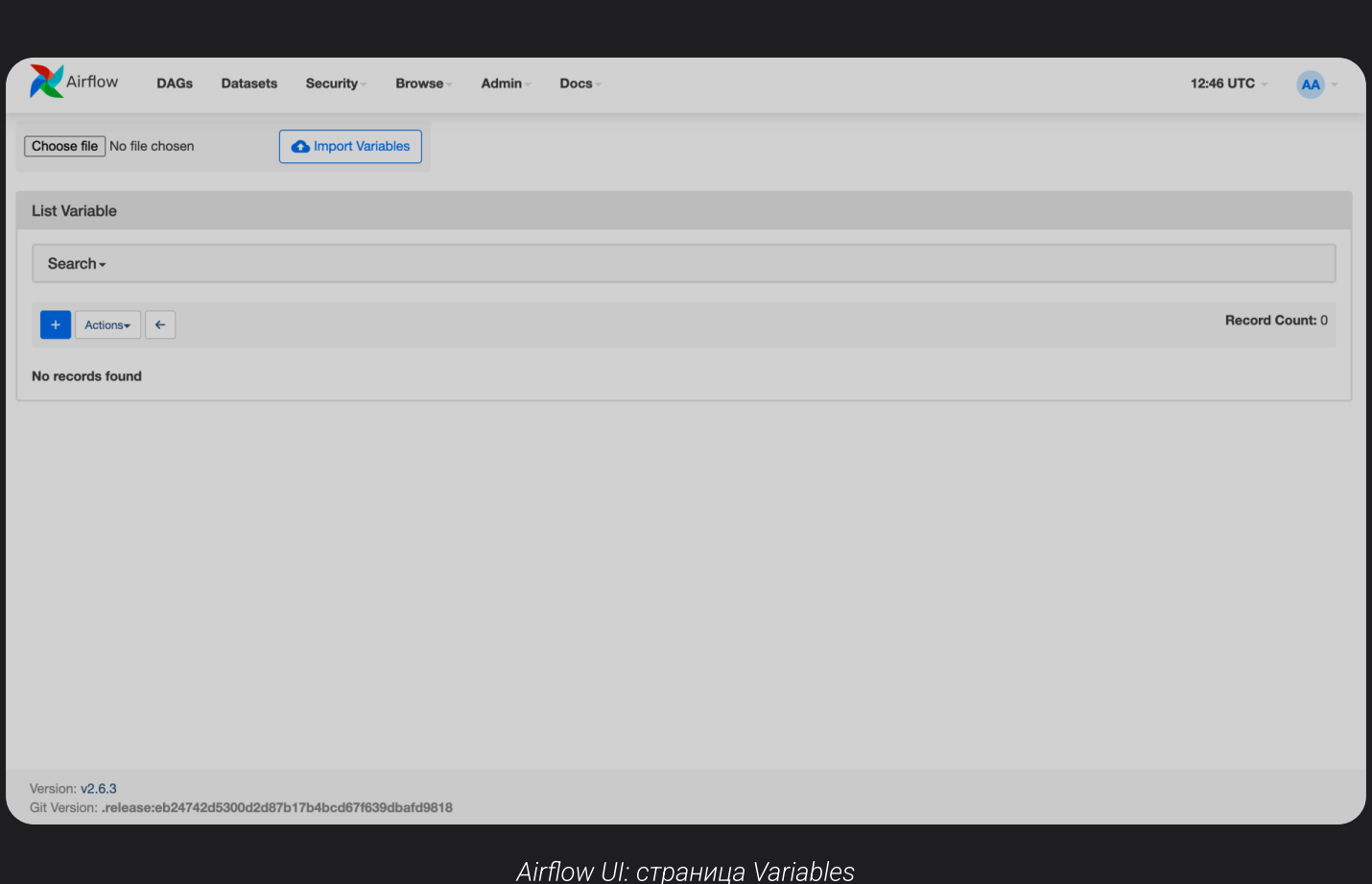
Variables

Переменная в Airflow — это пара ключ/значение, которая содержит строку. Страница ***Variables*** позволяет создавать переменные, которые можно использовать в DAG и в коде задач.

По умолчанию значение переменной будет скрыто, если ключ содержит следующие слова: `password`, `secret`, `passwd`, `authorization`, `api_key`, `apikey`, `access_token`. Вы можете изменить эту настройку по умолчанию, чтобы переменная отображала пароли и токены.

Чтобы добавить одну переменную, нажмите **+** и введите ключ/значение переменной.

Чтобы загрузить несколько переменных одновременно, нажмите ***Choose file***, загрузите файл с переменными и нажмите ***Import Variables***.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:46 UTC
Choose file
No file chosen
Import Variables
Search
Actions

List Variable

Record Count: 0

No records found

Version: v2.6.3

Airflow UI: страница Variables

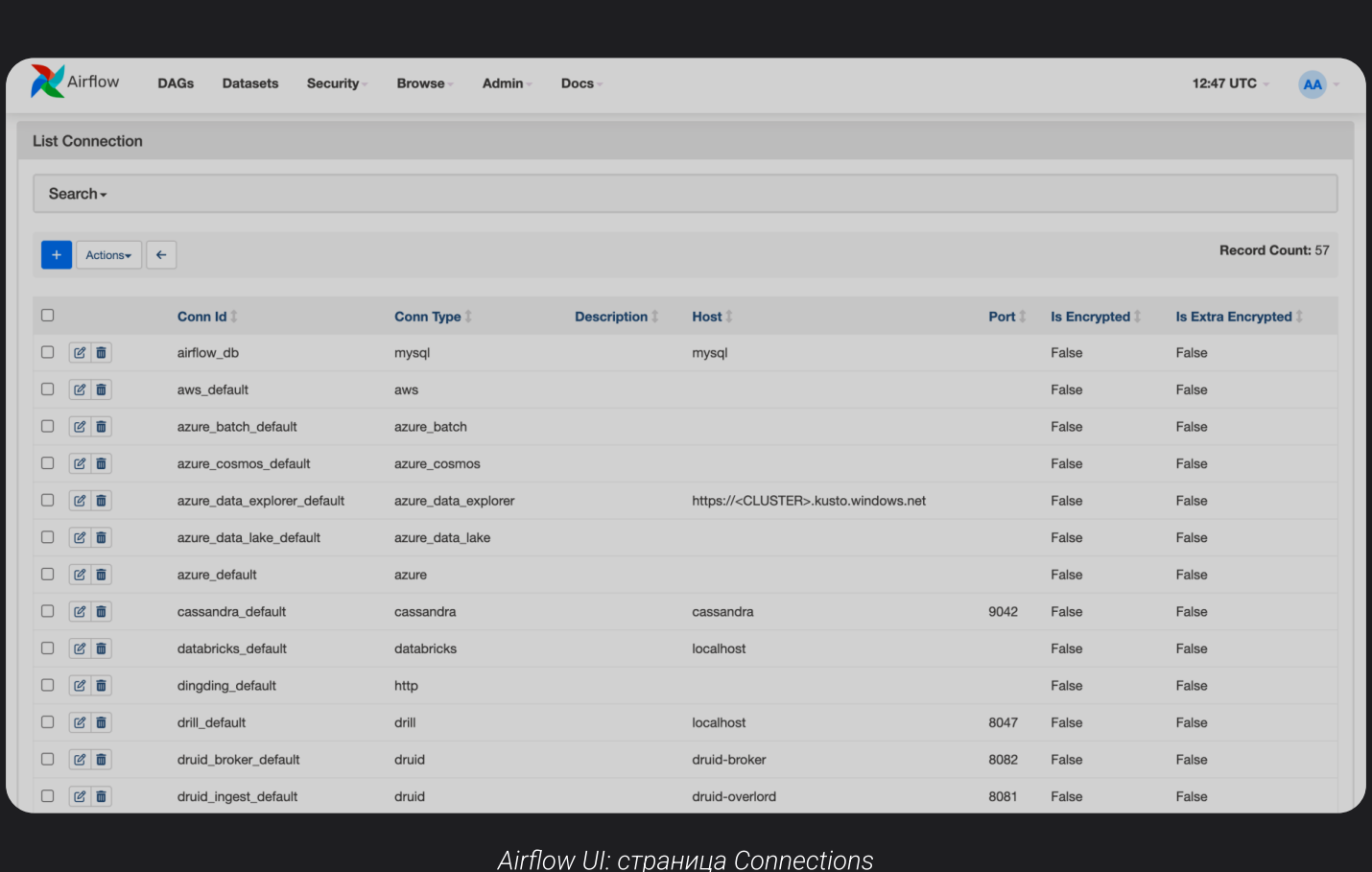
Configurations

На странице может отображаться таблица с конфигурацией инстанса Airflow. По умолчанию страница пустая.

Connections

Соединения (connections) позволяют интегрировать Airflow с другими сервисами. На этой странице отображается список созданных соединений.

Для создания соединения нажмите **+** и введите параметры подключения. После сохранения информации о соединении его можно проверить, нажав ***Test***.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:47 UTC
Search
Actions

List Connection

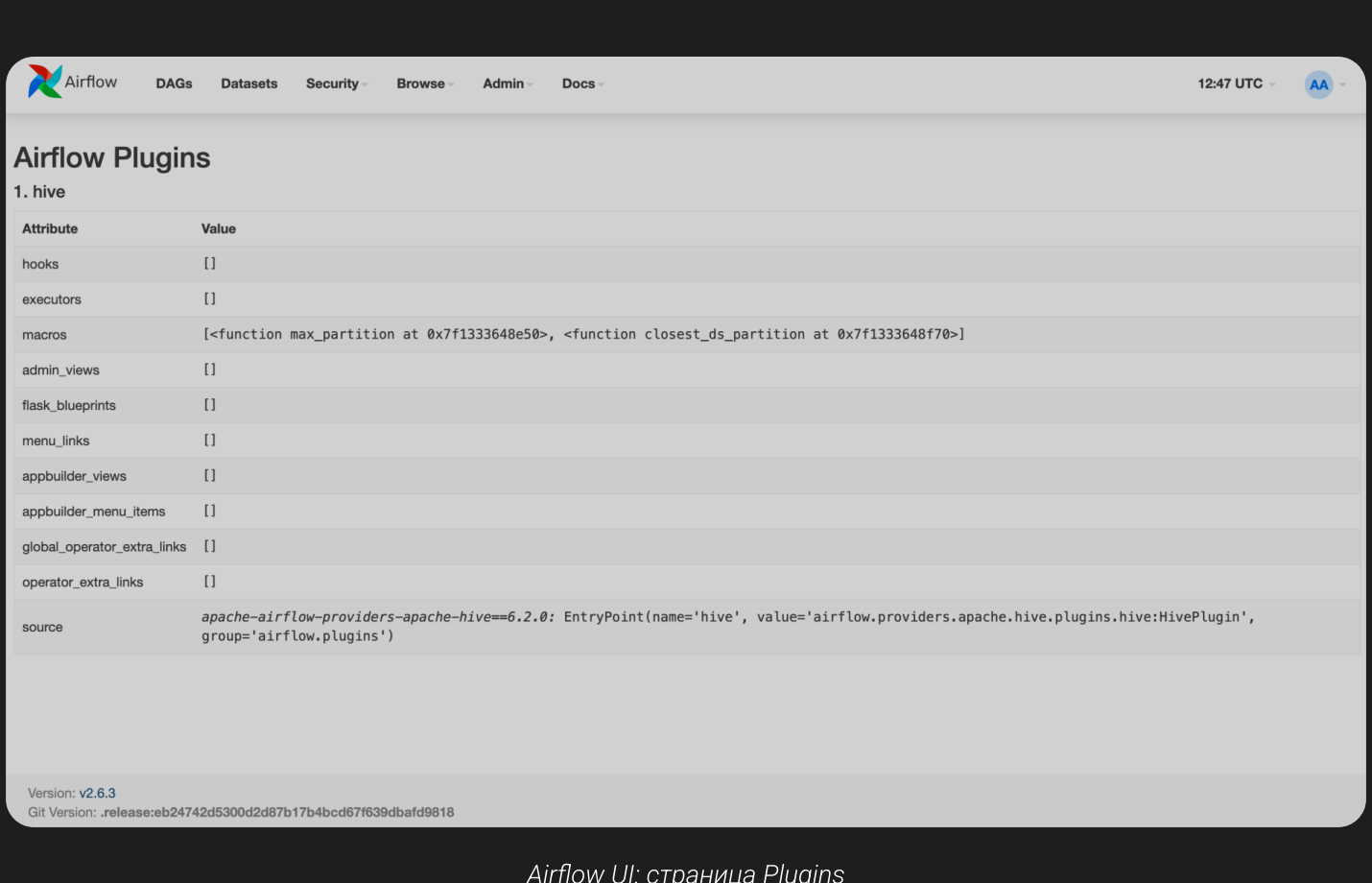
Record Count: 57

Conn Id	Conn Type	Description	Host	Port	Is Encrypted	Is Extra Encrypted
airflow_db	mysql		mysql		False	False
aws_default	aws				False	False
azure_batch_default	azure_batch				False	False
azure_cosmos_default	azure_cosmos				False	False
azure_data_explorer_default	azure_data_explorer		https://<CLUSTER>.kusto.windows.net		False	False
azure_data_lake_default	azure_data_lake				False	False
azure_default	azure				False	False
cassandra_default	cassandra		cassandra	9042	False	False
databricks_default	databricks		localhost		False	False
dingding_default	http				False	False
drill_default	drill		localhost	8047	False	False
druid_broker_default	druid		druid-broker	8082	False	False
druid_ingest_default	druid		druid-overlord	8081	False	False

Airflow UI: страница Connections

Plugins

На этой странице представлен список доступных плагинов. Чтобы добавить плагин, сначала необходимо добавить соединение.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:47 UTC

Airflow Plugins

1. hive

Attribute	Value
hooks	[]
executors	[]
macros	[<function max_partition at 0x7f1333640e50>, <function closest_ds_partition at 0x7f1333640f70>]
admin_views	[]
flask_blueprints	[]
menu_links	[]
appbuilder_views	[]
appbuilder_menu_items	[]
global_operator_extra_links	[]
operator_extra_links	[]
source	apache-airflow-providers-apache-hive==6.2.0: EntryPoint(name='hive', value='airflow.providers.apache.hive.plugins.hive:HivePlugin', group='airflow.plugins')

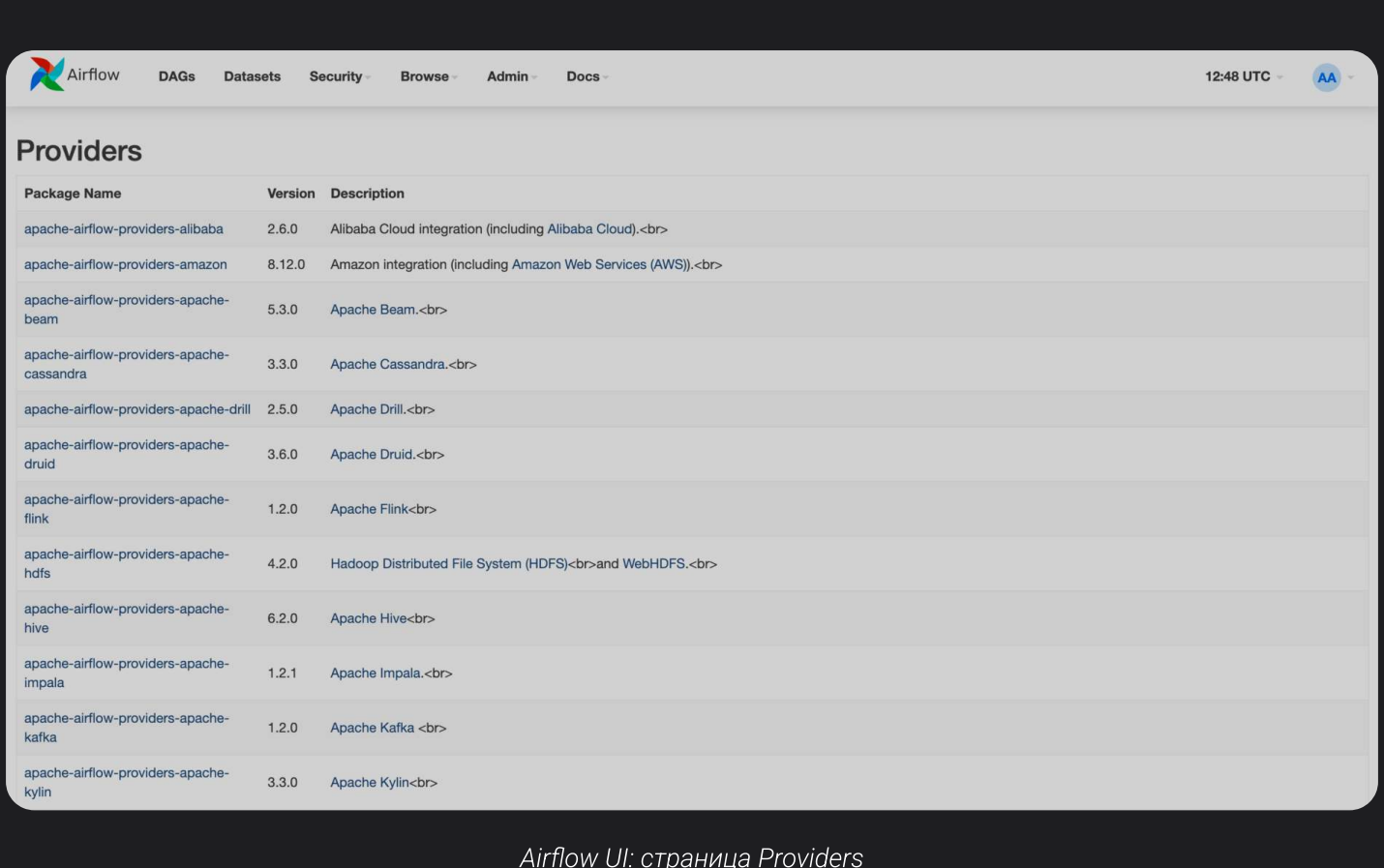
Version: v2.6.3

Airflow UI: страница Plugins

Providers

На этой странице отображаются доступные провайдеры (providers) для интеграции с другими сервисами.

Нажав на имя провайдера, можно перейти на страницу документации Airflow этого провайдера. Ссылки в столбце ***Description*** ведут на официальные страницы сервисов.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:48 UTC

Providers

Package Name	Version	Description
apache-airflow-providers-alibaba	2.6.0	Alibaba Cloud integration (including Alibaba Cloud).
apache-airflow-providers-amazon	8.12.0	Amazon integration (including Amazon Web Services (AWS)).
apache-airflow-providers-apache-beam	5.5.0	Apache Beam.
apache-airflow-providers-apache-cassandra	3.3.0	Apache Cassandra.
apache-airflow-providers-apache-drill	2.5.0	Apache Drill.
apache-airflow-providers-apache-druid	3.6.0	Apache Druid.
apache-airflow-providers-apache-flink	1.2.0	Apache Flink.
apache-airflow-providers-apache-hdfs	4.2.0	Hadoop Distributed File System (HDFS) and WebHDFS.
apache-airflow-providers-apache-hive	6.2.0	Apache Hive
apache-airflow-providers-apache-impala	1.2.1	Apache Impala.
apache-airflow-providers-apache-kafka	1.2.0	Apache Kafka
apache-airflow-providers-apache-kylin	3.3.0	Apache Kylin

Airflow UI: страница Providers

Pools

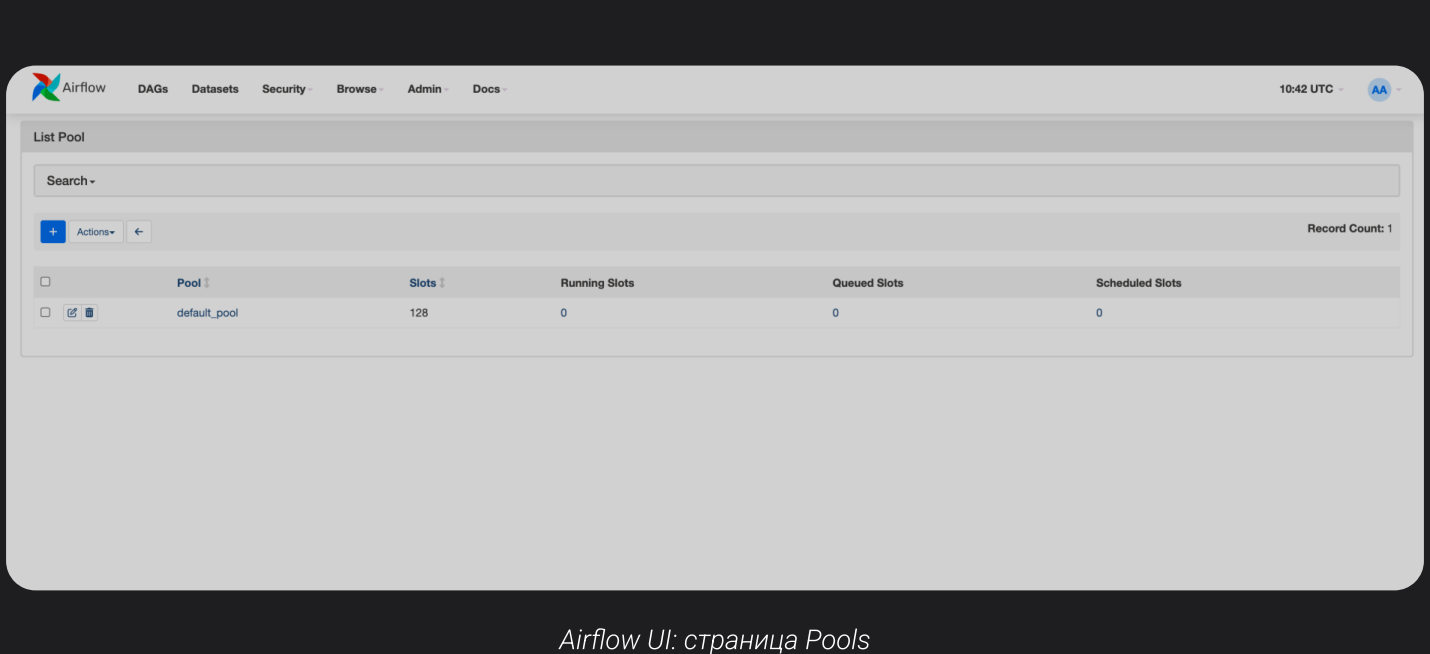
На этой странице отображается список существующих пулов (pools). В Airflow пулы позволяют ограничить параллелизм набора задач. По умолчанию все задачи выполняются через пул ***default_pool***.

Каждый пул имеет следующие параметры:

- ***Pool*** — имя пула. Нажав на имя пула, можно перейти к списку задач, принадлежащих этому пулу.
- ***Slots*** — количество задач, которые может содержать пул.
- ***Running Slots*** — количество занятых слотов в пуле. Нажав на количество слотов, можно перейти к списку задач, занимающих эти слоты.
- ***Queued Slots*** — количество зарезервированных слотов для задач в очереди. Нажав на количество слотов, можно перейти к списку задач, занимающих эти слоты.
- ***Scheduled Slots*** — количество слотов, необходимое для запланированных задач. Нажав на количество слотов, можно перейти к списку задач, занимающих эти слоты.

Чтобы создать пул, нажмите **+** и введите параметры пула.

Вы можете редактировать или удалить пул, выбрав соответствующее действие напротив имени пула.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
10:42 UTC
Search
Actions

List Pool

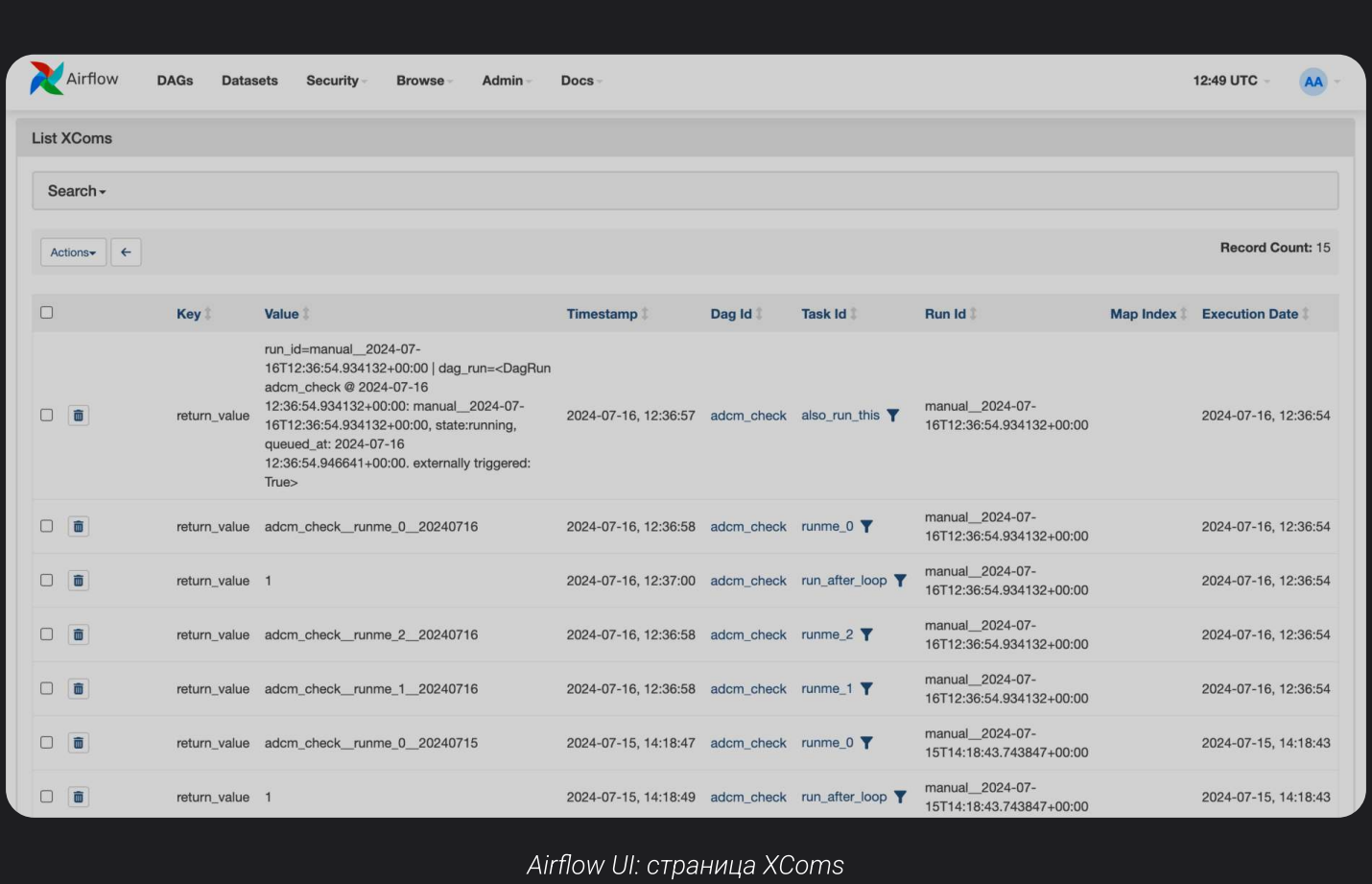
Record Count: 1

Pool	Slots	Running Slots	Queued Slots	Scheduled Slots
default_pool	128	0	0	0

Airflow UI: страница Pools

XComs

В Airflow XCom — это инструмент, который позволяет обмениваться данными между задачами. На этой странице отображается список доступных XCom.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
12:49 UTC
Search
Actions

List XComs

Record Count: 15

Key	Value	Timestamp	Dag Id	Task Id	Run Id	Map Index	Execution Date
return_value	run_id=manual__2024-07-16T12:36:54.934132+00:00 | dag_run=<DagRun adcm_check @ 2024-07-16 12:36:54.934132+00:00: manual__2024-07-16T12:36:54.934132+00:00, state:running, queued_at: 2024-07-16 12:36:54.946641+00:00. externally triggered: True>	2024-07-16, 12:36:57	adcm_check	also_run_this	manual__2024-07-16T12:36:54.934132+00:00		2024-07-16, 12:36:54
return_value	adcm_check__runme_0__20240716	2024-07-16, 12:36:58	adcm_check	runme_0	manual__2024-07-16T12:36:54.934132+00:00		2024-07-16, 12:36:54
return_value	1	2024-07-16, 12:37:00	adcm_check	run_after_loop	manual__2024-07-16T12:36:54.934132+00:00		2024-07-16, 12:36:54
return_value	adcm_check__runme_2__20240716	2024-07-16, 12:36:58	adcm_check	runme_2	manual__2024-07-16T12:36:54.934132+00:00		2024-07-16, 12:36:54
return_value	adcm_check__runme_1__20240716	2024-07-16, 12:36:58	adcm_check	runme_1	manual__2024-07-16T12:36:54.934132+00:00		2024-07-16, 12:36:54
return_value	adcm_check__runme_0__20240715	2024-07-15, 14:18:47	adcm_check	runme_0	manual__2024-07-15T14:18:43.743847+00:00		2024-07-15, 14:18:43
return_value	1	2024-07-15, 14:18:49	adcm_check	run_after_loop	manual__2024-07-15T14:18:43.743847+00:00		2024-07-15, 14:18:43

Airflow UI: страница XComs

Docs

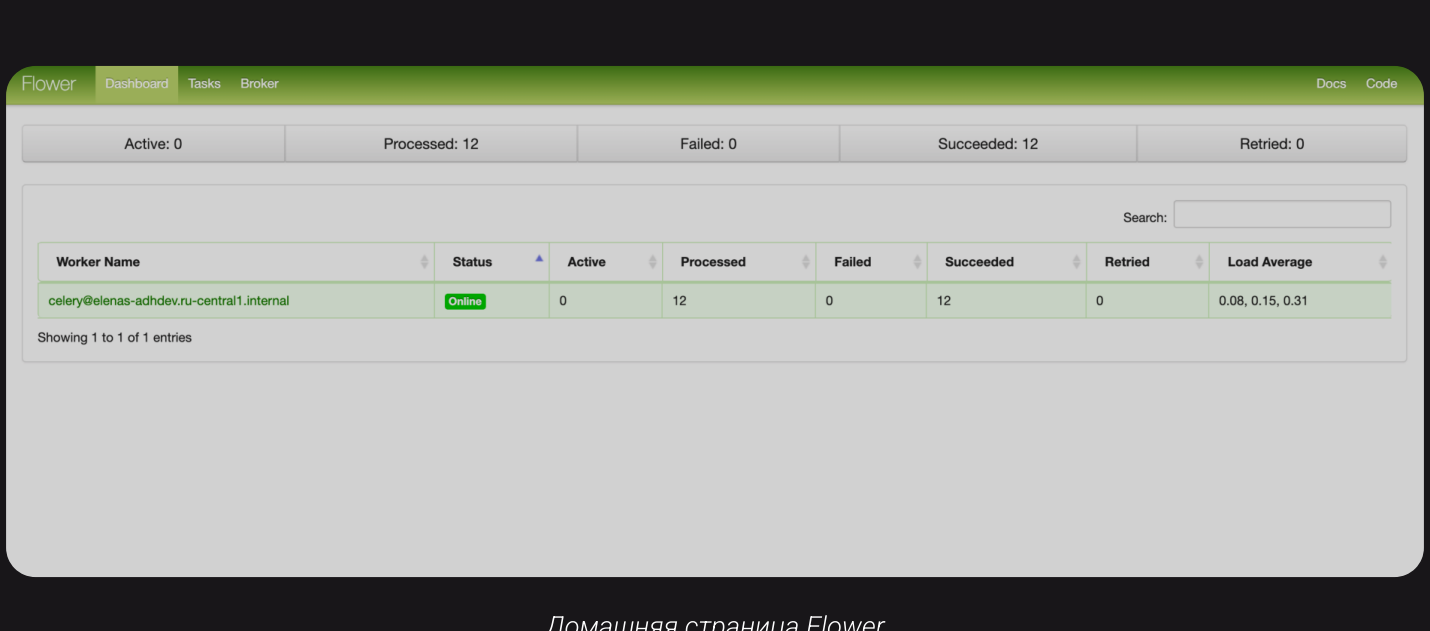
Меню ***Docs*** содержит ссылки на следующие ресурсы:

- ***Documentation*** — официальная документация Airflow.
- ***Airflow Website*** — официальный сайт Airflow.
- ***GitHub Repo*** — репозиторий проекта Airflow на GitHub.
- ***REST API Reference (Swagger UI)*** — описание методов REST API в формате Swagger.
- ***REST API Reference (Redoc)*** — описание методов REST API в формате Redoc.

Flower UI

Flower — это инструмент с веб-интерфейсом для мониторинга и администрирования кластеров Celery. В веб-интерфейсе Flower есть четыре вкладки:

- ***Workers*** — страница со списком существующих Celery workers и информация о них.
- ***Tasks*** — страница со списком задач с указанием информации о них: их worker, UUID, состояние, `args`/`kwargs`, время старта и продолжительность выполнения.
- ***Broker*** — на странице отображается информация о брокерах Celery.
- ***Documentation*** — вкладка перенаправляет на официальную документацию Flower.


Flower
Dashboard
Tasks
Broker
Docs
Code
Search:

Active: 0 | Processed: 12 | Failed: 0 | Succeeded: 12 | Retried: 0

Worker Name	Status	Active	Processed	Failed	Succeeded	Retried	Load Average
celery@elenas-adhdev.ru-central1.internal	Online	0	12	0	12	0	0.08, 0.15, 0.31

Showing 1 to 1 of 1 entries

Домашняя страница Flower

Дополнительную информацию о компонентах Flower можно получить в документации Flower.

Создание простого DAG

Альберт Багдасарян

Содержание

Назад к содержанию

Шаг 1. Подготовка DAG-файла

Шаг 2. Запуск DAG через Web-интерфейс

Шаг 3. Проверка результатов

Данный пример поясняет процесс создания и запуска вашего первого DAG с использованием файлов CSV. Этот DAG будет имитировать процесс ETL (Extract, Transform, Load) и включать следующие задачи:

- **Extract**. Эта задача загружает записи из исходного файла CSV и передает их в следующую задачу. Мы будем использовать тестовый файл people_ages_titles.csv, содержащий список из 1000 человек с их именами и возрастом.
- **Transform**. Эта задача выбирает записи, полученные от предыдущей задачи, по следующему критерию: возраст людей должен быть не старше 18 лет.
- **Load**. Эта задача загружает обработанные записи, полученные от предыдущей задачи, в новый файл CSV.

ПРИМЕЧАНИЕ

- На практике вы можете модифицировать этот пример путем замены входных файлов на соединения с входными базами данных.
- Другие примеры можно найти в документации Airflow.
- Описанный в данной статье пример актуален для версии Airflow 1.10.11.

Шаг 1. Подготовка DAG-файла

Подготовьте DAG с помощью следующих шагов:

1. Скопируйте файл ETL_test.py в папку вашего DAG, которая обычно расположена в домашнем каталоге Airflow и называется *dags*. Выясним содержимое файла:

```
import pandas as pd ①
from datetime import timedelta, datetime
from airflow import DAG
from airflow.operators.python_operator import PythonOperator

dag = DAG('ETL_test', ②
    schedule_interval=timedelta(days=1),
    start_date=datetime(2021, 12, 17, 0))

def extract_people(path: str, **context) -> None: ③
    extracted_people = pd.read_csv(path,sep=',',header=1)
    context['ti'].xcom_push(key='extracted_people', value=extracted_people)

def transform_people(**context) -> None: ④
    extracted_people = context['ti'].xcom_pull(key='extracted_people', task_ids=['extract_people'])[0]
    transformed_people = []
    for person in extracted_people.values:
        if int(person[1]) <= 18:
            transformed_people.append({
                'Name': person[0],
                'Age': person[1]
            })
    context['ti'].xcom_push(key='transformed_people', value=transformed_people)

def load_people(path: str, **context) -> None: ⑤
    transformed_people = context['ti'].xcom_pull(key='transformed_people', task_ids=['transform_people'])
    loaded_people = pd.DataFrame(transformed_people[0])
    loaded_people.to_csv(path, index=None)

task_extract_people = PythonOperator( ⑥
    task_id='extract_people',
    python_callable=extract_people,
    op_kwargs={'path': '/opt/airflow/input/people_ages_titles.csv'},
    dag=dag,
    provide_context=True
)

task_transform_people = PythonOperator( ⑥
    task_id='transform_people',
    python_callable=transform_people,
    dag=dag,
    provide_context=True
)

task_load_people = PythonOperator( ⑥
    task_id='load_people',
    python_callable=load_people,
    op_kwargs={'path': '/opt/airflow/output/loaded_people.csv'},
    dag=dag,
    provide_context=True
)

task_extract_people >> task_transform_people >> task_load_people ⑦
```

① **Импортирование**. Импортируем пакет pandas для обработки файлов CSV.

② **Определение DAG**. Создаем объект `ETL_test` типа DAG и планируем его запуск, начиная с 2021-12-17 00:00:00.

③ **Первая функция Python**. Ее использует первая задача `extract_people`. Эта функция считывает файл CSV по пути, заданному переменной `path`, с помощью модуля `pandas`. Она использует объект `context` для передачи загруженных данных через сервис транспортирования данных XCOM в следующую задачу `transform_people`.

④ **Вторая функция Python**. Ее использует вторая задача `transform_people`. Эта функция получает данные, собранные первой задачей `extract_people`, и выделяет из них записи с определенными значениями колонки `Age` (меньшими или равными 18). Она копирует найденные данные в массив и делает этот массив доступным для следующей задачи `load_people`.

⑤ **Третья функция Python**. Ее использует третья задача `load_people`. Эта функция считывает данные, сформированные второй задачей `transform_people`, и сохраняет их в новом файле по пути, заданному параметром `path`. Для этой цели она использует модуль `pandas`.

⑥ **Декларирование задач**. Этот фрагмент определяет операторы, функции и входные параметры, необходимые для задач. Он также привязывает задачи к графу DAG, заданному ранее.

⑦ **Определение зависимостей**. Этот фрагмент задает зависимости между созданными задачами.

ВАЖНО

Если вы используете AirFlow, установленный в контейнере Docker (как это поставляется в составе ADH), учитывайте различия в используемых путях. Например, используйте */opt/airflow/* вместо */srv/airflow/home/*.

2. Убедитесь, что файл расположен в каталоге DAG с помощью следующей команды:

```
$ ls -la /srv/airflow/home/dags/
```

Вывод на консоль приведен ниже:

```
total 8
drwxr-xr-x. 3 50000 50000   65 Dec 18 13:11 .
drwxr-xr-x. 6 50000 50000  126 Dec 18 11:07 ..
-rw-r--r--. 1 50000 50000  799 Dec 16 07:18 adcm_check.py
-rw-r--r--  1 root  root  1771 Dec 18 13:10 ETL_test.py
drwxr-xr-x. 2 50000 50000   74 Dec 18 13:10 __pycache__
```

3. Создайте папки *input* и *output* в домашнем каталоге Airflow:

```
$ sudo mkdir /srv/airflow/home/input
$ sudo mkdir /srv/airflow/home/output
```

Убедитесь, что эти папки созданы в домашнем каталоге Airflow:

```
$ ls -la /srv/airflow/home/
```

Вывод на консоль приведен ниже:

```
total 48
drwxr-xr-x. 6 50000 50000   126 Dec 18 13:58 .
drwxr-xr-x. 4 root  root     35 Dec 16 07:17 ..
-rw-r--r--. 1 root  root  37224 Dec 16 07:18 airflow.cfg
-rw-r--r--  1 50000 50000     3 Dec 18 11:07 airflow-webserver.pid
drwxr-xr-x. 3 50000 50000    65 Dec 18 13:11 dags
drwxr-xr-x  2 root  root      6 Dec 18 13:57 input
drwxr-xr-x. 7 50000 50000   121 Dec 17 15:55 logs
drwxr-xr-x  2 root  root      6 Dec 18 13:58 output
-rw-r--r--. 1 50000 50000  2528 Dec 16 07:19 unittests.cfg
```

4. Загрузите исходный файл people_ages_titles.csv в локальную файловую систему сервера Airflow. Убедитесь, что файл загружен успешно, используя следующую команду:

```
$ ls -la ~
```

Вывод на консоль приведен ниже:

```
total 64
drwx------. 7 dasha dasha   196 Dec 17 15:44 .
drwxr-xr-x. 3 root  root     19 Aug 31 11:55 ..
drwx------. 3 dasha dasha    17 Aug 31 15:29 .ansible
-rw-------. 1 dasha dasha 23583 Dec 17 18:17 .bash_history
-rw-r--r--. 1 dasha dasha    18 Apr  1  2020 .bash_logout
-rw-r--r--. 1 dasha dasha   193 Apr  1  2020 .bash_profile
-rw-r--r--. 1 dasha dasha   231 Apr  1  2020 .bashrc
drwx------. 3 dasha dasha    17 Dec  7 11:46 .cache
-rw-rw-r--. 1 dasha dasha  7217 Nov 25 07:05 dasha
-rw-rw-r--. 1 dasha dasha 17661 Dec 17 15:44 people_ages_titles.csv
drwxrw----. 3 dasha dasha    19 Dec  6 14:06 .pki
drwxrwxr-x. 2 dasha dasha    21 Nov 30 09:25 .sqlline
drwx------. 2 dasha dasha    29 Aug 31 11:55 .ssh
```

5. Скопируйте файл *people_ages_titles.csv* в ранее созданную папку *input*:

```
$ sudo cp ~/people_ages_titles.csv /srv/airflow/home/input
```

Убедитесь, что операция выполнилась успешно, используя следующую команду:

```
$ ls -la /srv/airflow/home/input
```

Вывод на консоль приведен ниже:

```
total 20
drwxr-xr-x  2 root  root     36 Dec 18 14:03 .
drwxr-xr-x. 6 50000 50000   126 Dec 18 13:58 ..
-rw-r--r--  1 root  root  17661 Dec 18 14:03 people_ages_titles.csv
```

6. Измените владельца созданных папок *input* and *output*, включая их содержимое, на `airflow`. В противном случае возможно возникновение ошибок доступа к файлам в процессе исполнения DAG. Для назначения владельца используйте следующие команды:

```
$ sudo chown -R 50000:50000 /srv/airflow/home/input
$ sudo chown -R 50000:50000 /srv/airflow/home/output
```

Для проверки результата воспользуйтесь следующей командой:

```
$ ls -la /srv/airflow/home/
```

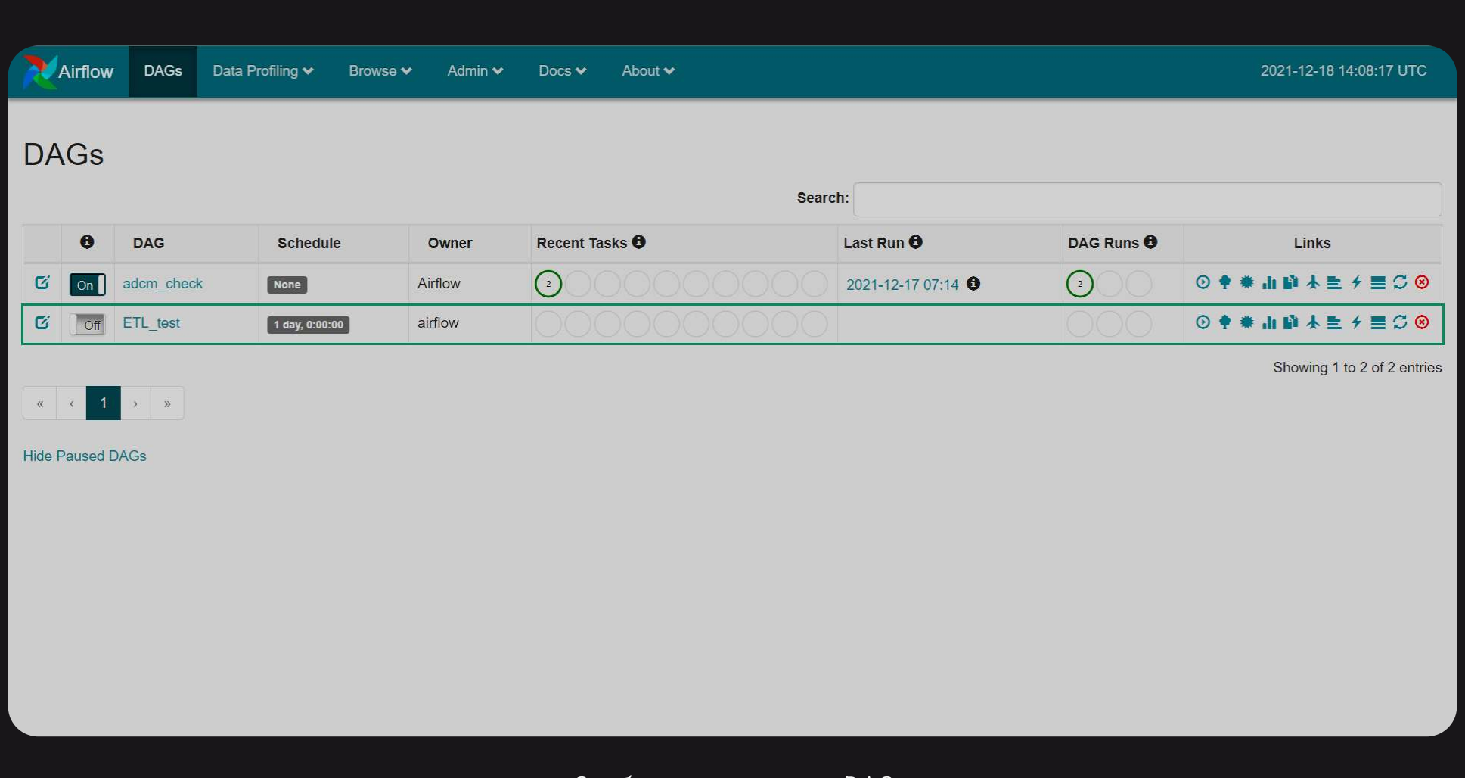
Вывод на консоль приведен ниже:

```
total 48
drwxr-xr-x. 6 50000 50000   126 Dec 18 13:58 .
drwxr-xr-x. 4 root  root     35 Dec 16 07:17 ..
-rw-r--r--. 1 root  root  37224 Dec 16 07:18 airflow.cfg
-rw-r--r--  1 50000 50000     3 Dec 18 11:07 airflow-webserver.pid
drwxr-xr-x. 3 50000 50000    65 Dec 18 13:11 dags
drwxr-xr-x  2 50000 50000    36 Dec 18 14:03 input
drwxr-xr-x. 8 50000 50000   137 Dec 18 14:12 logs
drwxr-xr-x  2 50000 50000     6 Dec 18 13:58 output
-rw-r--r--. 1 50000 50000  2528 Dec 16 07:19 unittests.cfg
```

Шаг 2. Запуск DAG через Web-интерфейс

Для запуска DAG и проверки результатов воспользуйтесь следующими шагами:

1. Откройте домашнюю страницу Web-интерфейса Airflow. Если все настроено правильно, вы увидите состояние графа во вкладке ***DAGs***. В противном случае будут отображены ошибки.

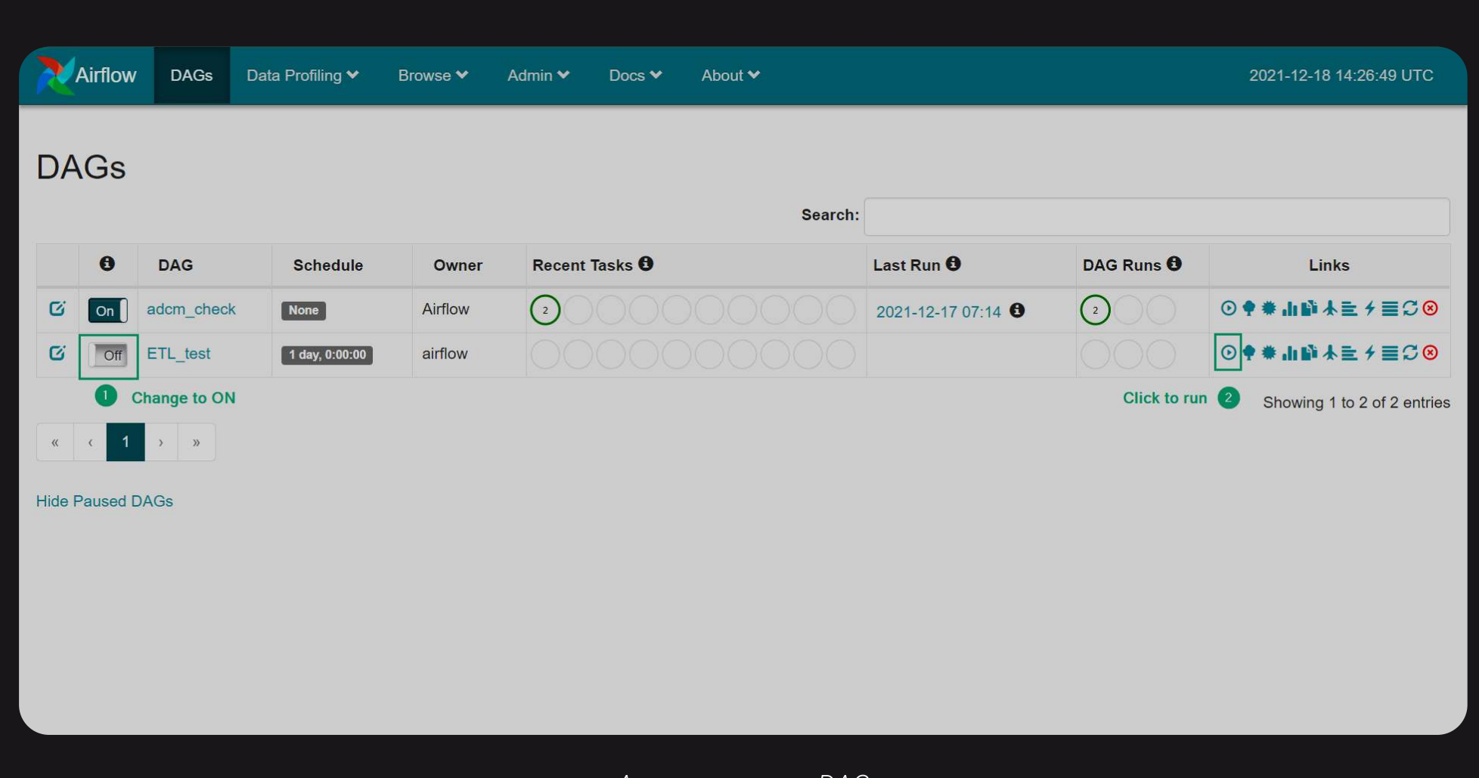


Airflow
DAGs
Data Profiling
Browse
Admin
Docs
About
2021-12-18 14:08:17 UTC
DAGs
Search:
Showing 1 to 2 of 2 entries
Hide Paused DAGs

		DAG	Schedule	Owner	Recent Tasks	Last Run	DAG Runs	Links
	On	adcm_check	None	Airflow	2	2021-12-17 07:14	2	
	Off	ETL_test	1 day, 0:00:00	airflow				

Отображение нового DAG

2. Активизируйте DAG с помощью одного из способов:
   - Нажмите переключатель и ожидайте запуска согласно настроенному расписанию.
   - Откройте DAG и щелкните на иконке ***Trigger Dag*** для немедленного запуска DAG.

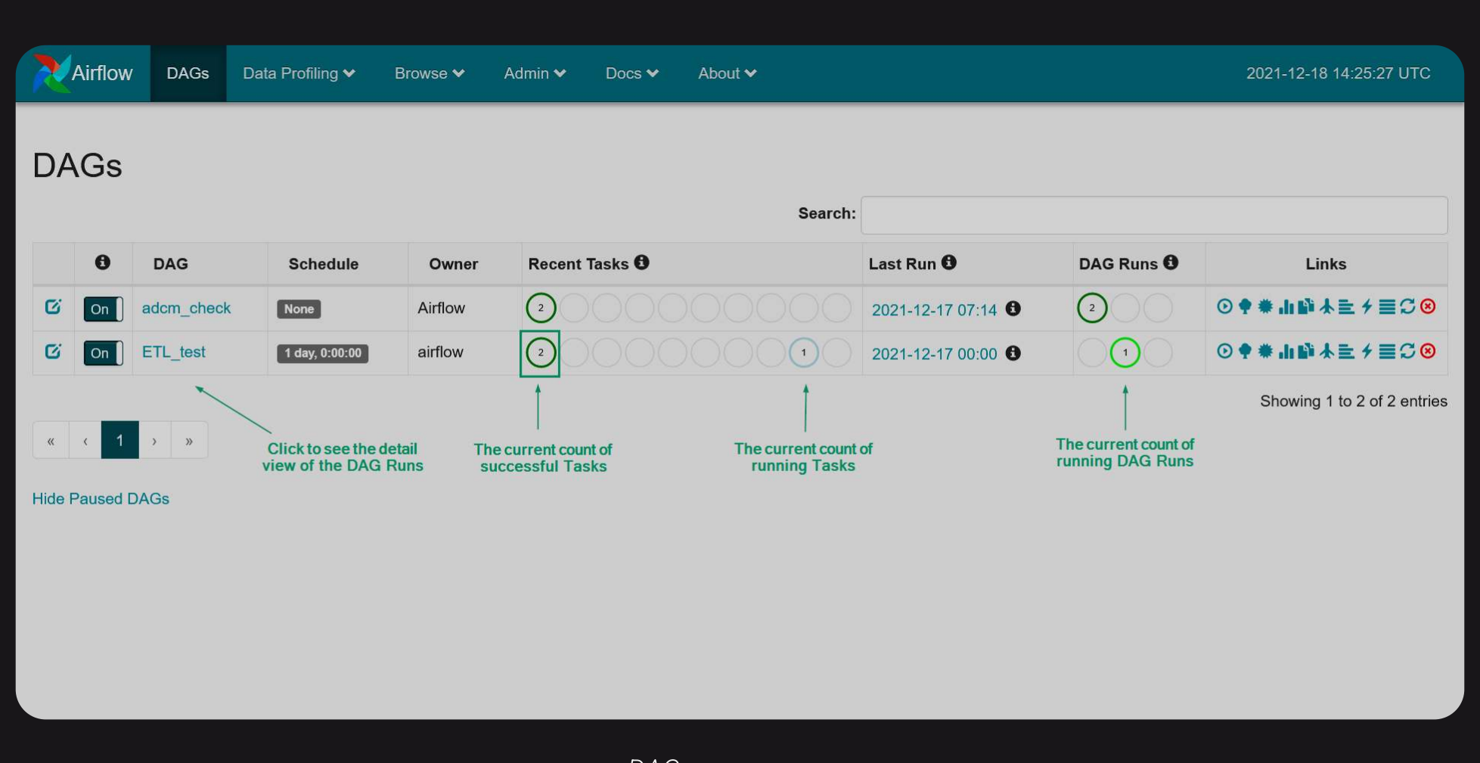


Airflow
DAGs
Data Profiling
Browse
Admin
Docs
About
2021-12-18 14:26:49 UTC
DAGs
Search:
① Change to ON
Click to run ②
Showing 1 to 2 of 2 entries
Hide Paused DAGs

		DAG	Schedule	Owner	Recent Tasks	Last Run	DAG Runs	Links
	On	adcm_check	None	Airflow	2	2021-12-17 07:14	2	
	Off	ETL_test	1 day, 0:00:00	airflow				

Активирование DAG

3. После запуска DAG вы можете увидеть информацию о его состоянии на домашней странице: количество успешных (successful), неудачных (failed), активных (running) и других запусков DAG, а также состояния экземпляров задач. Для просмотра подробной информации о запусках конкретного DAG (DAG runs), щелкните на названии DAG.

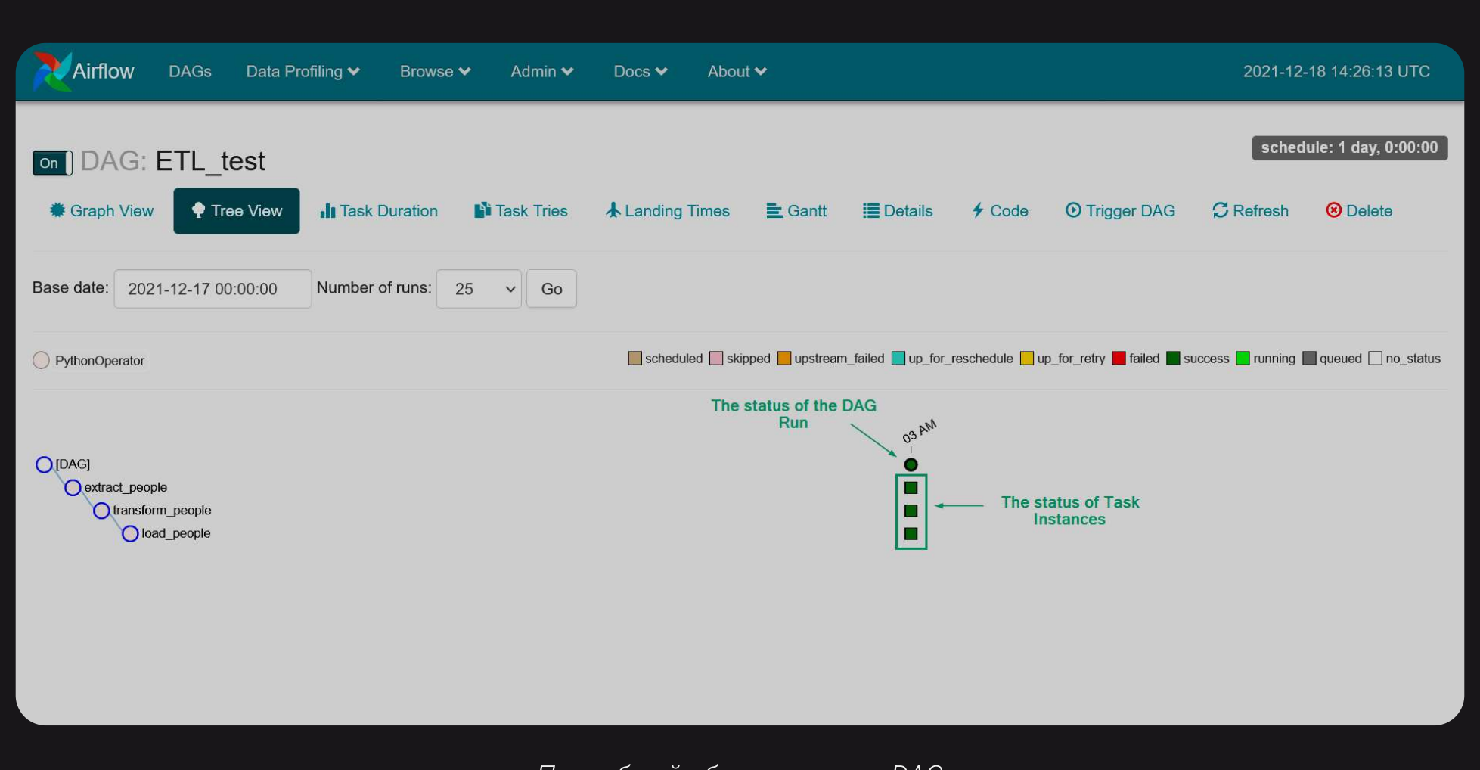


Airflow
DAGs
Data Profiling
Browse
Admin
Docs
About
2021-12-18 14:25:27 UTC
DAGs
Search:
Showing 1 to 2 of 2 entries
Click to see the detail view of the DAG Runs
The current count of successful Tasks
The current count of running Tasks
The current count of running DAG Runs
Hide Paused DAGs

		DAG	Schedule	Owner	Recent Tasks	Last Run	DAG Runs	Links
	On	adcm_check	None	Airflow	2	2021-12-17 07:14	2	
	On	ETL_test	1 day, 0:00:00	airflow	2 1	2021-12-17 00:00	1	

DAG выполняется

4. Просматривая подробную информацию, вы можете увидеть состояния всех запусков DAG и экземпляров задач. При необходимости измените режим просмотра на ***Graph View*** для анализа длительности выполнения задач или выполнения других действий, доступных на текущей странице.

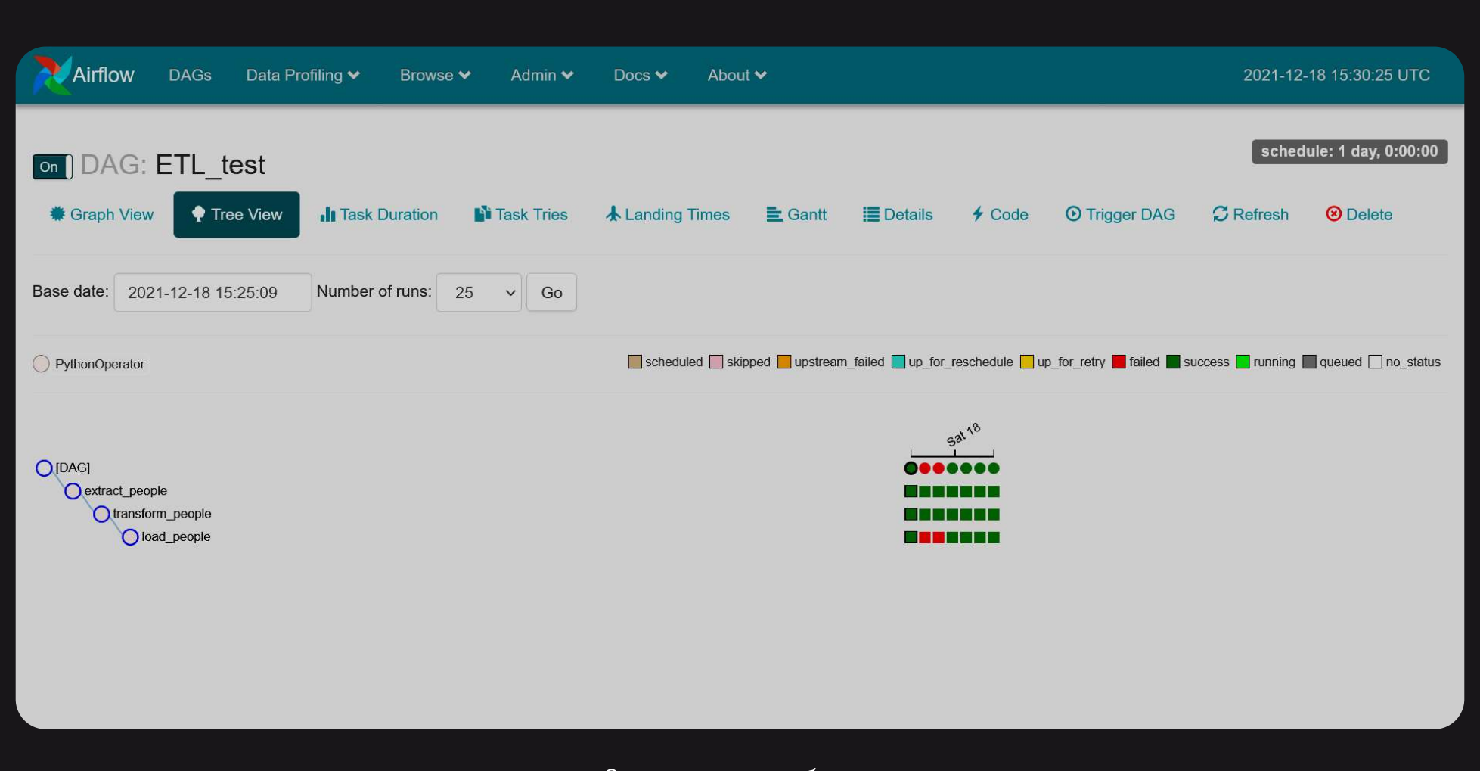


Airflow
DAGs
Data Profiling
Browse
Admin
Docs
About
2021-12-18 14:26:13 UTC
On
DAG: ETL_test
schedule: 1 day, 0:00:00
Graph View
Tree View
Task Duration
Task Tries
Landing Times
Gantt
Details
Code
Trigger DAG
Refresh
Delete
Go
PythonOperator
scheduled
skipped
upstream_failed
up_for_reschedule
up_for_retry
failed
success
running
queued
no_status
The status of the DAG Run
03 AM
The status of Task Instances
[DAG]
extract_people
transform_people
load_people

Base date: 2021-12-17 00:00:00 Number of runs: 25

Подробный обзор запусков DAG

Если в процессе исполнения DAG появятся ошибки, страница будет выглядеть как представлено ниже.



Airflow
DAGs
Data Profiling
Browse
Admin
Docs
About
2021-12-18 15:30:25 UTC
On
DAG: ETL_test
schedule: 1 day, 0:00:00
Graph View
Tree View
Task Duration
Task Tries
Landing Times
Gantt
Details
Code
Trigger DAG
Refresh
Delete
Go
PythonOperator
scheduled
skipped
upstream_failed
up_for_reschedule
up_for_retry
failed
success
running
queued
no_status
Sat 18
[DAG]
extract_people
transform_people
load_people

Base date: 2021-12-18 15:25:09 Number of runs: 25

Страница с ошибками

Шаг 3. Проверка результатов

В завершение, проверим исполнил ли DAG все задачи корректно. Убедитесь, что выходной файл *loaded_people.csv* появился в папке *output*:

1. Проверьте содержимое папки *output*:

```
$ ls -la /srv/airflow/home/output/
```

Вывод на консоль приведен ниже:

```
total 4
drwxr-xr-x  2 50000 50000  31 Dec 18 14:25 .
drwxr-xr-x. 6 50000 50000 126 Dec 18 13:58 ..
-rw-r--r--  1 50000 50000 832 Dec 18 14:25 loaded_people.csv
```

2. Получите выходные данные:

```
$ cat /srv/airflow/home/output/loaded_people.csv
```

Данные выглядят следующим образом:

```
Name,Age
Arnold Bettie,18
Campbell Todd,18
Thompson Warren,18
Stokes Mittie,18
Davis Alan,18
Vaughn Belle,18
Munoz Ricardo,18
Norton Bertha,18
McGee Isabelle,18
Bailey Cameron,18
James Loretta,18
Myers Ricky,18
Torres Elsie,18
Phelps Rena,18
Hill Katharine,18
Hodges Abbie,18
Fitzgerald Timothy,18
Miller Francis,18
Banks Catherine,18
Casey Louise,18
```

Работа с TaskFlow

Елена Костюченко

Содержание

Назад к содержанию

- Использование TaskFlow с традиционными операторами
  - Передача данных между задачами
  - Кросс-функциональные зависимости
- Структура DAG в TaskFlow

TaskFlow API, добавленный в Airflow 2, представляет новый подход к созданию DAG, который позволяет передавать параметры от функции к функции напрямую и помогает сделать код DAG более лаконичным и удобным для чтения.

В предыдущих версиях Airflow наиболее распространенным способом написания DAG было использование операторов Python (PythonOperator). В традиционном подходе к созданию DAG задачи описывались следующим образом:

```
def extract_people(path: str, **context) -> None:
    extracted_people = pd.read_csv(path,sep=',',header=1)
    context['ti'].xcom_push(key='extracted_people', value=extracted_people)

def load_people(path: str, **context) -> None:
    extracted_people = context['ti'].xcom_pull(key='extracted_people', task_ids=['extract_people'])[0]
    loaded_people = pd.DataFrame(extracted_people[0])
    loaded_people.to_csv(path, index=None)

task_extract_people = PythonOperator(
    task_id='extract_people',
    python_callable=extract_people,
    op_kwargs={'path': '/opt/airflow/input/people_ages_titles.csv'},
    dag=dag,
    provide_context=True
)

task_load_people = PythonOperator(
    task_id='load_people',
    python_callable=load_people,
    op_kwargs={'path': '/opt/airflow/output/loaded_people.csv'},
    dag=dag,
    provide_context=True
)

task_extract_people >> task_load_people
```

Это упрощенная версия традиционного DAG. Полный пример доступен в статье Создание простого DAG.

Пример выше иллюстрирует ограничения традиционного подхода, которые могут привести к раздутому коду:

- Необходимость использования объекта контекста для передачи данных в следующую задачу через сервис XCom.
- Необходимость отдельно определять операторы, функции и входные параметры, которые будут использованы в задачах.

В TaskFlow API есть специальные декораторы, которые убирают необходимость использовать традиционные операторы и обеспечивают передачу данных между задачами с помощью функций того же XCom, но без необходимости их явного вызова.

При создании декорированной функции не нужно указывать `task_id` или `python_callable`. TaskFlow использует имя функции в качестве идентификатора задачи, но вы можете также передать `task_id` в декораторе, чтобы задать другое имя.

Airflow поддерживает декораторы для задач, групп задач и DAG. В парадигме TaskFlow тот же пример будет выглядеть следующим образом:

```
@task(task_id="extract_people", retries=2) ①
def extract_people(path: str) -> pd.DataFrame: ②
    extracted_people = pd.read_csv(path, sep=',', header=1)
    return extracted_people ③

@task()
def transform_people(extracted_people: pd.DataFrame) -> list:
    transformed_people = []
    for person in extracted_people.values:
        if int(person[1]) <= 18:
            transformed_people.append({
                'Name': person[0],
                'Age': person[1]
            })
    return transformed_people

extracted_data = extract_people(SOURCE_PATH) ④
transformed_data = transform_people(extracted_data)
```

① Определите задачу с помощью декоратора `@task`. При необходимости передайте нужные параметры задачи в декораторе.

② Определите функцию задачи.

③ Здесь больше нет необходимости использовать `xcom_pull` и `xcom_push`, поскольку возвращаемое функцией значение автоматически передается другим задачам через XCom, когда определяются зависимости.

④ Определите зависимости задач, вызвав функции задач.

Использование TaskFlow с традиционными операторами

В то время, как традиционные операторы предлагают более детальный контроль над конвейерами данных, TaskFlow обеспечивает более четкие переходы в рабочих процессах и более читаемый код.

В Airflow вы можете использовать как традиционные операторы, так и API TaskFlow.

Передача данных между задачами

Чтобы перенести данные из декорированной задачи в традиционную, используйте атрибут `.output` объекта задачи при вызове декорированной функции.

Например:

```
def traditional_function():
    return 10

@task
def taskflow_function(x):
    return x + 5

sum_task = PythonOperator(
    task_id="sum_task",
    python_callable=traditional_function,
)

taskflow_function(sum_task.output)  # the task will return 15
```

Чтобы перенести данные из традиционной задачи в декорированную, укажите вызванную декорированную задачу непосредственно в параметре традиционного оператора.

Например:

```
@task
def taskflow_function():
    return 10

def traditional_function(x):
    return x + 5

sum_task = PythonOperator(
    task_id="sum_task",
    python_callable=traditional_function,
    op_args=[taskflow_function()]
)

traditional_function >> taskflow_function()

# sum_task will return 15
```

Кроме того, существует возможность передавать данные с помощью `xcom_pull` и `xcom_push` между традиционными и декорированными функциями.

Поскольку декораторы TaskFlow используют ту же традиционную функциональность XCom для сохранения возвращаемых значений функции, можно извлечь данные из декорированной задачи, используя тот же синтаксис `xcom_pull`.

Например:

```
@task
def taskflow_function():
    return 10

def traditional_function(**kwargs):
    ti = kwargs["ti"]
    received_data = ti.xcom_pull(task_ids="taskflow_function")
    print(received_data)

traditional_task = PythonOperator(
    task_id="traditional_task",
    python_callable=traditional_function,
)

traditional_function >> taskflow_function()

# sum_task will return 15
```

ВАЖНО

TaskFlow сохраняет возвращаемое значение функции только в том случае, если оно используется где-либо еще в DAG.

Кросс-функциональные зависимости

При использовании двух типов функций в одном DAG необходимо определить зависимости в соответствии с каждым типом функции: передача параметров от функции к функции — для TaskFlow, и использование операторов битового сдвига — для традиционных функций.

Например:

```
@task
def taskflow_function_10():
    return 10

@task
def taskflow_function_20(x):
    return x + 20

def traditional_function(x):
    return x + 30

sum_task = PythonOperator(
    task_id="sum_task",
    python_callable=traditional_function,
)

taskflow_function_20(taskflow_function_10()) >> traditional_function

# sum_task will return 60
```

Структура DAG в TaskFlow

Задачи в Airflow записываются как функции Python, декорированные `@task`. При этом сами задачи должны находиться внутри функции DAG, декорированной `@dag`.

Например:

```
from datetime import timedelta, datetime
import pandas as pd
from airflow.decorators import dag, task ①


SOURCE_PATH = '/opt/airflow/input/people_ages_titles.csv'
TARGET_PATH = '/opt/airflow/output/loaded_people.csv'

@dag(                                      ②
    schedule_interval=timedelta(days=1),
    start_date=datetime(2021, 12, 17),
    catchup=False,
    tags=['example'],
)
def etl_example():

    @task()                                ③
    def extract_people(path: str) -> pd.DataFrame:
        extracted_people = pd.read_csv(path, sep=',', header=1)
        return extracted_people

    @task()
    def transform_people(extracted_people: pd.DataFrame) -> list:
        transformed_people = []
        for person in extracted_people.values:
            if int(person[1]) <= 18:
                transformed_people.append({
                    'Name': person[0],
                    'Age': person[1]
                })
        return transformed_people

    @task()
    def load_people(transformed_people: list, path: str) -> None:
        loaded_people = pd.DataFrame(transformed_people)
        loaded_people.to_csv(path, index=None)

    extracted_data = extract_people(SOURCE_PATH)    ④
    transformed_data = transform_people(extracted_data)
    load_people(transformed_data, TARGET_PATH)

etl_dag = etl_example()    ⑤
```

① Импортируйте декораторы задач (task) и DAG.

② Создайте экземпляр DAG с помощью декоратора `@dag` и его функцию. По умолчанию имя функции используется в качестве DAG ID.

③ Внутри функции DAG определите функции задач.

④ Определите зависимости, чтобы обеспечить передачу данных между задачами.

⑤ Вызовите функцию DAG. Все декорированные функции в DAG должны быть вызваны в файле этого DAG, чтобы Airflow мог их зарегистрировать.

Использование сенсоров в Airflow

Елена Костюченко

Содержание

Назад к содержанию

Параметры сенсора
Работа с сенсорами
Создание кастомного сенсора

Сенсор в Airflow — это оператор, который используется, когда в DAG требуется дождаться выполнения условия для продолжения выполнения задач.

Условия срабатывания сенсора зависят от его типа. Airflow предоставляет набор стандартных операторов, которые позволяют, например, ждать истечения определенного времени (TimeDeltaSensor), ждать появления файла (FileSensor) или завершения задачи в другом DAG (ExternalTaskSensor). Когда условие сенсора выполняется, он может запустить последующие задачи.

Параметры сенсора

У сенсоров есть следующие базовые параметры:

- `poke_interval` — время, в течение которого задача должна ждать перед следующей проверкой условия. Значение по умолчанию — 60 секунд.
- `timeout` — максимальное время работы сенсора. Значение по умолчанию — 7 дней или 60 * 60 * 24 * 7 секунд. Когда время истечет, задача сенсора будет завершена. Значение должно быть меньше, чем дата следующего запуска DAG, чтобы избежать зависания задач.
- `mode` — индикатор того, как долго задача сенсора будет занимать рабочий слот. Может принимать одно из двух значений:
  - `poke` — сенсор всегда занимает слот, пока не завершится;
  - `reschedule` — сенсор занимает слот только во время проверок, а на время, указанное в `poke_interval`, планировщик освобождает слот.

Работа с сенсорами

Операторы сенсоров являются производными от класса `airflow.sensors.base` и наследуют его атрибуты. Airflow поддерживает несколько сенсоров, которые описаны в таблице ниже, но некоторые провайдеры предоставляют также дополнительные сенсоры.

Стандартные сенсоры Airflow

Название	Описание
airflow.sensors.bash	Выполняет команду/скрипт bash, который можно использовать в качестве настраиваемого сенсора. Команда должна возвращать `0` при успешном выполнении, а в остальных случаях — любое другое значение
airflow.sensors.date_time	Приостанавливает выполнение DAG до наступления определенной даты и времени. Главным преимуществом этого сенсора является идемпотентность для `target_time`. Он обрабатывает некоторые случаи, для которых TimeSensor и TimeDeltaSensor не подходят
airflow.sensors.external_task	Ожидает завершения задачи в другом DAG
airflow.sensors.filesystem	Ожидает появления файла или директории в системе. Если указанный путь — непустая директория, то сенсор вернет значение `True`
airflow.sensors.python	Выполняет произвольный Python-код и ждет его завершения. Рекомендуется использовать декоратор `@task.sensor` вместо классического PythonSensor для выполнения Python-функций
airflow.sensors.time_delta	Находится в ожидании определенное время
airflow.sensors.time_sensor	Ожидает наступления указанного времени дня
airflow.sensors.weekday	Ожидает определенного дня недели. Например, если день выполнения задачи — `2018-12-22` (суббота), а в сенсор передается значение `FRIDAY` (пятница), задача будет ждать до следующей пятницы

Более подробное описание каждого сенсора доступно в документации Airflow.

Пример возможного использования сенсора в DAG:

```
from datetime import datetime, timedelta

from airflow import DAG
from airflow.sensors.time_delta import TimeDeltaSensor, TimeDeltaSensorAsync ①

dag = DAG(
    dag_id="sensor_example",
    schedule=None,
    start_date=datetime(2023, 6, 4),
)

time_sensor_example = TimeDeltaSensor( ②
   task_id='time_sensor_example',
   dag=dag,
   delta=timedelta(minutes=1), ③
   poke_interval=10, ④
   timeout=30, ⑤
   soft_fail=True, ⑥
)
```

① Импорт операторов сенсоров. `TimeDeltaSensorAsync` — это асинхронная версия оператора `TimeDeltaSensor`.

② Создание экземпляра сенсора. Если режим сенсора не указан, по умолчанию используется `poke`.

③ Параметр `delta` указывает, как долго ждать, прежде чем задача может быть отмечена как выполненная.

④ Параметр `poke_interval` определяет, как часто сенсор проверяет условие (в секундах).

⑤ Параметр `timeout` определяет, как долго сенсор будет работать.

⑥ Параметр `soft_fail` отмечает задачу сенсора как `SKIPPED`, если она не выполнена.

> Чтобы избежать блокировки (deadlock) задач и других проблем с производительностью, рассмотрите следующие рекомендации по работе с сенсорами:
>
> - Устанавливайте `timeout` для сенсора, который подойдет для вашей задачи. Значение по умолчанию для этого параметра может быть слишком большим в некоторых случаях.
> - Используйте режим `reschedule` для долго работающих сенсоров, чтобы они не занимали рабочий слот постоянно, а режим `poke` — для задач с частой проверкой условий (т.е. менее 5 минут), чтобы избежать перегрузки планировщика.
> - Убедитесь, что параметр `poke_interval` подходит для конкретной задачи. Значение по умолчанию предполагает очень частые проверки условий.

Создание кастомного сенсора

Чтобы создать кастомный сенсор, необходимо унаследовать класс `BaseSensorOperator`. В примере ниже демонстрируется создание нового сенсора, который проверяет наличие файла в директории.

1. Добавьте файл Python, который будет содержать код сенсора, в каталог *dags* домашней директории Airflow.

```
$ sudo vi /opt/airflow/dags/custom_sensor.py
```

2. Напишите код для сенсора:

```
import os
from airflow.sensors.base import BaseSensorOperator ①

class MyCustomFileSensor(BaseSensorOperator): ②
    def __init__(self, filepath, *args, **kwargs): ③
        self.filepath = filepath
        super().__init__(*args, **kwargs)

    def poke(self, context): ④
        print(f'checking if a file {self.filepath} exists...')
        return os.path.exists(self.filepath) ⑤
```

① Импорт базового класса сенсора.

② Наследование базового класса.

③ Добавление дополнительного аргумента `filepath` и наследование аргументов родительского класса.

④ Переопределение метода `poke`.

⑤ Python-логика для проверки существования файла.

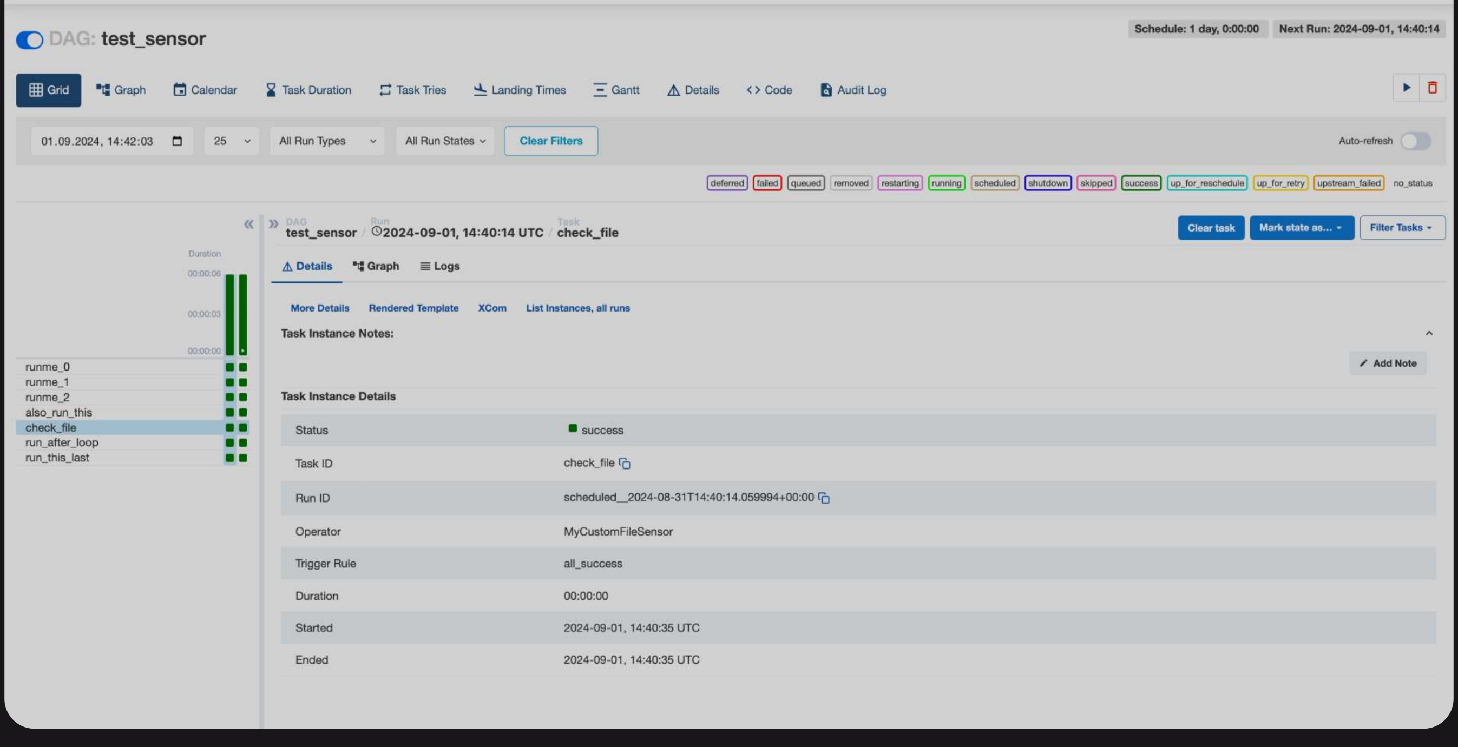
3. Вызовите новый сенсор внутри любого DAG:

```
from custom_sensor import MyCustomFileSensor ①

check_file = MyCustomFileSensor(    ②
    task_id='check_file',
    filepath='/opt/airflow/test_file.txt',
    poke_interval=10,
    timeout=60,
    dag=dag
)
```

① Импорт класса кастомного сенсора.

② Вызов функции сенсора с нужными параметрами.

4. Перейдите в веб-интерфейс Airflow, чтобы запустить DAG и убедиться, что функция кастомного сенсора работает корректно.


Airflow
DAGs
Datasets
Security
Browse
Admin
Docs
14:42 UTC
AA
DAG: test_sensor
Schedule: 1 day, 0:00:00
Next Run: 2024-09-01, 14:40:14
Grid
Graph
Calendar
Task Duration
Task Tries
Landing Times
Gantt
Details
Code
Audit Log
01.09.2024, 14:42:03
25
All Run Types
All Run States
Clear Filters
Auto-refresh
deferred
failed
queued
removed
restarting
running
scheduled
shutdown
skipped
success
up_for_reschedule
up_for_retry
upstream_failed
no_status
Duration
00:00:06
00:00:03
00:00:00
runme_0
runme_1
runme_2
also_run_this
check_file
run_after_loop
run_this_last
DAG
Run
Task
test_sensor
2024-09-01, 14:40:14 UTC
check_file
Clear task
Mark state as...
Filter Tasks
Details
Graph
Logs
More Details
Rendered Template
XCom
List Instances, all runs
Task Instance Notes:
Add Note

Task Instance Details

Status	success
Task ID	check_file
Run ID	scheduled__2024-08-31T14:40:14.059994+00:00
Operator	MyCustomFileSensor
Trigger Rule	all_success
Duration	00:00:00
Started	2024-09-01, 14:40:35 UTC
Ended	2024-09-01, 14:40:35 UTC

Airflow UI: страница DAG

На странице DAG видно, что задача `check_file` с классом `MyCustomFileSensor` успешно завершена.

Кастомизация расписания DAG

Елена Костюченко

Содержание

Назад к содержанию

Обзор

Cron-расписания

Timedelta-расписания

Расписание событий

Кастомное расписание

Обзор

Airflow предоставляет возможность запускать конвейеры данных по расписанию, позволяя задавать временной интервал (data interval) и логическую дату (logical date) каждого запуска DAG.

Расписание DAG можно определить с помощью одного из следующих инструментов:

- встроенный объект timedelta модуля Python `datetime`;
- Linux Cron;
- кастомные варианты классов `timetable`.

Все параметры расписания, такие как интервал, дата, время и другие, определяются во внутреннем расписании DAG (timetable). DAG, запланированные с помощью выражения cron или объекта `timedelta`, также внутренне преобразуются в расписание.

В Airflow доступны расписания по умолчанию, которые обрабатывают выражения cron или `timedelta`, но для DAG с более специфичными требованиями к расписанию может потребоваться кастомный класс расписания DAG.

Примеры случаев, когда требуется кастомный класс `timetable`:

- Конвейеры данных запускаются в разное время каждый день.
- Расписание DAG требует использования другого формата календаря.
- DAG запускается с перекрывающимися интервалами данных. Например, запуск DAG должен охватывать период предыдущих запусков.
- DAG запускается с неравными временными интервалами.

Cron-расписания

Выражения Cron в Airflow следуют стандартному синтаксису cron, который состоит из пяти полей, разделенных пробелами, где каждое поле может принимать значения, перечисленные в таблице ниже.

Cron-выражения задаются в формате `<1> <2> <3> <4> <5>`, где:

- `<1>` — минуты в диапазоне: 0-59.
- `<2>` — час в диапазоне: 0-23.
- `<3>` — день месяца в диапазоне: 1-31.
- `<4>` — месяц в году. Диапазон от 1 до 12, где 1 — это январь.
- `<5>` — числовой индикатор дня недели. Каждый день представлен числом в диапазоне от 0 до 6, где 0 — это воскресенье.

Например, расписание DAG, представленное выражением cron `0 18 * * 5`, запускается каждую пятницу в 18.00.

Возможные значения	Пример
Единственное значение	5
Диапазон значений, определяемый с помощью тире	1-5
Список значений, разделенных запятыми	1,3,5
Каждое возможное значение (звездочка)	*

В Airflow предусмотрены два метода для обработки cron-расписаний: `CronTriggerTimetable` и `CronDataIntervalTimetable`. Оба расписания запускают DAG в одно и то же время, но временная метка для `run_id` у DAG будет разной.

Например, если DAG с расписанием, определенным как `0 0 * * *` (12.00 каждый день), включается 31 января в 15.00, `CronTriggerTimetable` запустит DAG 1 февраля в 12.00, но `CronDataIntervalTimetable` запустит DAG немедленно, поскольку запуск DAG для ежедневного временного интервала, начинающегося 31 января в 12.00, еще не произошел.

Пример DAG для `CronTriggerTimetable`, который запускает DAG в 1 час ночи в среду:

```python
from datetime import timedelta
from airflow import DAG
from airflow.timetables.trigger import CronTriggerTimetable

def cron_timetable(*args, **kwargs):
    return CronTriggerTimetable(kwargs['cron_expression'])

dag = DAG(
    dag_id='my_dag',
    timetable=cron_timetable(cron_expression='0 1 * * 3', timezone="UTC"),
    start_date=days_ago(2),
    tags=['example'],
)

@dag(schedule=CronTriggerTimetable("0 1 * * 3", timezone="UTC"), ...)
def example_dag():
    pass
```

При использовании `CronDataIntervalTimetable` можно указать статический интервал данных с помощью аргумента `interval`. Он должен использовать объекты `datetime.timedelta` или `dateutil.relativedelta.relativedelta`.

Если статический интервал указан, временной интервал запуска DAG охватывает указанную продолжительность и заканчивается временем срабатывания (trigger time).

Например:

```python
from datetime import timedelta
from airflow.timetables.trigger import CronTriggerTimetable

@dag(
    schedule=CronTriggerTimetable(
        "0 18 * * 5",
        timezone="UTC",
        interval=timedelta(days=4, hours=9),
    ),
    ...,
)
def example_dag():
    pass
```

DAG из приведенного выше примера будет запускаться каждую пятницу в 18.00.

Timedelta-расписания

Расписания, использующие timedelta, позволяют запускать DAG с указанным временным интервалом. Это расписание опирается на значение временного интервала, предоставленное пользователем, и не обязательно согласует даты выполнения с началом дня или часа.

Вы можете использовать timedelta-расписание, указав значения `datetime.timedelta` или `dateutil.relativedelta.relativedelta` в параметре `schedule`.

Например:

```python
@dag(schedule=datetime.timedelta(minutes=30))
def example_dag():
    pass
```

Расписание событий

Расписание событий позволяет запускать DAG в указанные даты и время. Список дат должен быть небольшого размера, поскольку он должен загружаться каждый раз при обработке DAG.

Например:

```python
from airflow import DAG
import pendulum
from airflow.timetables.events import EventsTimetable

@dag(
    schedule=EventsTimetable(
        event_dates=[
            pendulum.datetime(2024, 4, 5, 8, 27, tz="America/Vancouver"),
            pendulum.datetime(2024, 4, 17, 8, 27, tz="America/Vancouver"),
            pendulum.datetime(2024, 4, 22, 20, 50, tz="America/Vancouver"),
        ],
        description="Example dates",
        restrict_to_events=False,
    ),
    ...,
)
def example_dag():
    pass
```

Кастомное расписание

Кастомные расписания позволяют задавать произвольный график запусков DAG в коде Python, когда расписаний на основе cron и `timedelta` недостаточно.

Чтобы создать кастомное расписание, необходимо:

- создать класс, который наследует класс расписания (`Timetable`);
- реализовать методы `next_dagrun_info()` и `infer_manual_data_interval()`, оба из которых возвращают объект `DataInterval`;
- зарегистрировать расписание как часть плагина Airflow.

Метод `next_dagrun_info()` возвращает интервал данных для стандартного расписания DAG и описывает логику для его расчета. Этот метод также содержит логику для расчета параметров `start_date`, `end_date` и `catchup`.

Метод `infer_manual_data_interval()` описывает, как определяется интервал работы DAG в случае ручного запуска.

В примере ниже показано, как написать DAG с кастомным расписанием.

Чтобы создать DAG с кастомным расписанием:

1. В директории *plugins* домашнего каталога Airflow добавьте новый файл Python, который будет содержать кастомное расписание:

```
$ sudo vi /opt/airflow/plugins/custom_timetable.py
```

2. В файле *custom_timetable.py* определите структуру расписания:

```python
from datetime import timedelta
from typing import Optional
from pendulum import Date, DateTime, Time, timezone
from airflow.plugins_manager import AirflowPlugin
from airflow.timetables.base import DagRunInfo, DataInterval, TimeRestriction, Timetable ①

UTC = timezone("UTC") ②

class CustomTimetable(Timetable): ③
    pass

class CustomTimetablePlugin(AirflowPlugin): ④
    name = "custom_timetable_plugin"
    timetables = [CustomTimetable]
```

① Импорт зависимостей.

② Объявление глобальных переменных.

③ Наследование базового класса.

④ Регистрация расписания как плагина Airflow.

3. В классе `CustomTimetable` напишите реализацию метода `next_dagrun_info()`:

```python
class CustomTimetable(Timetable):

    def next_dagrun_info(
        self,
        *,
        last_automated_data_interval: Optional[DataInterval],
        restriction: TimeRestriction,
    ) -> Optional[DagRunInfo]:
        if last_automated_data_interval is not None:  ①
            last_start = last_automated_data_interval.start
            delta = timedelta(days=1)
            if last_start.hour == 6:
                next_start = last_start.set(hour=16, minute=30).replace(tzinfo=UTC)
                next_end = (last_start+delta).replace(tzinfo=UTC)
            else:
                next_start = (last_start+delta).set(hour=6, minute=0).replace(tzinfo=UTC)
                next_end = (last_start+delta).replace(tzinfo=UTC)
        else: ②
            next_start = restriction.earliest
            if next_start is None:
                return None
            if not restriction.catchup: ③
                next_start = max(next_start, DateTime.combine(Date.today(), Time.min).replace(tzinfo=UTC))
            next_start = next_start.set(hour=6, minute=0).replace(tzinfo=UTC)
            next_end = next_start.set(hour=16, minute=30).replace(tzinfo=UTC)
        if restriction.latest is not None and next_start > restriction.latest: ④
            return None
        return DagRunInfo.interval(start=next_start, end=next_end)
```

① Описание логики для случая, когда уже был предыдущий запуск DAG.

② Описание логики первого запуска DAG.

③ Реализация поведения catchup.

④ Проверка того, что следующий запуск DAG не начнется вне расписания.

4. В классе `CustomTimetable` опишите логику планирования запусков в случае ручного запуска в методе `infer_manual_data_interval()`:

```python
class CustomTimetable(Timetable):

    def infer_manual_data_interval(self, run_after: DateTime) -> DataInterval:
        delta = timedelta(days=1)
        if run_after >= run_after.set(hour=6, minute=0) and run_after <= run_after.set(hour=16, minute=30):
            start = (run_after-delta).set(hour=16, minute=30, second=0).replace(tzinfo=UTC)
            end = run_after.set(hour=6, minute=0, second=0).replace(tzinfo=UTC)
        elif run_after >= run_after.set(hour=16, minute=30) and run_after.hour <= 23:
            start = run_after.set(hour=6, minute=0, second=0).replace(tzinfo=UTC)
            end = run_after.set(hour=16, minute=30, second=0).replace(tzinfo=UTC)
        else:
            start = (run_after-delta).set(hour=6, minute=0).replace(tzinfo=UTC)
            end = (run_after-delta).set(hour=16, minute=30).replace(tzinfo=UTC)
        return DataInterval(start=start, end=end)
```

5. Используйте кастомное расписание в DAG:

```python
from custom_timetable import CustomTimetable ①

@dag(
    dag_id="example_dag",
    start_date=datetime(2024, 3, 10),
    schedule=CustomTimetable(), ②
    default_args={
        "retries": 1,
        "retry_delay": duration(minutes=3),
    },
    catchup=True
)
```

① Импорт кастомного расписания.

② Вызов кастомного класса для использования его в качестве расписание запуска DAG.

Дополнительную информацию о запуске DAG можно получить в статье Создание простого DAG.

ВАЖНО

Добавление или обновление плагина Airflow, включая пользовательские расписания, требует перезапуска Airflow Scheduler и веб-сервера.

Добавление кастомных операторов и хуков в Airflow

Елена Костюченко

Содержание

Назад к содержанию

Обзор
Создание кастомного оператора
Создание кастомного хука

Обзор

Airflow предоставляет стандартные операторы, которые позволяют, например, выполнять команды bash, вызывать произвольные функции Python, создавать соединение с внешним приложением или проверять выполнение определенного условия.

Airflow предоставляет "из коробки" большое количество операторов, но в некоторых случаях необходимо изменить функцинальность существующего оператора или создать новый.

В этой статье описывается, как создавать пользовательские операторы и хуки. О создании кастомных сенсоров в Airflow можно прочитать в статье Использование сенсоров в Airflow.

При создании любого оператора Airflow необходимо:

1. Написать код для нового оператора в методе `__init__`.
2. Наследовать класс Airflow, который нужно переписать или изменить, переопределив методы в производном классе:
   - В конструкторе (constructor) определите параметры нового оператора.
   - В методе `execute` напишите код, который будет выполняться при вызове оператора.
3. Поместите файл с оператором в директорию, которая присутствует в `PYTHONPATH` Airflow. По умолчанию это директории *dags*, *plugins* и *config*.

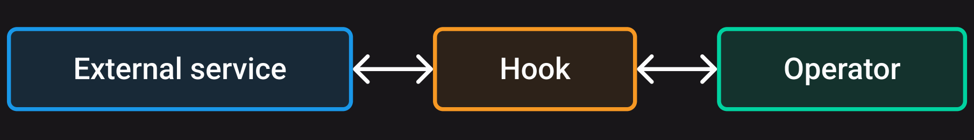
В примерах ниже показывается, как написать кастомные операторы и хуки, руководствуясь этими шагами.

Разделение логики между операторами и хуками

Оператор должен содержать бизнес-логику задачи и состояния в памяти. Если вам необходимо создать соединение с внешним сервисом (API, базой данных и т.д.), лучше написать его как хук.

Хук не должен содержать логику оператора, а только предоставлять методы, которые затем будут использоваться оператором для взаимодействия с внешним сервисом.

Разделив логику на два компонента, можно избежать дублирования кода и создать несколько операторов, которые используют один и тот же хук.



Взаимодействие операторов и хуков

Создание кастомного оператора

Ниже приведен пример создания кастомного оператора на основе класса `airflow.models.baseoperator.BaseOperator`:

1. В директории *dags* в домашнем каталоге Airflow добавьте файл Python, который будет содержать код оператора:

```
$ sudo vi /opt/airflow/dags/custom_operator.py
```

2. Добавьте код оператора:

```
from airflow.models.baseoperator import BaseOperator  ①

class CustomOperator(BaseOperator):  ②
    def __init__(self, name: str, **kwargs) -> None:  ③
        super().__init__(**kwargs)
        self.name = name

    def execute(self, context):  ④
        message = f"Hello, {self.name}!"
        print(message)
        return message
```

① Импортируйте базовый класс оператора.

② Наследуйте базовый класс в своем операторе.

③ Добавьте дополнительные аргументы и сохраните аргументы родительского класса.

④ Переопределите метод `execute`, задав желаемую функциональность.

3. Вызовите кастомный оператор в DAG:

```
from custom_operator import CustomOperator ①

print_message = CustomOperator(    ②
    task_id='print_message',
    name='CustomOperator',
    poke_interval=10,
    timeout=60,
    dag=dag
)
```

① Импортируйте класс кастомного оператора.

② Вызовите функцию оператора.

Создание кастомного хука

Процесс создания кастомного хука отличается от процесса создания кастомного оператора тремя шагами:

- Хук должен быть унаследован от класса другого хука, например, класса `BaseHook` или любого другого.
- В код хука можно включить метод `.get_conn()`, оборачивающий вызов метода `.get_connection()`, чтобы получить информацию о соединении Airflow.
- Должно существовать соединение (Airflow connection), которое будет использоваться хуком.

Ниже приведен пример того, как создать пользовательскую функцию хука, которая инициирует новое соединение с внешним сервисом:

1. В директории *dags* домашнего каталога Airflow добавьте файл Python, который будет содержать кастомный хук:

```
$ sudo vi /opt/airflow/dags/custom_hook.py
```

2. Добавьте код хука:

```
from airflow.hooks.base import BaseHook ①

class CustomHook(BaseHook): ②

    conn_name_attr = "custom_conn_id" ③
    default_conn_name = "custom_conn_default" ④
    conn_type = "http" ⑤
    hook_name = "CustomHook" ⑥

    def __init__(
        self, custom_conn_id: str = default_conn_name, *args, **kwargs
    ) -> None: ⑦
        super().__init__(*args, **kwargs) ⑧
        self.custom_conn_id = custom_conn_id ⑨
        self.get_conn() ⑩

    def get_conn(self):
        conn_id = getattr(self, self.conn_name_attr)
        conn = self.get_connection(conn_id)
        return conn
```

① Импортируйте базовый класс хука.

② Определите класс для наследования от базового класса.

③ Добавьте параметр, который сохраняет ID соединения.

④ Укажите ID соединения по умолчанию.

⑤ Укажите тип соединения.

⑥ Укажите имя хука.

⑦ Определите метод `.init()`.

⑧ Инициализируйте родительский хук.

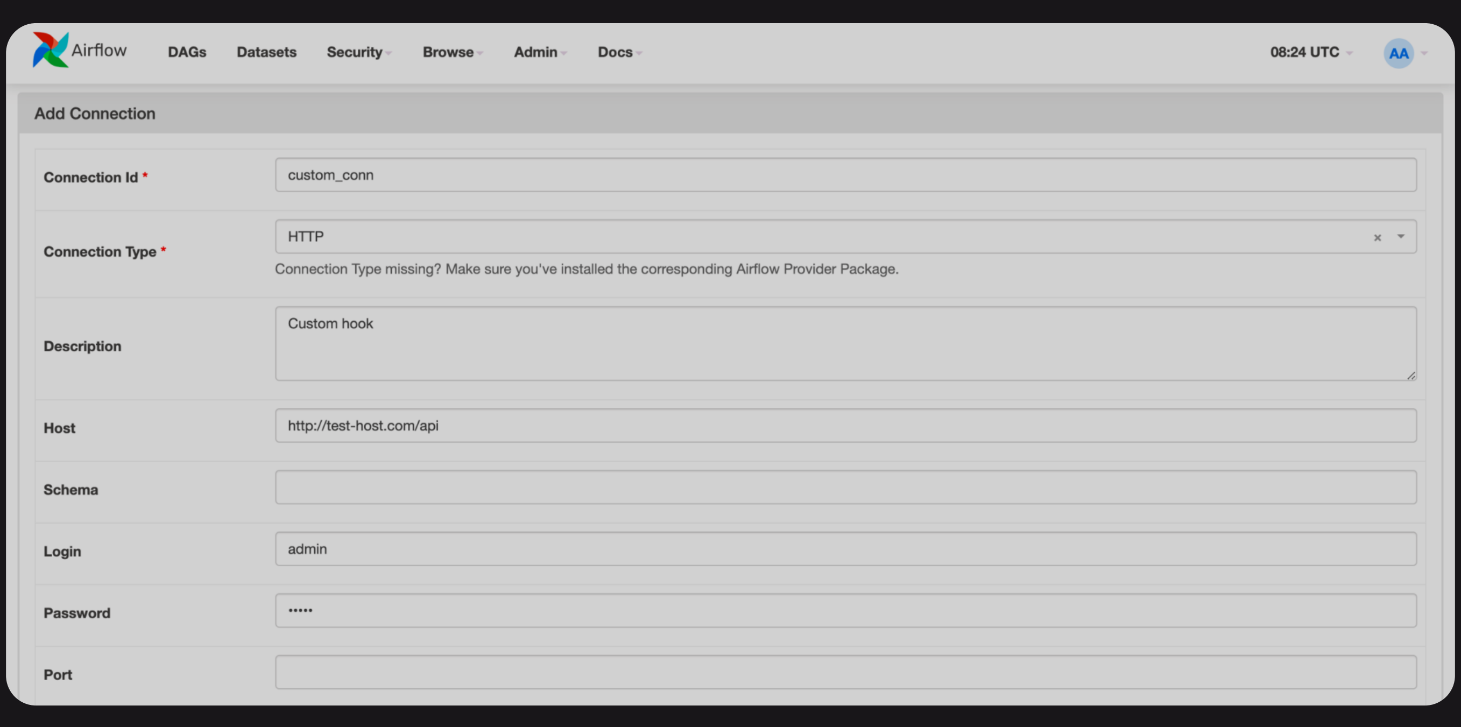
⑨ Назначьте переменные класса.

⑩ При необходимости вызовите метод `.get_conn()` при инициализации.

3. Вызовите новый хук внутри DAG:

```
from custom_hook import CustomHook
```

4. Чтобы использовать пользовательский хук, необходимо создать соединение Airflow с параметрами, указанными в коде хука.



Add Connection

Connection Id *	custom_conn
Connection Type *	HTTP Connection Type missing? Make sure you've installed the corresponding Airflow Provider Package.
Description	Custom hook
Host	http://test-host.com/api
Schema	
Login	admin
Password	•••••
Port	

Добавление нового соединения в веб-интерфейсе Airflow

Более подробно о веб-интерфейсе Airflow читайте в статье Обзор UI в Airflow.

Динамическая генерация DAG в Airflow

Елена Костюченко

Содержание

Назад к содержанию

Обзор
Динамическая генерация DAG
Динамический мэппинг задач

Обзор

Традиционные DAG удобно использовать для конвейера данных с одним источником, например, когда нужно извлечь данные из файла, обработать их и загрузить результаты.

В случаях, когда необходимо запустить один и тот же пайплайн с разными параметрами или для нескольких источников данных, лучшим решением будет генерировать DAG динамически для каждого случая.

Существует два способа генерации DAG с вариативными параметрами:

- **Динамическая генерация DAG (dynamic DAGs generation)**
  В динамически генерируемых DAG структура DAG меняется в зависимости от определенных статических значений (файлов конфигурации, переменных среды, имен файлов и т.д.).
- **Динамический мэппинг задач (dynamic task mapping)**
  Динамический мэппинг задач позволяет генерировать параллельные задачи во время выполнения на основе вывода предыдущей задачи.

Для удобства примеры в этой статье написаны с использованием API TaskFlow, но вы также можете динамически генерировать DAG, используя традиционный синтаксис.

Динамическая генерация DAG

Процесс динамического создания DAG похож на процесс создания обычного DAG. Поскольку Airflow выполняет весь Python-код, размещенный в директории *dags*, вы можете запустить выполнение любого Python-файла, который генерирует объекты DAG.

Частота генерации новых DAG определяется параметром `min_file_process_interval`, значение которого по умолчанию составляет 30 секунд. Вы можете изменить значение этого параметра, отредактировав поле ***cfg_properties_template*** через ADCM.

Чтобы отредактировать свойство `min_file_process_interval` через ADCM:

1. Перейдите в веб-интерфейс ADCM и выберите нужный кластер ADH.
2. Перейдите в ***Services*** → ***Airflow2*** → ***Primary configuration*** и включите опцию ***Show advanced***.
3. Откройте раздел ***airflow.cfg*** и выберите ***cfg_properties_template***.
4. Установите желаемое значение для `min_file_process_interval`.
5. Сохраните конфигурацию, выбрав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Restart***.

> **ВНИМАНИЕ**
>
> Если значение параметра `min_file_process_interval` слишком мало, DAG будут генерироваться слишком часто, что может создать большую нагрузку на планировщик или отдельный процесс `dag-processor`.

Чтобы создать динамически генерируемый DAG:

1. В директории *dags* домашнего каталога Airflow добавьте новый файл Python, который будет содержать код генератора DAG:

```
$ sudo vi /opt/airflow/dags/dag_generator.py
```

2. Добавьте код для генератора DAG:

```
from airflow import DAG
from airflow.decorators import task
from datetime import datetime   ①

for file in ("dev_data.csv", "test_data.csv", "prod_data.csv"):  ②
    dag_id = f"generated_dag_{file}"

    @dag(dag_id=dag_id, schedule="@daily",
    default_args=default_args, catchup=False,
    start_date=datetime(2025,1,1)) ③

    def create_dag(filename):

        @task ④
        def extract(filename):
            return filename

        @task
        def process(filename):
            return filename

        @task
        def load(filename):
            print(filename)
            return filename

        load(process(extract(filename))) ⑤

    create_dag(filename)  ⑥
```

① Импорт зависимостей.

② Создание цикла для генерации DAG.

③ Создание функции DAG.

④ Добавление задач.

⑤ Определение зависимостей задач через вызов функций.

⑥ Вызов функции DAG.

При необходимости конфигурации DAG с помощью переменных рекомендуется использовать переменные окружения, а не переменные Airflow.

Использование переменных Airflow в коде верхнего уровня создает соединение с базой данных, где хранятся метаданные, что может замедлить синтаксический анализ и создать дополнительную нагрузку на базу данных.

Вы также можете импортировать переменные из одного из нескольких внешних источников, таких как файлы конфигурации.

Например, в директории *include/dag-configs/* домашнего каталога Airflow размещено два файла конфигурации JSON. Файлы имеют следующее содержимое:

- *dev_config.json*:

```
{
    "dag_id": "development_dag",
    "schedule": "@daily",
    "input": "dev_data.csv"
}
```

- *prod_config.json*:

```
{
    "dag_id": "production_dag",
    "schedule": "@daily",
    "input": "prod_data.csv"
}
```

Вы можете создать скрипт Python, который будет генерировать отдельные файлы DAG для каждого файла конфигурации, используя указанный файл DAG в качестве шаблона.

Чтобы генерировать DAG с использованием шаблона:

1. В домашней директории Airflow добавьте новый файл Python, который будет содержать шаблон DAG:

```
$ sudo vi /opt/airflow/include/templates/dag_template.py
```

2. Добавьте код, который будет использован генератором в качестве шаблона для DAG. Например, можно использовать код из предыдущего примера и определить параметры, которые будут меняться при генерации:

```
from airflow import DAG
from airflow.decorators import task
from datetime import datetime

@dag(dag_id = "dag_id_placeholder", schedule="schedule_placeholder",
catchup=False, start_date=datetime(2025,1,1)) ①

def create_dag(filename):

    @task
    def extract(filename):
        return filename

    @task
    def process(filename):
        return filename

    @task
    def load(filename):
        print(filename)
        return filename

    load(process(extract("input_placeholder"))) ②

create_dag(filename)
```

① Добавление переменных, куда будут подставляться значения параметров `dag_id` и `schedule_interval`.

② Добавление переменной для входного значения.

3. В домашней директории Airflow добавьте файл Python, который будет генерировать DAG с использованием шаблона:

```
$ sudo vi /opt/airflow/include/scripts/dag_generator.py
```

4. Напишите код генератора DAG:

```
import json
import os
import shutil
import fileinput

config_filepath = "include/dag-configs/" ①
template_filepath = "include/templates/dag_template.py" ②

for filename in os.listdir(config_filepath): ③
    if filename.endswith('.json'):
        config = json.load(open(f"include/data/{filename}"))
        generated_dag = f"dags/generated_dag_{config['dag_id']}.py"
        shutil.copyfile(template_filepath, generated_dag)
        for line in fileinput.input(generated_dag, inplace=True):
            line = line.replace("dag_id_placeholder", config['dag_id'])
            line = line.replace("schedule_placeholder", config['schedule'])
            line = line.replace("input_placeholder", config['input'])
            print(line, end="")
```

① Путь к файлам конфигурации.

② Путь к файлу шаблона.

③ Описание логики замены переменных на нужные значения из файлов конфигурации.

При выполнении скрипта директория *dag-configs* проверяется на наличие доступных файлов конфигурации JSON и генерируется файл DAG для каждого файла конфигурации на основе указанного шаблона. Переменные в шаблоне сопоставляются с полями из файлов конфигурации и заменяются на нужные значения.

После генерации файлов Python Airflow загрузит их как обычные DAG.

Динамический мэппинг задач

В примерах выше демонстрируется, как генерировать DAG с разными параметрами. При этом количество задач в сгенерированных DAG остается прежним и не может меняться.

Чтобы создавать задачи в DAG на основе меняющихся выходных данных другой задачи, можно использовать динамический мэппинг задач.

Этот подход использует фреймворк MapReduce и работает с помощью метода `expand()`. Этот метод создает новые задачи путем копирования той, которая его вызывает, с использованием предоставленных аргументов и вывода предыдущей задачи.

Метод `expand()` используется с оператором и принимает список или словарь значений в качестве входных данных. Эти значения также могут быть выходными данными другой задачи в формате `XComArg`.

Пример динамического мэппинга задач:

```
from airflow.decorators import dag, task
from datetime import datetime

@dag(dag_id = "dynamic_task_dag", schedule="@daily", catchup=False, start_date=datetime(2025,1,1))
def create_tasks():

    @task
    def download_files(file: str): ①
        print(file)

    files = download_files.expand(file=["file_a", "file_b", "file_c"]) ②

    download_files(file)

create_tasks()
```

① Создание функции задачи. В данном случае — функции, которая выводит содержимое файла.

② Вызов метода `expand()` и передача ему списка файлов в качестве аргумента. DAG сгенерирует три задачи для каждого файла.

Пример динамического мэппинга задач, когда вывод предыдущей задачи не определен:

```
from airflow.decorators import dag, task
from datetime import datetime
import random

@dag(dag_id = "dynamic_task_dag", schedule="@daily",
catchup=False, start_date=datetime(2025,1,1))

def create_tasks():

    @task
    def get_files():
        return ["file_{i}" for i in range(random.randint(3,5))] ①

    @task
    def download_files(file: str):
        print(file)

    files = download_files.expand(file=get_files()) ②

create_tasks()
```

① Создание функции со случайным выводом.

② Вызов метода `expand()` и передача ему вывода функции в качестве аргумента. DAG сгенерирует задачу для каждого файла.

Логирование в Airflow

Альберт Багдасарян, Елена Костюченко

Содержание

Назад к содержанию

Поиск по логам
Уровни логирования
Конфигурация логирования

Airflow ведет текстовые логи для анализа ошибок, которые могут возникнуть в процессе работы DAG. Эти логи расположены в директории */var/log/airflow/* хоста сервера Airflow, но они также доступны в пользовательском интерфейсе Airflow.

Пути к логам

Полный путь к лог-файлам выглядит следующим образом:

/var/log/airflow/dag_id=<DAG_ID>/run_id=<DAG_Run_ID>/task_id=<Task_ID>/<log_number>.log

Здесь:

- `<DAG_ID>` — идентификатор DAG.
- `<DAG_Run_ID>` — идентификатор запуска DAG, который объединяет в себе тип запуска и временную метку. Например, *run_id=scheduled__2024-07-14T14:18:33.254657+00:00*.
- `<Task_ID>` — идентификатор задачи.
- `<log_number>` — номер лог-файла (отсчет начинается с 1).

Полный путь к логам менеджера процессов (process manager) — */var/log/airflow/dag_processor_manager/*.

Полный путь к логам планировщика (scheduler) — */var/log/airflow/scheduler/*.

Для просмотра логов Airflow на хосте:

1. Подключитесь к серверу Airflow с помощью SSH и запустите следующую команду:

```
$ ls -la /var/log/airflow/
```

Вывод в консоль приведен ниже:

```
total 12
drwxr-xr-x.  5 airflow airflow   77 Jul 15 14:18 .
drwxr-xr-x. 16 root    root    4096 Aug  5 08:44 ..
drwxr-xr-x. 13 airflow airflow 4096 Jul 30 10:12 dag_id=adcm_check
drwxr-xr-x.  2 airflow airflow  109 Jul 29 23:54 dag_processor_manager
drwxr-xr-x. 20 airflow airflow 4096 Aug  5 07:56 scheduler
```

2. Откройте содержимое нужного лог-файла:

```
$ cat /var/log/airflow/dag_id=adcm_check/run_id=manual__2024-07-15T14:18:43.743847+00:00/task_id=runme_1/attempt=1.log
```

Вывод в консоль приведен ниже:

```
[2024-07-15T14:18:46.143+0000] {taskinstance.py:1103} INFO - Dependencies all met for dep_context=requeueable deps ti=
<TaskInstance: adcm_check.runme_1 manual__2024-07-15T14:18:43.743847+00:00 [queued]>
[2024-07-15T14:18:46.144+0000] {taskinstance.py:1308} INFO - Starting attempt 1 of 4
[2024-07-15T14:18:46.157+0000] {taskinstance.py:1327} INFO - Executing <Task(BashOperator): runme_1> on 2024-07-15 14:18:43.
743847+00:00
[2024-07-15T14:18:46.164+0000] {standard_task_runner.py:57} INFO - Started process 7082 to run task
[2024-07-15T14:18:46.168+0000] {standard_task_runner.py:84} INFO - Running: ['airflow', 'tasks', 'run', 'adcm_check', 'runme_1',
'manual__2024-07-15T14:18:43.743847+00:00', '--job-id', '12', '--raw', '--subdir', 'DAGS_FOLDER/adcm_check.py', '--cfg-path',
'/tmp/tmpr_3qshf4']
[2024-07-15T14:18:46.171+0000] {standard_task_runner.py:85} INFO - Job 12: Subtask runme_1
[2024-07-15T14:18:46.233+0000] {task_command.py:410} INFO - Running <TaskInstance: adcm_check.runme_1 manual__2024-07-15T14:18:43.
743847+00:00 [running]> on host elenas-adh3.ru-central1.internal
[2024-07-15T14:18:46.326+0000] {taskinstance.py:1545} INFO - Exporting env vars: AIRFLOW_CTX_DAG_OWNER='airflow'
AIRFLOW_CTX_DAG_ID='adcm_check' AIRFLOW_CTX_TASK_ID='runme_1' AIRFLOW_CTX_EXECUTION_DATE='2024-07-15T14:18:43.743847+00:00'
AIRFLOW_CTX_TRY_NUMBER='1' AIRFLOW_CTX_DAG_RUN_ID='manual__2024-07-15T14:18:43.743847+00:00'
```

Поиск по логам

Чтобы найти в логах интересующую вас информацию, например, сообщения об ошибках, подключитесь к хосту с логами, которые вы хотите просмотреть, и воспользуйтесь командой `grep`.

Например:

```
$ cat /var/log/airflow/dag_processor_manager/dag_processor_manager.log | grep -i -A3 -B1 error | grep -i -A3 -B1 error
```

Эта команда парсит лог менеджера процессов и ищет сообщения, содержащие слово `error`. Опция `-i` позволяет игнорировать различия в регистре, а опции `-A3 -B1` добавляют к найденной строке с ошибкой одну строчку до нее и три строчки после.

Пример вывода команды:

```
File Path                                                                                                                  PID
Runtime      # DAGs    # Errors  Last Runtime    Last Run
-------------------------------------------------------------------------------------------------------------------------  -----
---------  --------  ----------  --------------  -------------------
/opt/airflow/lib/python3.10/site-packages/airflow/example_dags/example_setup_teardown.py
0           0  0.03s           2024-08-05T10:38:25
/opt/airflow/lib/python3.10/site-packages/airflow/example_dags/example_setup_teardown_taskflow.py
0           0  0.03s           2024-08-05T10:38:25
--
...
File Path                                                                                                                  PID
Runtime      # DAGs    # Errors  Last Runtime    Last Run
-------------------------------------------------------------------------------------------------------------------------  -----
---------  --------  ----------  --------------  -------------------
/opt/airflow/lib/python3.10/site-packages/airflow/example_dags/example_setup_teardown.py
0           0  0.03s           2024-08-05T10:38:55
/opt/airflow/lib/python3.10/site-packages/airflow/example_dags/example_setup_teardown_taskflow.py
0           0  0.03s           2024-08-05T10:38:55
--
```

Уровни логирования

Airflow использует стандартный фреймворк ведения логов в Python и поддерживает следующие уровни логирования (от наименее к наиболее информативным):

1. `CRITICAL` — уведомляет о серьезной ошибке, которая может остановить работу сервиса.
2. `FATAL` — сообщает, что операция не может быть выполнена и будет завершена.
3. `ERROR` — уведомляет, что программа работает неправильно или прекратила работу.
4. `WARN` — предупреждает о потенциальных проблемах. Это означает, что программа работает не по стандартному сценарию и в будущем могут возникнуть проблемы.
5. `INFO` — передает информацию о жизненном цикле или состоянии программы.
6. `DEBUG` — выводит отладочную информацию о внутренних состояниях программы.

Включение одного уровня логирования включит этот уровень и все уровни выше него. Например, если установлен уровень логирования `ERROR`, то в логи попадут только ошибки, фатальные и критические сообщения, но не `INFO`, `DEBUG` и `WARN`.

Конфигурация логирования

Чтобы изменить параметры логирования через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на ***Airflow2***.
3. Выберите параметр и внесите необходимые изменения.
4. Подтвердите изменения в конфигурации, нажав ***Save***.
5. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Параметры логирования Airflow:

- ***Logging level*** — уровень логирования для сервиса Airflow.
- ***Logging level for Flask-appbuilder UI*** — уровень логирования для Flask-appbuilder.
- ***cfg_properties_template*** — файл конфигурации Airflow, содержащий также настройки логирования для задач.

Настройки логирования в файле конфигурации cfg_properties_template

```
[logging]
base_log_folder = /var/log/airflow
remote_logging = False
remote_log_conn_id =
google_key_path =

remote_base_log_folder =
encrypt_s3_logs = False
{% endraw %}
logging_level = {{ services.airflow2.config.airflow_cfg.logging_level }}
{% raw -%}
celery_logging_level =
{% endraw %}
fab_logging_level = {{ services.airflow2.config.airflow_cfg.fab_logging_level }}
{% raw -%}
logging_config_class =
colored_console_log = True
colored_log_format = [%%(blue)s%%(asctime)s%%(reset)s] {%%(blue)s%%(filename)s:%%(reset)s%%(lineno)d} %%(log_color)s%%
(levelname)s%%(reset)s - %%(log_color)s%%(message)s%%(reset)s
colored_formatter_class = airflow.utils.log.colored_log.CustomTTYColoredFormatter
log_format = [%%(asctime)s] {%%(filename)s:%%(lineno)d} %%(levelname)s - %%(message)s
simple_log_format = %%(asctime)s %%(levelname)s - %%(message)s
task_log_prefix_template =
log_filename_template = dag_id={{ ti.dag_id }}/run_id={{ ti.run_id }}/task_id={{ ti.task_id }}/{%% if ti.map_index >= 0
%%}map_index={{ ti.map_index }}/{%% endif %%}attempt={{ try_number }}.log
log_processor_filename_template = {{ filename }}.log
dag_processor_manager_log_location = /var/log/airflow/dag_processor_manager/dag_processor_manager.log
task_log_reader = task
extra_logger_names =
```

Управление сервисом Airflow через ADCM

Елена Костюченко

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Airflow. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Airflow, перечислены в таблице ниже.

Подробное описание компонентов Airflow приведено в статье Airflow.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Airflow
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются

Действия с компонентами Airflow

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Получение клиентских конфигураций

Константин Алпашкин

Содержание

Назад к содержанию

- Обзор работы
  - Экспортируемые файлы конфигурации
- Пример экспорта конфигурации

Начиная с версии ADH 4.0.0 вы можете экспортировать клиентские конфигурации сервисов, установленных в кластере ADH. Экспортированные конфигурации можно использовать, например, при настройке клиентского ПО, которое взаимодействует с ADH-кластером или для настройки сервисов другого кластера ADH.

Обзор работы

Функционал экспорта конфигураций встроен в сервис ***Core configuration***. В частности, у сервиса есть действие ***Export client configurations***, которое собирает конфигурации установленных в кластере сервисов, архивирует их и предоставляет для загрузки через Nginx-эндпойнт.

ПРИМЕЧАНИЕ

Не допускается установка компонентов Configuration server и Hive Tez на один хост.

Конфигурации экспортируются в виде отдельных архивов *.tar.gz* для каждого сервиса. Список экспортируемых сервисов и конфигурационных файлов представлен ниже.

Экспортируемые файлы конфигурации

При выполнении действия ***Export client configurations*** экспортируются конфигурации для следующих сервисов.

Сервис	Конфигурационный файл(ы)
Core configuration	core-site.xml ssl-client.xml log4j.properties.xml
HDFS	hdfs-site.xml log4j.properties.xml
YARN	yarn-env.sh yarn-site.xml mapred-site.xml
Hive	hive-site.xml hive-env.sh beeline-hs2-connection.xml beeline-log4j2.properties hive-log4j2.properties hive-log4j.properties hive-metastore-env.sh hive-server2-env.sh tez-site.xml
Solr	solr.xml solr-env.sh log4j2-console.xml log4j2.xml
HBase	hbase-site.xml hbase-env.sh log4j.properties
Flink	flink-conf.yaml flink-env.sh log4j-cli.properties log4j.properties
Impala	impalarc impala-shell-env.sh hive-site.xml
Spark	spark-defaults.conf spark-env.sh hive-site.xml log4j2.properties
Ozone	ozone-site.xml

ПРИМЕЧАНИЕ

Конфиденциальные данные (пароли, секретные ключи, JCEK-хранилища) не включаются в экспортируемые файлы конфигурации.

Пример экспорта конфигурации

Следующие шаги показывают, как экспортировать конфигурацию кластера ADH:

1. В ADCM UI перейдите на страницу ***Clusters*** → ***<cluster_name>*** → ***Services*** → ***Core configuration***.
2. Выполните сервисное действие ***Export client configurations***.
3. Откройте страницу ***Core configuration*** → ***Info*** и перейдите по ссылке ***Configuration server web interface*** (*http://<configuration_server_host>:9998*). На странице отобразится список архивов *tar.gz* с конфигурациями для каждого сервиса. Например:

```
../
all_configs.tar.gz   13-Aug-2025 16:06    7948
core.tar.gz          13-Aug-2025 16:06    4632
hdfs.tar.gz          13-Aug-2025 16:06    1494
yarn.tar.gz          13-Aug-2025 16:06    2486
```

Для загрузки архива кликните по ссылке.

Управление сервисом Core configuration через ADCM

Владимир Адаменков

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Core configuration. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Core configuration, перечислены в таблице ниже.

Действия с сервисом Core configuration

Действие	Описание
Add/Remove components	Добавляет/удаляет компонент(ы) Core configuration. При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете распределить компоненты Core configuration между хостами
Check	Проверяет доступность порта компонента Configuration server
Export client configurations	Сохраняет в архив все клиентские конфигурации (для всех сервисов и для кластера) без применения их на хостах
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Reinstall Core	Переустанавливает сервис Core configuration
Update Core configuration	Обновляет файлы *core-site.xml*, *hadoop-env.sh*, *ssl-server.xml* и *ssl-client.xml*
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис

Действия, доступные для компонентов сервиса Core configuration, перечислены в таблице ниже.

Действия с компонентами Core configuration

Действие	Описание
Check	Выполняет проверку работоспособности компонента
Restart	Перезапускает компонент. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт компонента выполняется с учетом всех параметров, указанных в настройках этого компонента в ADCM. В противном случае настройки компонента в ADCM игнорируются
Start	Запускает компонент. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск компонента выполняется с учетом всех параметров, указанных в настройках этого компонента в ADCM. В противном случае настройки компонента в ADCM игнорируются
Stop	Останавливает компонент
Export client configurations	Сохраняет в архив все клиентские конфигурации (для всех сервисов и для кластера) без применения их на хостах

Обзор Flink UI

Константин Алпашкин

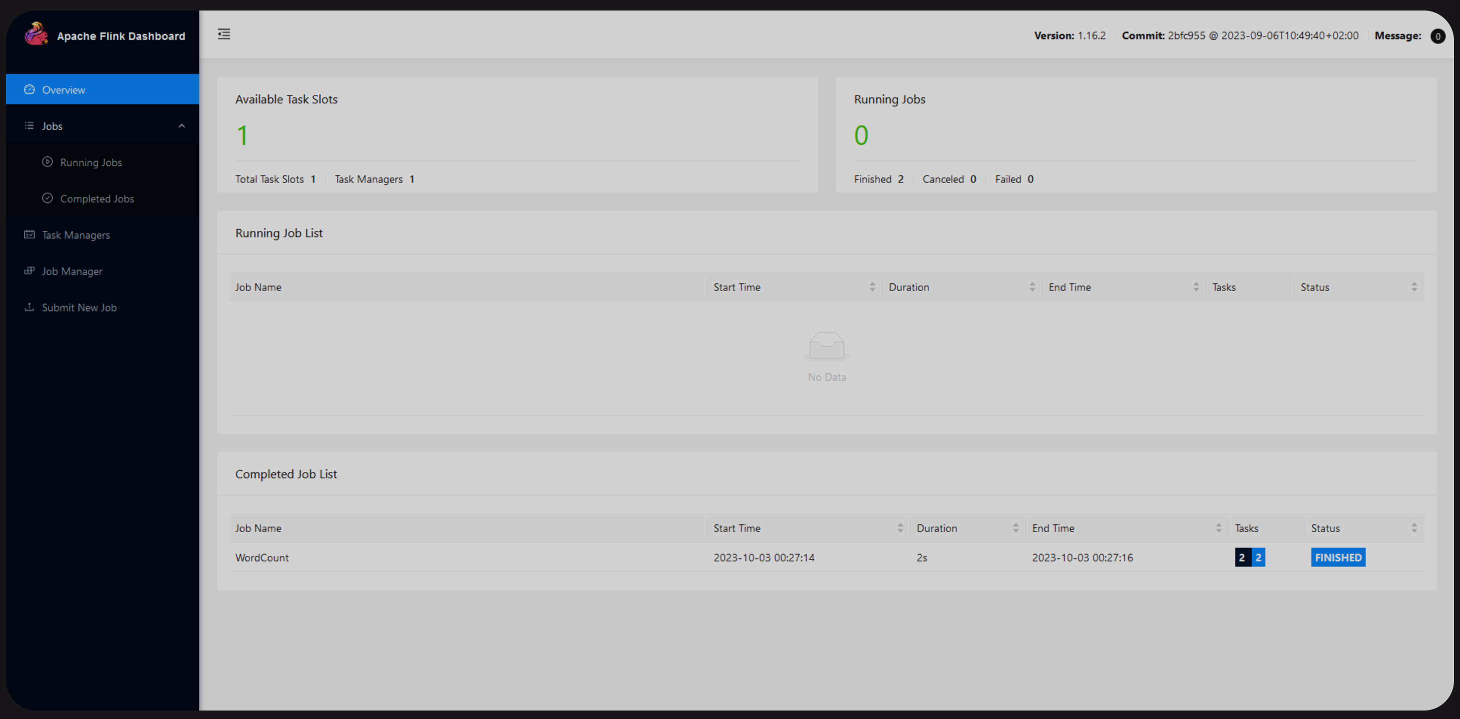
Содержание

Назад к содержанию

Веб-интерфейс Flink History Server

Flink Dashboard — это веб-интерфейс для мониторинга работы приложений Flink. На данном дашборде вы можете видеть активные, выполненные и неудачные задачи. Чтобы открыть дашборд, используйте страницу ***Flink Service*** в ADCM.

Данный веб-интерфейс служит универсальным способом для мониторинга всех задач, выполняемых в YARN-кластере. В интерфейсе вы можете видеть все активные, выполненные и неудачные задачи. URL для доступа к интерфейсу выглядит следующим образом: *http://FQDN:8081*, а сам интерфейс показан ниже.


Apache Flink Dashboard
Overview
Jobs
Running Jobs
Completed Jobs
Task Managers
Job Manager
Submit New Job

Version: 1.16.2 **Commit:** 2bfc955 @ 2023-09-06T10:49:40+02:00 **Message:** 0

Available Task Slots
1
Total Task Slots 1 Task Managers 1

Running Jobs
0
Finished 2 Canceled 0 Failed 0

Running Job List

Job Name	Start Time	Duration	End Time	Tasks	Status

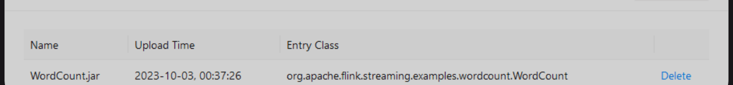
No Data

Completed Job List

Job Name	Start Time	Duration	End Time	Tasks	Status
WordCount	2023-10-03 00:27:14	2s	2023-10-03 00:27:16	2 2	FINISHED

Flink Dashboard

Вы можете загрузить jar-файл на странице ***Submit new Job***.

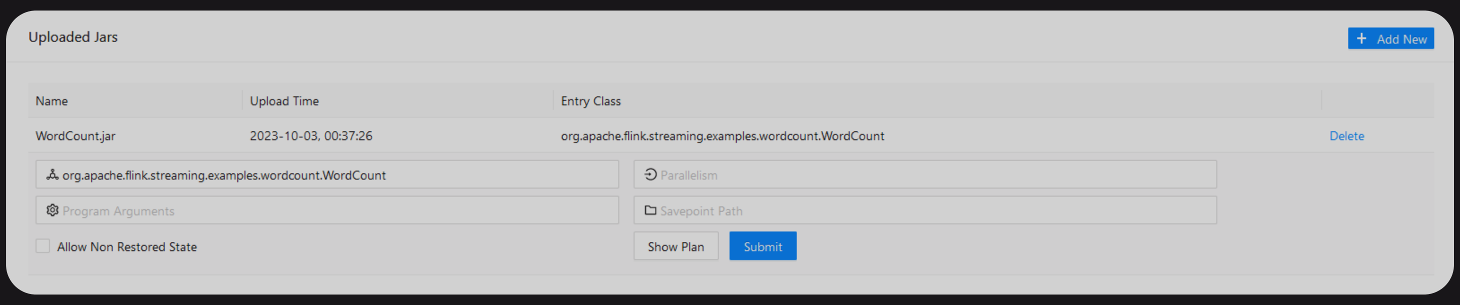


Uploaded Jars

Name	Upload Time	Entry Class	
WordCount.jar	2023-10-03, 00:37:26	org.apache.flink.streaming.examples.wordcount.WordCount	Delete

Submit new Job

Загрузив jar, вы можете запустить его с параметрами.


+ Add New
Show Plan
Submit

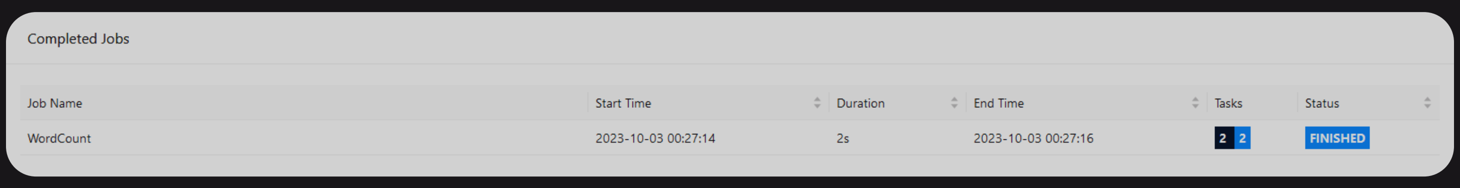
Uploaded Jars

Name	Upload Time	Entry Class	
WordCount.jar	2023-10-03, 00:37:26	org.apache.flink.streaming.examples.wordcount.WordCount	Delete

org.apache.flink.streaming.examples.wordcount.WordCount
Parallelism
Program Arguments
Savepoint Path
☐ Allow Non Restored State

Uploaded Jars

Вы можете указать параметры и запустить задачу. Когда задача будет выполнена, детали выполнения станут доступны на странице ***Completed jobs***.

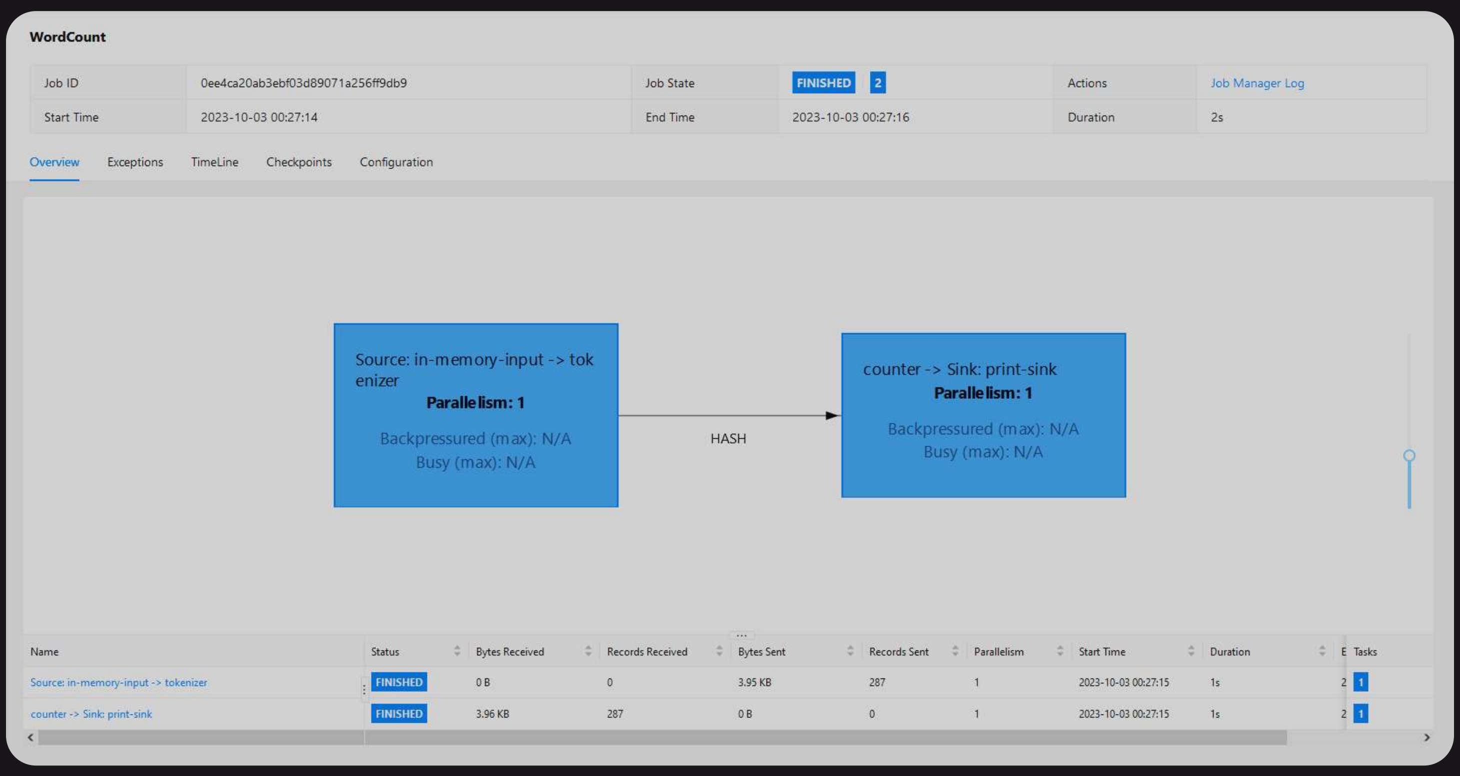


Completed Jobs

Job Name	Start Time	Duration	End Time	Tasks	Status
WordCount	2023-10-03 00:27:14	2s	2023-10-03 00:27:16	2 2	FINISHED

Completed Jobs

Кликните на ***Completed Jobs***, чтобы увидеть детальный обзор по задачам.


Overview
Exceptions
TimeLine
Checkpoints
Configuration
Source: in-memory-input -> tokenizer
Parallelism: 1
Backpressured (max): N/A
Busy (max): N/A
HASH
counter -> Sink: print-sink
Parallelism: 1
Backpressured (max): N/A
Busy (max): N/A

WordCount

Job ID	0ee4ca20ab3ebf03d89071a256ff9db9	Job State	FINISHED 2	Actions	Job Manager Log
Start Time	2023-10-03 00:27:14	End Time	2023-10-03 00:27:16	Duration	2s

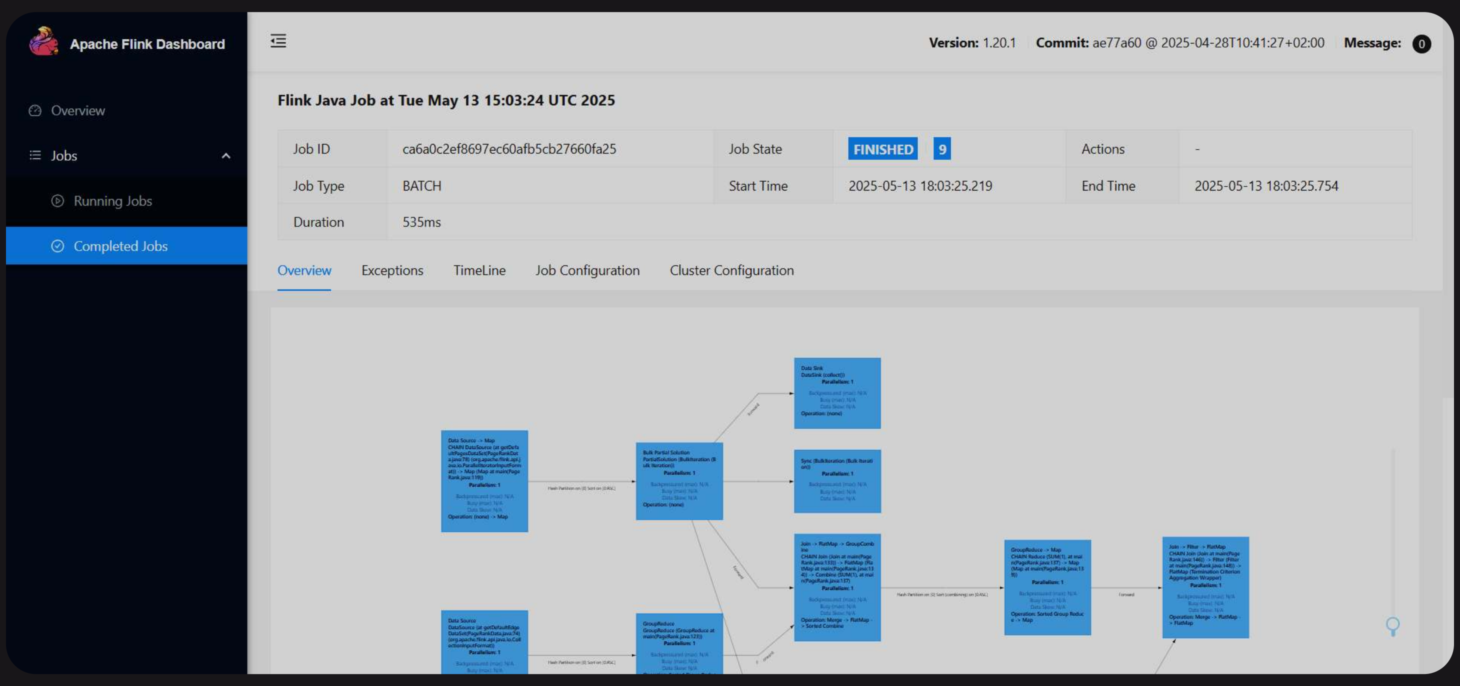
Name	Status	Bytes Received	Records Received	Bytes Sent	Records Sent	Parallelism	Start Time	Duration	E	Tasks
Source: in-memory-input -> tokenizer	FINISHED	0 B	0	3.95 KB	287	1	2023-10-03 00:27:15	1s	2	1
counter -> Sink: print-sink	FINISHED	3.96 KB	287	0 B	0	1	2023-10-03 00:27:15	1s	2	1

Jobs Overview

Веб-интерфейс Flink History Server

Компонент Flink History Server предоставляет дополнительный веб-интерфейс, в котором представлена статистика о выполненных задачах Flink. Актуальный URL веб-интерфейса Flink History Server можно найти в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Flink*** → ***Info***).

Веб-интерфейс Flink History Server показан ниже.


Apache Flink Dashboard
Overview
Jobs
Running Jobs
Completed Jobs
Overview
Exceptions
TimeLine
Job Configuration
Cluster Configuration

Version: 1.20.1 **Commit:** ae77a60 @ 2025-04-28T10:41:27+02:00 **Message:** 0

Flink Java Job at Tue May 13 15:03:24 UTC 2025

Job ID	ca6a0c2ef8697ec60afb5cb27660fa25	Job State	FINISHED 9	Actions	-
Job Type	BATCH	Start Time	2025-05-13 18:03:25.219	End Time	2025-05-13 18:03:25.754
Duration	535ms				

Веб-интерфейс Flink History Server

Подключение к Flink через CLI

Константин Алпашкин

Назад к содержанию

Flink имеет CLI-интерфейс для запуска и контроля программ, упакованных в виде Java-архивов (jar). Для создания новой задачи (**job**), необходимо загрузить jar-файл со всеми зависимостями в активный кластер Flink и запустить jar. Примеры задач доступны в директории */usr/lib/flink/examples/*.

Чтобы развернуть демонстрационное приложения WordCount в запущенном кластере, выполните следующую команду:

```
$ flink run /usr/lib/flink/examples/streaming/WordCount.jar
```

Вы можете запустить другие примеры из директории аналогичным способом. Большинство примеров не требуют аргументов и используют данные, зашитые в приложение. Чтобы запустить *WordCount.jar* с реальными данными, укажите путь к входным данным:

```
$ ./bin/flink run ./examples/batch/WordCount.jar --input /path/to/some/text/data --output /path/to/result
```

ПРИМЕЧАНИЕ

Для работы с удаленной файловой системой необходимо указать префикс, например `hdfs://`.

Результат выполнения задачи можно увидеть в лог-файлах:

```
$ tail log/flink-*-taskexecutor-*.out
```

Пример вывода:

```
(nymph,1)
(in,3)
(thy,1)
(orisons,1)
(be,4)
(all,2)
(my,1)
(sins,1)
(remember,1)
(d,4)
```

Примеры использования PyFlink

Константин Алпашкин

Содержание

Назад к содержанию

- PyFlink API: DataStream vs Table
- Table API
  - Создание source-таблицы
  - Запросы и операции с таблицами
  - Запись данных в sink-таблицу
- DataStream API
  - Создание среды выполнения
  - Получение данных из источника
  - Операторы преобразования
  - Запись данных в приемник
- Управление зависимостями
  - Зависимости JAR
  - Зависимости Python

PyFlink — это Python API для Flink, который позволяет создавать масштабируемые приложения для пакетной и потоковой обработки данных. "Под капотом" PyFlink использует библиотеку Py4J, которая подключается к Flink JVM и взаимодействует с объектами Flink.

PyFlink доступен в виде модуля `pip`. ADH поставляется с интерпретатором Python, в котором уже установлен модуль PyFlink, а также все необходимые зависимости. Данный интерпретатор расположен в */opt/pyflink-python/* на хостах ADH с установленными компонентами Flink. Используя данный интерпретатор, вы можете запускать приложения PyFlink как обычные Python-скрипты, например:

```
$ source /opt/pyflink-python/bin/activate
$ /opt/pyflink-python/bin/python3 create_table.py
```

PyFlink API: DataStream vs Table

PyFlink предоставляет два различных API в зависимости от необходимого уровня абстракции:

- Table API. Позволяет выполнять реляционные запросы к данным, аналогично использованию SQL или работе с табличными данными в Python.
- DataStream API. Предоставляет низкоуровневый доступ к функционалу Flink. Позволяет напрямую взаимодействовать с данными состояния (state data), контрольными точками, системой восстановления и так далее. Позволяет создавать более сложные задачи Flink.

В этой статье приведены примеры работы с обоими API. Дополнительные примеры использования PyFlink доступны в GitHub. Актуальная справочная информация по PyFlink API доступна в документации Flink.

Table API

PyFlink Table API — это универсальный реляционный API для пакетной и потоковой обработки. Позволяет использовать SQL-подобный язык для запросов и операций с таблицами. Запросы выполняются одинаково для ограниченных и неограниченных потоков данных.

Первым шагом в приложении является создание объекта среды выполнения (TableEnvironment). Данный объект — основа любого приложения Table API; используется для настройки и выполнения операций с таблицами. Пример использования объекта среды показан ниже:

```
from pyflink.table import TableEnvironment, EnvironmentSettings

env_settings = EnvironmentSettings.in_streaming_mode() ①
t_env = TableEnvironment.create(env_settings) ②
t_env.get_config().set("parallelism.default", "1") ③
```

① Установка режима работы (потоковый или пакетный).

② Создание объекта `TableEnvironment`.

③ Установка параметров конфигурации (параллелизм, восстановление, данные состояния и так далее) на уровне приложения.

Создание source-таблицы

Используя `TableEnvironment`, можно создавать таблицы Flink. Таблица Flink (объект `Table`) описывает пайплайн преобразования данных. Сама таблица не хранит никаких данных, а лишь указывает, откуда данные необходимо получить (source-таблица) или куда данные необходимо сохранить после обработки (sink-таблица).

Далее показаны несколько способов создания source-таблиц.

Из коллекции

Следующий код создает таблицу из списка Python.

```
from pyflink.table import EnvironmentSettings, TableEnvironment
from pyflink.table.expressions import col

env_settings = EnvironmentSettings.in_streaming_mode()
t_env = TableEnvironment.create(env_settings)
t_env.get_config().set("parallelism.default", "1")

mock_data = [
    (1, 1001, 100.00, "2025-01-01"),
    (2, 1002, 20.00, "2025-01-02"),
    (3, 1003, 70.00, "2025-01-03"),
]
table_1 = t_env.from_elements(elements=mock_data, ①
                              schema=['txn_id', 'acc_id', 'txn_value', 'txn_date'])
print("Inferred data types for table_1:")
table_1.print_schema()

from pyflink.table import DataTypes
table_2 = t_env.from_elements(elements=mock_data, ②
                              schema=DataTypes.ROW([DataTypes.FIELD("txn_id", DataTypes.TINYINT()),
                                                    DataTypes.FIELD("acc_id", DataTypes.INT()),
                                                    DataTypes.FIELD("txn_value", DataTypes.DOUBLE()),
                                                    DataTypes.FIELD("txn_date", DataTypes.STRING())]))
print(f"The data type of 'txn_date': {table_2.get_schema().get_field_data_type('txn_date')}")
table_2.print_schema()
table_2 = table_2.select(
    col("txn_id"),
    col("acc_id"),
    col("txn_value"),
    col("txn_date").cast(DataTypes.DATE()).alias("txn_date")
)
print(f"Data type of 'txn_date' after casting:"
      f"{table_2.get_schema().get_field_data_type('txn_date')}")
print("SELECT * FROM table_2:")
table_2.execute().print() ③
```

① Создание таблицы, схема которой формируется автоматически на основе данных коллекции.

② Создание таблицы с явным указанием типов данных в схеме.

③ Вывод содержимого таблицы.

Пример вывода

```
Inferred data types for table_1:
(
  `txn_id` BIGINT,
  `acc_id` BIGINT,
  `txn_value` DOUBLE,
  `txn_date` STRING
)
The data type of 'txn_date': VARCHAR
(
  `txn_id` TINYINT,
  `acc_id` INT,
  `txn_value` DOUBLE,
  `txn_date` STRING
)
Data type of 'txn_date' after casting:DATE
SELECT * FROM table_2:
+----+--------+-------------+--------------------------------+------------+
| op | txn_id |      acc_id |                      txn_value |   txn_date |
+----+--------+-------------+--------------------------------+------------+
| +I |      1 |        1001 |                          100.0 | 2025-01-01 |
| +I |      2 |        1002 |                           20.0 | 2025-01-02 |
| +I |      3 |        1003 |                           70.0 | 2025-01-03 |
+----+--------+-------------+--------------------------------+------------+
3 rows in set
```

Из топика Kafka

Следующий код создает таблицу, подписанную на Kafka-топик через TableDescriptor.

```
from pyflink.table import EnvironmentSettings, TableEnvironment, Schema, DataTypes, TableDescriptor

env_settings = EnvironmentSettings.in_streaming_mode()
t_env = TableEnvironment.create(env_settings)
t_env.get_config().set("parallelism.default", "1")
t_env.get_config().set("pipeline.jars", "file:///home/konstantin/pyflink_demo/flink-connector-kafka-3.3.0-1.19.jar") ①

t_env.create_temporary_table( ②
    "kafka_source_table",
    TableDescriptor.for_connector("kafka")
    .schema(Schema.new_builder()
            .column("txn_id", DataTypes.INT())
            .column("acc_id", DataTypes.INT())
            .column("txn_value", DataTypes.DOUBLE())
            .column("txn_date", DataTypes.DATE())
            .build())
    .option("properties.bootstrap.servers", "<host>:<port>")
    .option("topic", "transactions_test")
    .option("properties.group.id", "txn_group")
    .option("scan.startup.mode", "earliest-offset")
    .option("format", "json")
    .option("json.fail-on-missing-field", "false")
    .option("json.ignore-parse-errors", "true")
    .build()
)
```

① Добавление JAR Kafka-коннектора в контекст задачи PyFlink. Информация об актуальной версии коннектора доступна в документации Flink.

② Создание временной таблицы, в которую попадают данные из Kafka-топика.

Из файла, используя DDL

Следующий код создает таблицу на основе CSV-файла, используя SQL.

```
from pyflink.table import EnvironmentSettings, TableEnvironment

env_settings = EnvironmentSettings.in_streaming_mode()
t_env = TableEnvironment.create(env_settings)

t_env.execute_sql("""
CREATE TABLE src_table_from_csv (
    txn_id INT,
    acc_id INT,
    txn_value INT,
    txn_date STRING
) WITH (
    'connector' = 'filesystem', ①
    'format' = 'csv',
    'path' = 'test.csv',
    'csv.field-delimiter' = ','
)
""")

table = t_env.from_path("src_table_from_csv")
table.execute().print()
```

① Использование встроенного коннектора для работы с файлами.

Пример вывода

```
+----+-------------+-------------+-------------+--------------------------------+
| op |      txn_id |      acc_id |   txn_value |                       txn_date |
+----+-------------+-------------+-------------+--------------------------------+
| +I |           1 |        1001 |          20 |                     2025-01-02 |
| +I |           2 |        1002 |         110 |                     2025-01-01 |
| +I |           3 |        1003 |          23 |                     2025-01-01 |
| +I |           4 |        1002 |          50 |                     2025-01-03 |
| +I |           5 |        1001 |          78 |                     2025-01-02 |
+----+-------------+-------------+-------------+--------------------------------+
```

Больше информации о других способах создания таблиц в Table API доступно в документации Flink.

Запросы и операции с таблицами

Объект `Table` предоставляет множество методов для выполнения реляционных операций. Актуальная справочная информация по API доступна в документации Flink. Основные операции с таблицами показаны в следующем примере.

```
from pyflink.table import EnvironmentSettings, TableEnvironment
from pyflink.table import DataTypes
from pyflink.table.expressions import col, call

env_settings = EnvironmentSettings.in_batch_mode() ①
t_env = TableEnvironment.create(env_settings)

txn_data = [
    (1, 1001, 100.00, "2025-01-01"),
    (2, 1002, 20.00, "2025-01-01"),
    (3, 1002, 70.00, "2025-01-01"),
    (4, 1001, 40.00, "2025-01-02"),
    (5, 1003, 50.00, "2025-01-01"),
    (6, 1001, 10.00, "2025-01-02"),
]

schema = DataTypes.ROW([
    DataTypes.FIELD("txn_id", DataTypes.INT()),
    DataTypes.FIELD("acc_id", DataTypes.INT()),
    DataTypes.FIELD("txn_value", DataTypes.DOUBLE()),
    DataTypes.FIELD("txn_date", DataTypes.STRING())
])

t_txns = t_env.from_elements(txn_data, schema)
print("Added table column:")
t_with_net_value = t_txns.add_columns( ②
    (col("txn_value") * 0.87).alias("net_value")
)
t_with_net_value.execute().print()

t_result = t_txns \ ③
    .filter(col("txn_date") != "2025-01-02") \
    .group_by(col("acc_id")) \
    .select(
    col("acc_id"),
    call("sum", col("txn_value")).alias("total_txn_value")
)
print("Filtered, groupped, and aggregated: ")
t_result.execute().print()

invoice_data = [
    (1, 1, True),
    (2, 4, False),
    (3, 3, True),
    (4, 2, True),
    (5, 6, False),
    (5, 5, True),
]
schema_i = DataTypes.ROW([
    DataTypes.FIELD("invoice_id", DataTypes.INT()),
    DataTypes.FIELD("tx_id", DataTypes.INT()),
    DataTypes.FIELD("is_confirmed", DataTypes.BOOLEAN()),
])

t_invoices = t_env.from_elements(invoice_data, schema_i)
t_joined = t_invoices.join(t_txns).where(col("tx_id") == col("txn_id")) \ ④
    .select(col("txn_id"),
            col("acc_id"),
            col("invoice_id"),
            col("is_confirmed"))
print("JOINed tables:")
t_joined.execute().print()
```

① Установка пакетного режима работы.

② Добавление столбца в таблицу.

③ Выполнение цепочки операторов и функций агрегации, таких как `filter()`, `group_by()`, `sum()`.

④ Использование Join.

Пример вывода

```
Added table column:
+-------------+-------------+--------------------------------+--------------------------------+--------------------------------
-+
|      txn_id |      acc_id |                      txn_value |                       txn_date |                       net_value
|
+-------------+-------------+--------------------------------+--------------------------------+--------------------------------
-+
|           1 |        1001 |                          100.0 |                     2025-01-01 |                            87.0
|
|           2 |        1002 |                           20.0 |                     2025-01-01 |                            17.4
|
|           3 |        1002 |                           70.0 |                     2025-01-01 |                            60.9
|
|           4 |        1001 |                           40.0 |                     2025-01-02 |                            34.8
|
|           5 |        1003 |                           50.0 |                     2025-01-01 |                            43.5
|
|           6 |        1001 |                           10.0 |                     2025-01-02 |                             8.7
|
+-------------+-------------+--------------------------------+--------------------------------+--------------------------------
-+
6 rows in set
Filtered, groupped, and aggregated:
+-------------+--------------------------------+
|      acc_id |                total_txn_value |
+-------------+--------------------------------+
|        1003 |                           50.0 |
|        1002 |                           90.0 |
|        1001 |                          100.0 |
+-------------+--------------------------------+
3 rows in set
JOINed tables:
+-------------+-------------+-------------+--------------+
|      txn_id |      acc_id |  invoice_id | is_confirmed |
+-------------+-------------+-------------+--------------+
|           5 |        1003 |           5 |         TRUE |
|           2 |        1002 |           4 |         TRUE |
|           3 |        1002 |           3 |         TRUE |
|           6 |        1001 |           5 |        FALSE |
|           1 |        1001 |           1 |         TRUE |
|           4 |        1001 |           2 |        FALSE |
+-------------+-------------+-------------+--------------+
6 rows in set
```

Операции с таблицами можно также выполнять с помощью SQL:

```
from pyflink.table import EnvironmentSettings, TableEnvironment

env_settings = EnvironmentSettings.in_streaming_mode()
table_env = TableEnvironment.create(env_settings)

table_env.execute_sql("""
    CREATE TEMPORARY TABLE t_txns (
        txn_id INT,
        acc_id INT,
        txn_value DOUBLE,
        txn_date STRING
    ) WITH (
        'connector' = 'filesystem',
        'path' = 'test.csv',
        'format' = 'csv'
    )
""")

table_env.execute_sql("""
    CREATE TABLE t_sink_print (
        acc_id INT,
        max_txn_value DOUBLE
    ) WITH (
        'connector' = 'print'
    )
""")

table_env.execute_sql("""
    INSERT INTO t_sink_print
    SELECT acc_id, MAX(txn_value) AS max_txn_value
    FROM t_txns
    GROUP BY acc_id
""").wait()
```

Пример вывода

```
15> +I[1002, 110.0]
3> +I[1001, 20.0]
2> +I[1003, 23.0]
3> -U[1001, 20.0]
3> +U[1001, 78.0]
```

Запись данных в sink-таблицу

В следующем примере показано создание таблиц типа sink, которые сохраняют данные во внешних системах:

```
from pyflink.table import EnvironmentSettings, TableEnvironment
from pyflink.table import DataTypes

env_settings = EnvironmentSettings.in_streaming_mode()
t_env = TableEnvironment.create(env_settings)
t_env.get_config().set("pipeline.jars",
                       "file:///home/konstantin/pyflink_demo/flink-sql-connector-kafka-3.3.0-1.19.jar")
mock_data = [
    (1, 1001, 50.75, "2025-01-01"),
    (2, 1002, 70.00, "2025-01-02"),
    (3, 1001, 20.25, "2025-01-02")
]
t_txns = t_env.from_elements(
    elements=mock_data,
    schema=DataTypes.ROW([
        DataTypes.FIELD("txn_id", DataTypes.INT()),
        DataTypes.FIELD("acc_id", DataTypes.INT()),
        DataTypes.FIELD("txn_value", DataTypes.DOUBLE()),
        DataTypes.FIELD("txn_date", DataTypes.STRING())
    ])
)
t_env.create_temporary_view("txn_table", t_txns) ①

t_env.execute_sql(""" ②
    CREATE TABLE sink_table (
        txn_id INT,
        acc_id INT,
        txn_value DOUBLE,
        txn_date STRING
    ) WITH (
        'connector' = 'print' ③
    )
""")

t_env.execute_sql(""" ④
    INSERT INTO sink_table
    SELECT * FROM txn_table
    WHERE `acc_id` = 1002
""").wait()

t_env.create_temporary_table( ⑤
    'kafka_sink',
    TableDescriptor.for_connector('kafka')
    .schema(Schema.new_builder()
            .column('txn_id', DataTypes.INT())
            .column('acc_id', DataTypes.INT())
            .column('txn_value', DataTypes.DOUBLE())
            .column('txn_date', DataTypes.STRING())
            .build())
    .option('topic', 'transactions_test')
    .option('properties.bootstrap.servers', 'localhost:9092')
    .format(FormatDescriptor.for_format('json')
            .build())
    .build())

t_env.execute_sql("""
    INSERT INTO kafka_sink
    SELECT * FROM txn_table
    WHERE `acc_id` = 1002
""").wait()
```

① Создание временного представления на основе таблицы.

② Создание sink-таблицы.

③ Вывод содержимого таблицы в STDOUT.

④ Вызов операции `INSERT`, которая инициирует запись данных в приемник (sink). Поскольку `INSERT` выполняется асинхронно в отдельном потоке, `wait()` необходим, чтобы дождаться вывода в STDOUT.

⑤ Создание sink-таблицы, которая использует коннектор Kafka для записи данных в Kafka-топик.

DataStream API

PyFlink DataStream API предоставляет более низкоуровневый контроль над функциями Flink, такими как управление состоянием (state) и временем обработки, использование контрольных точек, восстановление и так далее.

Центральный объект API — DataStream, который можно представить как неизменяемую коллекцию, проходящую через один или несколько этапов преобразований. Большинство приложений включает следующие шаги:

1. Создание среды выполнения.
2. Чтение данных из источника и создание объектов `DataStream`.
3. Обработка `DataStream` операторами преобразования. Каждый оператор получает на входе `DataStream` и возвращает новый `DataStream` с измененными данными.
4. Сохранение содержимого `DataStream` во внешней системе.

Создание среды выполнения

Точкой входа в приложение DataStream API является создание объекта среды выполнения (StreamExecutionEnvironment). Пример создания показан ниже.

```
menv = StreamExecutionEnvironment.get_execution_environment() ①
menv.set_parallelism(1) ②
menv.set_runtime_mode(RuntimeExecutionMode.STREAMING)

# data ingest, transformation, and sink logic

menv.execute() ③
```

① Создание объекта `StreamExecutionEnvironment`.

② Установка параметров конфигурации (параллелизм, восстановление, параметры сериализации и прочие).

③ Запуск задачи. Также можно использовать execute_async() для асинхронного запуска задачи.

Созданный `StreamExecutionEnvironment` используется для установки свойств конфигурации, создания источников (sources), приемников (sinks), операторов (operators) и для запуска задачи.

Получение данных из источника

Используя `StreamExecutionEnvironment`, можно создать источник Flink для чтения данных из внешней системы. Ниже представлены несколько способов создания источников.

Из коллекции

Следующий пример создает `DataStream`, используя Python-список в качестве источника данных:

```
from pyflink.common import Types, Configuration
from pyflink.datastream import StreamExecutionEnvironment

config = Configuration()
config.set_string("execution.runtime-mode", "STREAMING")

menv = StreamExecutionEnvironment.get_execution_environment()
menv.set_parallelism(1)

mock_data = [
    (1, 1001, 100.00, "2025-01-01"),
    (2, 1002, 20.00, "2025-01-02"),
    (3, 1003, 70.00, "2025-01-03"),
]
mds = menv.from_collection(mock_data, type_info=Types.ROW_NAMED(
       ["txn_id", "acc_id", "txn_value", "txn_date"],
       [Types.INT(), Types.INT(), Types.DOUBLE(), Types.STRING()]))
mds.print()

menv.execute()
```

Пример вывода

```
+I[1,1001,100.0,2025-01-01]
+I[2,1002,20.0,2025-01-02]
+I[3,1003,70.0,2025-01-03]
```

TIP

Элементы `+I`, `+U`, `-U` и `-D` означают операции над данными, а именно:

- `+I` — добавление новой строки.
- `+U` — предыдущая версия строки (удаляемая).
- `-U` — новая версия строки (вставляемая).
- `-D` — строка была удалена.

Из Kafka-топика

Следующий код создает источник Kafka с помощью Kafka-коннектора:

```
from pyflink.common import SimpleStringSchema, WatermarkStrategy
from pyflink.datastream import StreamExecutionEnvironment
from pyflink.datastream.connectors.kafka import KafkaSource, KafkaOffsetsInitializer

menv = StreamExecutionEnvironment.get_execution_environment()
menv.add_jars("file:///home/konstantin/pyflink_demo/flink-sql-connector-kafka-3.3.0-1.19.jar") ①

source_kafka = KafkaSource.builder() \ ②
    .set_bootstrap_servers("<host>:<port>") \
    .set_topics("m_topic") \
    .set_group_id("my_group") \
    .set_starting_offsets(KafkaOffsetsInitializer.earliest()) \
    .set_value_only_deserializer(SimpleStringSchema()) \
    .build()
mds = menv.from_source(source_kafka, ③
                       WatermarkStrategy.for_monotonous_timestamps(),
                       "Test Kafka source")

menv.execute()
```

① Добавляет JAR Kafka-коннектора в контекст задачи PyFlink. Для получения актуальной версии коннектора Kafka можно обратиться к документации Flink.

② Создание источника Kafka на основе топика Kafka.

③ Получение `DataStream` от источника Kafka.

Из файла

В примере ниже Flink считывает строки CSV-файла и возвращает `DataStream`. Каждая строка является отдельной записью в потоке.

```
from pyflink.common import WatermarkStrategy
from pyflink.datastream import StreamExecutionEnvironment
from pyflink.datastream.connectors.file_system import FileSource, StreamFormat

menv = StreamExecutionEnvironment.get_execution_environment()
file_source = FileSource.for_record_stream_format(
    StreamFormat.text_line_format(), ①
    "test.csv"
).build()
wm_strategy = WatermarkStrategy.for_monotonous_timestamps()

ds = menv.from_source(source=file_source, ②
                      source_name="test_file_source",
                      watermark_strategy=wm_strategy,

)
ds.print()
menv.execute()
```

① Использование встроенного StreamFormat, который позволяет читать файлы построчно и преобразовывать каждую строку в отдельную запись.

② Использование встроенной реализации водяного знака (watermark), предполагающей, что записи `DataStream` всегда поступают в нужном порядке.

Пример вывода

```
10> 1,1001,20,2025-01-02
10> 2,1002,110,2025-01-01
10> 3,1003,23,2025-01-01
10> 4,1002,50,2025-01-03
10> 5,1001,78,2025-01-02
```

РЕКОМЕНДАЦИЯ

`10>` указывает на идентификатор задачи или индекс подзадачи.

Операторы преобразования

PyFlink предоставляет множество операторов, таких как `map()`, `filter()`, `reduce()` и прочих. Каждый оператор получает на входе объект `DataStream`, выполняет преобразования данных и возвращает новый `DataStream`. Основные операции показаны ниже:

```
from pyflink.common import Types, Row
from pyflink.datastream import StreamExecutionEnvironment, MapFunction

menv = StreamExecutionEnvironment.get_execution_environment()
menv.set_parallelism(1)
mock_data = [
    (1, 1001, 100.00, "2025-01-01"),
    (2, 1002, 20.00, "2025-01-02"),
    (3, 1003, 70.00, "2025-01-03"),
]
ds = menv.from_collection(mock_data, type_info=Types.ROW_NAMED(
    ["txn_id", "acc_id", "txn_value", "txn_date"],
    [Types.INT(), Types.INT(), Types.DOUBLE(), Types.STRING()]))
print("Original DataStream:")
ds.print()

ds_filtered = ds.filter(lambda txn: txn["txn_value"] > 50.0) ①
print("Filtered DataStream:")
ds_filtered.print()

ds_mapped = ds.map( ②
    lambda txn: Row(
        txn["txn_id"],
        9999 if txn["acc_id"] == 1003 else txn["acc_id"],
        txn["txn_value"],
        txn["txn_date"]
    ), output_type=Types.ROW_NAMED(
        ["txn_id", "acc_id", "txn_value", "txn_date"],
        [Types.INT(), Types.INT(), Types.DOUBLE(), Types.STRING()]))
print("Lambda-mapped DataStream:")
ds_mapped.print()

class UpdateAccID(MapFunction): ③
    def map(self, value):
        txn_id, acc_id, txn_value, txn_date = value
        if acc_id < 2000:
            acc_id = acc_id * 2
        return (Row(txn_id, acc_id, txn_value, txn_date))

ds_mapped_class = ds.map(UpdateAccID(), ④
                         output_type=Types.ROW_NAMED(
                             ["txn_id", "acc_id", "txn_value", "txn_date"],
                             [Types.INT(), Types.INT(), Types.FLOAT(), Types.STRING()]
                         ))
print("MapFunction-mapped DataStream:")
ds_mapped_class.print()

menv.execute()
```

① Использование filter().

② Применение map() с лямбда-функцией.

③ Реализация класса `MapFunction` для map-оператора. Метод `map()` вызывается для каждого элемента `DataStream`.

④ Выполнение трансформации `map()` с использованием реализации `MapFunction`.

Пример вывода

```
Original DataStream:
+I[1,1001,100.0,2025-01-01]
+I[2,1002,20.0,2025-01-02]
+I[3,1003,70.0,2025-01-03]

Filtered DataStream:
+I[1,1001,100.0,2025-01-01]
+I[3,1003,70.0,2025-01-03]

Lambda-mapped DataStream:
+I[1,1001,100.0,2025-01-01]
+I[2,1002,20.0,2025-01-02]
+I[3,9999,70.0,2025-01-03]

Custom MapFunction-mapped DataStream:
+I[1,2002,100.0,2025-01-01]
+I[2,2004,20.0,2025-01-02]
+I[3,2006,70.0,2025-01-03]
```

ПРИМЕЧАНИЕ

Строки вывода могут быть перемешаны, поскольку каждый оператор `DataStream` отправляет данные в STDOUT из отдельного потока.

Запись данных в приемник

Результаты вычислений Flink можно записать во внешнюю систему с помощью приемника (sink).

Запись в файл

В следующем примере содержимое `DataStream` записывается в файл с помощью коннектора FileSystem.

```
from pyflink.common import Types, Encoder
from pyflink.datastream import StreamExecutionEnvironment
from pyflink.datastream.connectors.file_system import FileSink, RollingPolicy

OUTPUT_PATH = "file:///home/konstantin/pyflink_demo/fs_sink"
menv = StreamExecutionEnvironment.get_execution_environment()
menv.set_parallelism(1)
mock_data = [
    (1, 1001, 100.00, "2025-01-01"),
    (2, 1002, 20.00, "2025-01-02"),
    (3, 1003, 70.00, "2025-01-03"),
]
mds = menv.from_collection(mock_data, type_info=Types.ROW_NAMED(
    ["txn_id", "acc_id", "txn_value", "txn_date"],
    [Types.INT(), Types.INT(), Types.DOUBLE(), Types.STRING()]))
fs_sink = FileSink.for_row_format(OUTPUT_PATH, ①
                                  Encoder.simple_string_encoder("UTF-8")) \
    .with_rolling_policy(RollingPolicy.default_rolling_policy( ②
    part_size=1024 ** 3,
    rollover_interval=15 * 60 * 1000,
    inactivity_interval=5 * 60 * 1000)) \
    .build()

mds.sink_to(fs_sink) ③
menv.execute()
```

① Создание приемника типа `FileSink` с использованием формата построчного кодирования (row-encoded format).

② Установка политики ротации.

③ Запись содержимого `DataStream` в приемник.

Данный пример создает директорию, указанную в поле `OUTPUT_PATH`, например:

```
/home/konstantin/pyflink_demo/fs_sink
└─ /2025-04-08--13
    └─ part-4a8b2a20-187d-4ad6-ae7a-8f0e6dc880c5-0
```

Каждая запись `DataStream` записывается в файл в виде новой строки:

```
+I[1, 1001, 100.0, 2025-01-01]
+I[2, 1002, 20.0, 2025-01-02]
+I[3, 1003, 70.0, 2025-01-03]
```

Запись в файл ORC

Для записи данных в формате ORC, Parquet или Avro используйте bulk-encoded форматы. Ниже показан пример задачи, которая записывает данные в файл в формате ORC, используя `FileSink`.

```
from pyflink.common import Types, Configuration
from pyflink.datastream import StreamExecutionEnvironment
from pyflink.datastream.connectors.file_system import FileSink
from pyflink.datastream.formats.orc import OrcBulkWriters
from pyflink.table import DataTypes

menv = StreamExecutionEnvironment.get_execution_environment()
menv.set_parallelism(1)
menv.add_jars("file:///home/konstantin/pyflink_demo/flink-sql-orc-1.19.2.jar") ①
OUTPUT_PATH = "file:///home/konstantin/pyflink_demo/fs_sink_orc"
mock_data = [
    (1, 1001, 100.00, "2025-01-01"),
    (2, 1002, 20.00, "2025-01-02"),
    (3, 1003, 70.00, "2025-01-03"),
]
mds = menv.from_collection(mock_data,
                           type_info=Types.ROW_NAMED(
                               ["txn_id", "acc_id", "txn_value", "txn_date"],
                               [Types.INT(), Types.INT(), Types.DOUBLE(), Types.STRING()]))

row_type = DataTypes.ROW([
    DataTypes.FIELD('txn_id', DataTypes.INT()),
    DataTypes.FIELD('acc_id', DataTypes.INT()),
    DataTypes.FIELD('txn_value', DataTypes.DOUBLE()),
    DataTypes.FIELD('txn_date', DataTypes.STRING()),
])

file_sink_orc = FileSink.for_bulk_format(
    OUTPUT_PATH,
    OrcBulkWriters.for_row_type( ②
        row_type=row_type,
        writer_properties=Configuration(),
        hadoop_config=Configuration(),
    )
).build()

mds.sink_to(file_sink_orc)
menv.execute()
```

① Добавление зависимостей для работы с ORC.

② Использование формата ORC с `FileSink`.

Запись в Kafka-топик

Следующий код направляет содержимое `DataStream` в топик Kafka, используя приемник Kafka. Соединение с сервером устанавливается с помощью коннектора.

```
menv = StreamExecutionEnvironment.get_execution_environment()
menv.set_parallelism(1)
menv.add_jars("file:///D:/dev/py/pyflink/flink-sql-connector-kafka-3.3.0-1.19.jar") ①
mock_data = [
    (1, 1001, 100.00, "2025-01-01"),
    (2, 1002, 20.00, "2025-01-02"),
    (3, 1003, 70.00, "2025-01-03"),
]
mds = menv.from_collection(mock_data, type_info=Types.ROW_NAMED(
       ["txn_id", "acc_id", "txn_value", "txn_date"],
       [Types.INT(), Types.INT(), Types.DOUBLE(), Types.STRING()]))

kafka_sink = KafkaSink.builder() \ ②
    .set_bootstrap_servers("<host>:<port>") \
    .set_record_serializer(
        KafkaRecordSerializationSchema.builder()
            .set_topic("m_topic")
            .set_value_serialization_schema(SimpleStringSchema())
            .build()
    ) \
    .set_delivery_guarantee(DeliveryGuarantee.AT_LEAST_ONCE) \
    .build()

mds.sink_to(kafka_sink) ③
menv.execute()
```

① Добавление коннектора Kafka в контекст задачи PyFlink. Актуальная версия коннектора доступна в документации Flink.

② Создание объекта приемника, который подключается к брокеру Kafka, используя значения `"<host>:<port>"`.

③ Запись содержимого `DataStream` в приемник Kafka.

Управление зависимостями

В приложениях PyFlink могут использоваться сторонние зависимости, например: библиотеки Python, JAR-файлы коннекторов, фреймворки ML и прочие. Способы их импорта отличаются для Table API и DataStream API и показаны ниже.

Зависимости JAR

Table API

```
table_env.get_config().set("pipeline.jars", "file:///path/to/myJar1.jar;file:///path/to/myJar2.jar")
table_env.get_config().set("pipeline.classpaths", "file:///path/to/myJar1.jar;file:///path/to/myJar2.jar")
```

DataStream API

```
stream_ex_env.add_jars("file:///path/to/myJar1.jar;file:///path/to/myJar2.jar")
stream_ex_env.add_classpaths("file:///path/to/myJar1.jar", "file:///path/to/myJar2.jar")
```

Параметр CLI

```
$ flink run \
      --python test_flink_job.py \
      --jarfile path/to/myJar1.jar
```

Зависимости Python

Table API

```
table_env.add_python_file("path/to/myScript.py")
```

DataStream API

```
stream_execution_environment.add_python_file("path/to/myScript.py")
```

Параметр CLI

```
$ flink run \
      --python test_flink_job.py \
      --pyFiles file:///path/to/test.py
```

Flink SQL Gateway

Константин Алпашкин

Содержание

Назад к содержанию

Работа с REST endpoint
Пример использования

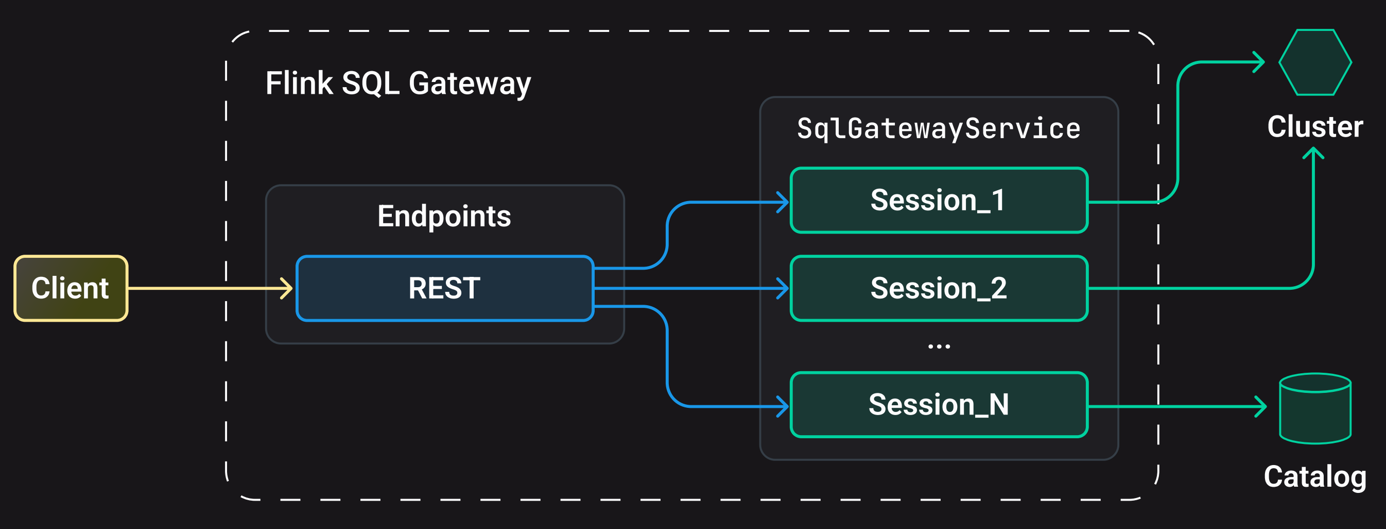
Flink SQL Gateway — это сервис, позволяющий нескольким клиентам параллельно выполнять запросы Flink SQL. Используя SQL Gateway, можно запускать задачи Flink, выполнять поиск по метаданным и анализировать данные в реальном времени.

SQL SQL Gateway состоит из следующих компонентов:

- SqlGatewayService, основной задачей которого является обработка SQL-запросов.
- Подключаемые endpoints, через которые пользователи могут отправлять запросы на сервер.

На момент релиза ADH 3.1.2.b1, основной поддерживаемый endpoint — REST endpoint, подразумевающий взаимодействие с сервисом по HTTP.

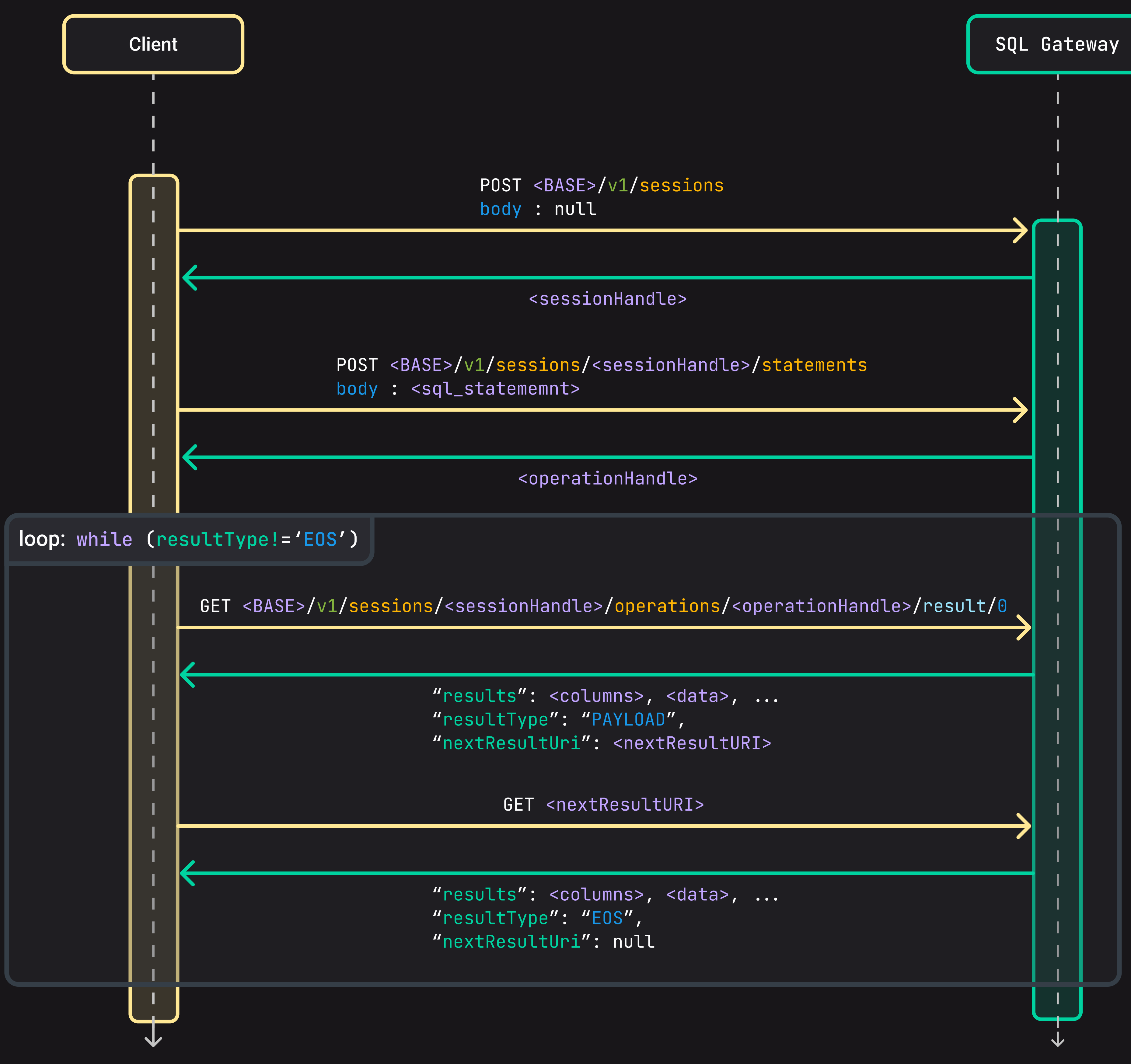
Ниже представлена высокоуровневая архитектура Flink SQL Gateway.



Архитектура Flink SQL Gateway

Работа с REST endpoint

Схема взаимодействия с использованием REST endpoint представлена ниже.


Client
SQL Gateway
POST <BASE>/v1/sessions
body : null
<sessionHandle>
POST <BASE>/v1/sessions/<sessionHandle>/statements
body : <sql_statememnt>
<operationHandle>
loop: while (resultType!='EOS')
GET <BASE>/v1/sessions/<sessionHandle>/operations/<operationHandle>/result/0
"results": <columns>, <data>, ...
"resultType": "PAYLOAD",
"nextResultUri": <nextResultURI>
GET <nextResultURI>
"results": <columns>, <data>, ...
"resultType": "EOS",
"nextResultUri": null

Взаимодействие через REST endpoint

Основные этапы схемы:

1. **Создание сессии**. Клиент отправляет POST-запрос на создание сессии. SQL Gateway создает новую сессию и возвращает ее идентификатор — `sessionHandle`, который нужен для дальнейшего общения с сервером. Сессия существует в течение настраиваемого периода времени, а срок ее жизни может быть продлен вручную.
2. **Отправка запроса**. После создания сессии клиент отправляет SQL-запрос на сервер SQL Gateway. Для каждого запроса SQL Gateway создает новую сущность Operation и возвращает ее идентификатор (`operationHandle`), необходимый в будущем для получения результатов запроса. Экземпляр Operation может быть принудительно отменен/закрыт для освобождения ресурсов.
3. **Получение результатов**. Используя `operationHandle`, клиент получает результаты из ранее созданного экземпляра Operation. Если вычисления по запросу готовы, SQL Gateway возвращает пакет результатов и URI, указывающий на следующий пакет. Если SQL Gateway вернул все результаты для данного запроса, возвращается специальный ответ с полем `"resultType": "EOS"`.

REST URL сервиса SQL Gateway доступен на странице ***Clusters*** → ***<YOUR_CLUSTER>*** → ***Services*** → ***Flink*** → ***Info*** в ADCM, например: *http://ka-adh-1.ru-central1.internal:8083/v1/info*. Полная справочная информация о SQL Gateway REST API доступна в документации Flink.

Пример использования

Ниже представлен пример взаимодействия с SQL Gateway, демонстрирующий основные операции с использованием REST endpoint. Вы можете выполнить следующие шаги, используя curl, или можете скачать Postman-коллекцию с готовыми запросами.

1. Проверьте доступность сервиса SQL Gateway.

```
$ curl http://<sql-gateway-host>:8083/v1/info
```

   В ответе содержится базовая информация о сервисе.

```
{"productName":"Apache Flink","version":"1.16.2"}
```

2. Создайте новую сессию.

```
$ curl -X POST http://<sql-gateway-host>:8083/v1/sessions
```

   Сервер возвращает `sessionHandle`, который уникально идентифицирует созданную сессию.

```
{"sessionHandle":"97fa59cf-6aa9-440b-b5cb-191f4167f0fb"}
```

3. Используя полученный `sessionHandle`, отправьте SQL-запрос.

```
$ curl -X POST http://<sql-gateway-host>:8083/v1/sessions/{sessionHandle}/statements/ -d @test_query.json
```

   В этом примере файл *test_query.json* содержит тестовый SQL-запрос, выполняющий подсчет количества слов, и выглядит следующим образом:

```
{
    "statement": "SELECT word, SUM(frequency) AS `count` FROM (   VALUES ('Hello', 1), ('Ciao', 1), ('Hello', 2) ) AS WordTable(word, frequency) GROUP BY word"
}
```

   В ответе содержится `operationHandle`, который идентифицирует отправленный запрос.

```
{"operationHandle":"b18e38b0-8a9b-4f36-9121-7d9f2eb46b81"}
```

4. Получите результаты запроса.

```
$ curl http://<sql-gateway-host>:8083/v1/sessions/{sessionHandle}/operations/{operationHandle}/result/0
```

   Ответ от сервера имеет следующий вид.

```
{"results":{"columns":[{"name":"word","logicalType":{"type":"VARCHAR","nullable":false,"length":5},"comment":null},
{"name":"count","logicalType":{"type":"INTEGER","nullable":false},"comment":null}],"data":[{"kind":"INSERT","fields":
["Hello",1]},{"kind":"INSERT","fields":["Ciao",1]},{"kind":"UPDATE_BEFORE","fields":["Hello",1]},{"kind":"UPDATE_AFTER","fields":
["Hello",3]}]},"resultType":"PAYLOAD","nextResultUri":"/v1/sessions/b9ea29d4-e373-463f-bd9a-a943d853b093/operations/c5df0878-
e102-4e02-a37d-3293aa53c038/result/1"}
```

   Обратите внимание на поля `"resultType":"PAYLOAD"` и `"nextResultUri":"<nextURI>"`. В последнем содержится URI, указывающий на следующий пакет с данными. Если при обращении к `<nextURI>` ответ содержит `"resultType": "EOS"`, это означает, что SQL Gateway вернул все результаты для данного запроса.

5. Чтобы продлить срок действия сессии, отправьте heartbeat-запрос.

```
$ curl -X POST http://<sql-gateway-host>:8083/v1/sessions/{sessionHandle}/heartbeat
```

   Ответ `200 OK` указывает, что таймер жизни сессии был сброшен.

Кроме вышеупомянутых запросов существуют и другие полезные операции, например, отмена/остановка/проверка статуса операции и так далее. Подробная информация о REST API доступна в документации Flink.

Запуск Flink в YARN

Константин Алпашкин

Содержание

Назад к содержанию

- Обзор
- Подготовка пользователя к взаимодействию с YARN
- Режимы работы Flink в YARN
  - Режим приложения
  - Режим сессии

Данная статья описывает использование сервиса Flink в режиме YARN.

Обзор

Изначально сервис Flink поддерживает два режима развертывания:

- **Режим standalone**. Данный режим предполагает, что компоненты TaskManager и JobManager установлены на хостах ADH-кластера. В этом режиме запуск и остановка кластера Flink выполняется с помощью ADCM, а кластер постоянно активен, ожидая новых Flink-задач. Количество ресурсов, выделяемых компонентам JobManager/TaskManager, фиксировано и указывается с помощью настроек Flink-сервиса.
- **Режим YARN**. В этом режиме сервис Flink не требует установки компонентов JobManager/TaskManager на хостах ADH-кластера. Для выполнения Flink-задач эти компоненты создаются динамически в контейнерах YARN, а затем утилизируются (жизненный цикл контейнеров зависит от используемого режима YARN). При запуске Flink-приложения в режиме YARN JAR-файл передается компоненту YARN ResourceManager, который создает YARN-контейнеры на хостах ADH под управлением YARN NodeManager. Непосредственно на этапе выполнения кода Flink динамически аллоцирует ресурсы для TaskManager. Подробности использования Flink в YARN с примерами приведены далее в статье.

Ключевым отличием режима standalone от YARN является то, где именно код Flink-приложения компилируется в JobGraph. В standalone-режиме компиляция выполняется компонентом Flink client, а затем JobGraph отправляется в JobManager. Как следствие, компоненты Flink client могут потреблять значительное количество ресурсов CPU/RAM и дополнительно создавать нагрузку на сеть, перемещая объекты JobGraph. В режиме YARN метод `main()` приложения Flink выполняется в выделенном YARN-контейнере с JobManager, что снижает нагрузку на компоненты Flink client.

Подготовка пользователя к взаимодействию с YARN

Перед использованием Flink в YARN выполните следующие шаги для подготовки локального пользователя к взаимодействию с YARN-кластером.

1. Создайте директорию в HDFS и установите права владельца. YARN использует эту директорию для хранения временных данных.

```
$ hadoop fs -mkdir /user/<username>
$ hadoop fs -chown -R <username>:hadoop /user/<username>
```

   где `<username>` — локальный пользователь, от которого запускаются команды `flink run …`.

2. Если кластер керберизирован и имя локального пользователя отличается от принципала, укажите сопоставление пользователь-принципал, используя правила `auth_to_local`. Например:

```
RULE:[1:$1@$0](k_alpashkin_krb2@AD.RANGER-TEST)s/.*/admin/
```

3. В сервисе Core configuration выполните действие ***Update Core configuration***.
4. Перезапустите сервисы HDFS, YARN и Flink.
5. Если плагины Ranger включены для HDFS и YARN, предоставьте соответствующие права пользователю.

Режимы работы Flink в YARN

Кластер Flink может работать в YARN в двух режимах:

- Режим приложения
- Режим сессии

Ниже описаны особенности каждого из режимов с примерами.

Режим приложения

При запуске Flink в YARN в режиме приложения (application mode) создается временный кластер Flink для выполнения одного Flink-приложения. После завершения работы приложения кластер Flink автоматически уничтожается. Данный режим обеспечивает наилучшую изоляцию при выполнении нескольких Flink-задач, поскольку задачи выполняются в выделенных кластерах Flink, не мешая друг другу.

Далее отражены основные события, которые происходят при запуске Flink в YARN в режиме приложения:

1. JAR-файл с приложением Flink передается клиенту Flink.
2. Клиент Flink запрашивает у YARN выделенный контейнер (Application Master) для развертывания в нем компонента JobManager. Код приложения со всеми зависимостями переносится в контейнер JobManager. Метод `main()` выполняется внутри этого контейнера.
3. В процессе выполнения JobManager запрашивает у YARN дополнительные контейнеры для TaskManager, которые необходимы для выполнения отдельных Flink-задач. Количество создаваемых TaskManager зависит от параметров запуска приложения и настроек сервиса Flink (параллелизм, количество слотов).
4. После завершения задачи (успешного, неудачного или отмены) все контейнеры, выделенные YARN, уничтожаются.

Ниже показан пример использования Flink в YARN в режиме приложения:

1. Запустите приложение Flink, указав путь к JAR-файлу:

```
$ flink run-application \ ①
-t yarn-application \ ②
/usr/lib/flink/examples/batch/WordCount.jar
```

   ① Команда `run-application` запускает кластер Flink в режиме приложения.

   ② Указывает YARN в качестве среды для развертывания.

   В выводе содержится информация о взаимодействии с компонентами YARN, а также Application Master ID:

```
Setting HBASE_CONF_DIR=/etc/hbase/conf because no HBASE_CONF_DIR was set.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/flink/lib/log4j-slf4j-impl-2.17.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
2025-06-24 15:37:27,169 INFO  org.apache.hadoop.yarn.client.DefaultNoHARMFailoverProxyProvider [] - Connecting to
ResourceManager at adh-ka-2.ru-central1.internal/10.92.41.75:8032
2025-06-24 15:37:27,411 INFO  org.apache.hadoop.yarn.client.AHSProxy                         [] - Connecting to Application
History server at adh-ka-3.ru-central1.internal/10.92.43.237:10200
2025-06-24 15:37:27,429 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - No path for the flink jar
passed. Using the location of class org.apache.flink.yarn.YarnClusterDescriptor to locate the jar
2025-06-24 15:37:27,668 INFO  org.apache.hadoop.conf.Configuration                         [] - resource-types.xml not found
2025-06-24 15:37:27,669 INFO  org.apache.hadoop.yarn.util.resource.ResourceUtils          [] - Unable to find 'resource-types.
xml'.
2025-06-24 15:37:27,758 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - Cluster specification:
ClusterSpecification{masterMemoryMB=2048, taskManagerMemoryMB=2048, slotsPerTaskManager=1}
2025-06-24 15:37:28,517 WARN  org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory      [] - The short-circuit local reads
feature cannot be used because libhadoop cannot be loaded.
2025-06-24 15:37:29,279 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - Cannot use kerberos delegation
token manager, no valid kerberos credentials provided.
2025-06-24 15:37:29,304 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - Submitting application master
application_1750778541795_0003
2025-06-24 15:37:29,363 INFO  org.apache.hadoop.yarn.client.api.impl.YarnClientImpl        [] - Submitted application
application_1750778541795_0003
2025-06-24 15:37:29,363 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - Waiting for the cluster to be
allocated
2025-06-24 15:37:29,367 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - Deploying cluster, current
state ACCEPTED
2025-06-24 15:37:34,956 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - YARN application has been
deployed successfully.
2025-06-24 15:37:34,957 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - Found Web Interface adh-ka-2.
ru-central1.internal:8081 of application 'application_1750778541795_0003'.
```

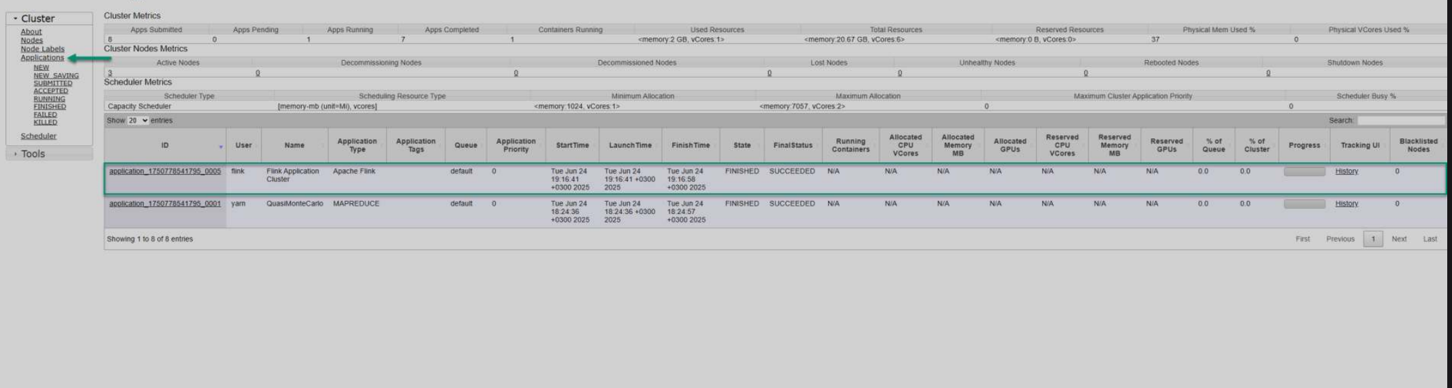
2. После запуска приложения вы можете использовать такие команды, как отмена, получение списка задач или создание точки сохранения (savepoint). Например:

```
$ flink list \ ①
  -t yarn-application \
  -Dyarn.application.id=application_1750778541795_0003

$ flink cancel \ ②
  -t yarn-application \
  -Dyarn.application.id=application_1750778541795_0003 <jobId>
```

   ① Выводит список задач Flink, принадлежащих YARN-приложению `application_1750778541795_0003`.

   ② Отменяет выполнение Flink-задачи `<jobId>`. Отмена задачи останавливает кластер Flink.

3. Проверьте статус приложения в веб-интерфейсе YARN ResourceManger. Актуальная ссылка на веб-интерфейс доступна в ADCM (***Clusters*** → ***<ADH_cluster>*** → ***Services*** → ***YARN*** → ***Info***).



Web-интерфейс YARN

РЕКОМЕНДАЦИЯ

Для более "легковесного" запуска задач используйте параметр `yarn.provided.lib.dirs`, предварительно загрузив JAR-файлы Flink в директорию HDFS, доступную на всех хостах ADH. В этом случае клиент Flink не переносит массивные JAR-файлы в JobManager, так как они могут быть считаны напрямую из HDFS. Например:

```
$ flink run-application \
    -t yarn-application \
    -Dyarn.provided.lib.dirs="hdfs://user/admin/flink_demo/test_flink.jar" \
    hdfs://user/admin/flink_demo/test_flink.jar
```

Дополнительная информация об использовании Flink с YARN в режиме приложения доступна в документации Flink.

Режим сессии

В этом режиме один кластер Flink (обычно долго работающий) разворачивается в контейнерах YARN для обслуживания множества Flink-задач. Когда одна задача Flink завершается, сессия Flink-кластера остается активной.

Ниже представлена последовательность событий, которые происходят при запуске Flink в YARN в режиме сессии:

1. Клиент Flink запускает сессию YARN с Flink-кластером. Для этого создается YARN-контейнер с JobManager (Application Master). Контейнеры для TaskManager инициализируются динамически с учетом параметров запуска и конфигурации сервиса Flink.
2. ID запущенного приложения YARN сохраняется (например, с помощью веб-интерфейса YARN, логов или команды `yarn application -list`).
3. С помощью клиента Flink одна или несколько задач отправляется на выполнение в Flink-кластер.
4. При необходимости сессия Flink-кластера завершается вручную.

Ниже показан пример использования Flink в YARN в режиме сессии:

1. Запустите сессию Flink-кластера в YARN:

```
$ flink-yarn-session \
    -d \ ①
    -nm my_flink_job \ ②
    -s 2 \ ③
    -tm 2048 ④
    -yj /usr/lib/flink/lib/flink-dist-1.20.1.jar ⑤
```

   ① Запуск сессии в фоновом (detached) режиме.

   ② Указание пользовательского имени для YARN-приложения.

   ③ Установка количества слотов для TaskManager.

   ④ Установка максимального объема памяти в мегабайтах, выделяемой для контейнеров с TaskManager.

   ⑤ Переопределение параметра конфигурации `yarn.flink-dist-jar`. Без указания данного параметра в логах может появиться ошибка: `No path for the flink jar passed…`.

2. Получите ID приложения YARN. Для этого используйте веб-интерфейс YARN или команду:

```
$ yarn application -list
```

   Вывод содержит ID YARN-приложения:

```
WARNING: YARN_OPTS has been replaced by HADOOP_OPTS. Using value of YARN_OPTS.
2025-06-26 14:24:07,737 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at adh-ka-2.ru-central1.
internal/10.92.41.75:8032
2025-06-26 14:24:07,974 INFO client.AHSProxy: Connecting to Application History server at adh-ka-3.ru-central1.internal/10.92.
43.237:10200
Total number of applications (application-types: [], states: [SUBMITTED, ACCEPTED, RUNNING] and tags: []):1
                Application-Id      Application-Name        Application-Type          User           Queue
State             Final-State             Progress                        Tracking-URL
application_1750778541795_0007           my_flink_job            Apache Flink         flink         default
RUNNING               UNDEFINED                 100%            http://10.92.43.237:8081
```

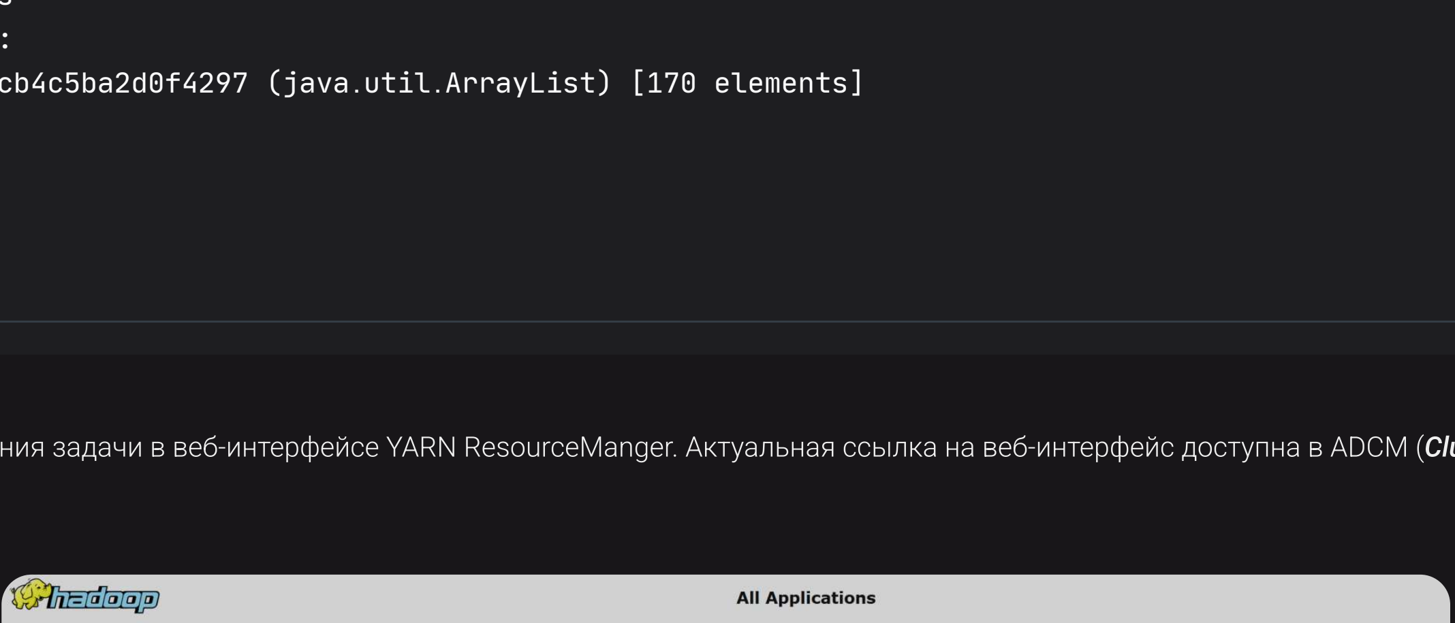
3. Запустите приложение Flink, подключившись к сессии YARN:

```
$ flink run
    -t yarn-session
    -Dyarn.application.id=application_1750778541795_0007
    /usr/lib/flink/examples/batch/WordCount.jar
```

   Пример вывода

```
Setting HBASE_CONF_DIR=/etc/hbase/conf because no HBASE_CONF_DIR was set.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/flink/lib/log4j-slf4j-impl-2.17.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
2025-06-26 14:27:37,600 INFO  org.apache.flink.yarn.cli.FlinkYarnSessionCli                [] - Found Yarn properties file
under /var/tmp/.yarn-properties-flink.
2025-06-26 14:27:37,600 INFO  org.apache.flink.yarn.cli.FlinkYarnSessionCli                [] - Found Yarn properties file
under /var/tmp/.yarn-properties-flink.
Executing WordCount example with default input data set.
Use --input to specify file input.
Printing result to stdout. Use --output to specify output path.
2025-06-26 14:27:38,422 INFO  org.apache.hadoop.yarn.client.DefaultNoHARMFailoverProxyProvider [] - Connecting to
ResourceManager at adh-ka-2.ru-central1.internal/10.92.41.75:8032
2025-06-26 14:27:38,659 INFO  org.apache.hadoop.yarn.client.AHSProxy                       [] - Connecting to Application
History server at adh-ka-3.ru-central1.internal/10.92.43.237:10200
2025-06-26 14:27:38,673 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - No path for the flink jar
passed. Using the location of class org.apache.flink.yarn.YarnClusterDescriptor to locate the jar
2025-06-26 14:27:38,815 INFO  org.apache.flink.yarn.YarnClusterDescriptor                  [] - Found Web Interface adh-ka-
3.ru-central1.internal:8081 of application 'application_1750778541795_0007'.
Job has been submitted with JobID 5fb3965a1ed2ee3ed5790a3fe4df84c1
Program execution finished
Job with JobID 5fb3965a1ed2ee3ed5790a3fe4df84c1 has finished.
Job Runtime: 7027 ms
Accumulator Results:
- 180f560f3af3522f5cb4c5ba2d0f4297 (java.util.ArrayList) [170 elements]

(a,5)
(action,1)
...
```

4. Проверьте детали выполнения задачи в веб-интерфейсе YARN ResourceManger. Актуальная ссылка на веб-интерфейс доступна в ADCM (***Clusters*** → ***<ADH_cluster>*** → ***Services*** → ***YARN*** → ***Info***).



Web-интерфейс YARN

5. Когда сессия Flink-кластера больше не нужна, завершите приложение YARN с помощью команды:

```
$ yarn application -kill application_1750778541795_0007
```

   Вывод:

```
2025-06-26 14:33:19,976 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at adh-ka-2.ru-central1.
internal/10.92.41.75:8032
2025-06-26 14:33:20,226 INFO client.AHSProxy: Connecting to Application History server at adh-ka-3.ru-central1.internal/10.92.
43.237:10200
Killing application application_1750778541795_0007
2025-06-26 14:33:20,609 INFO impl.YarnClientImpl: Killed application application_1750778541795_0007
```

Дополнительная информация об использовании Flink в режиме сессии YARN доступна в документации Flink.

Логирование в Flink

Константин Алпашкин

Содержание

Назад к содержанию

Расположение и формат лог-файлов

Типы логов Flink

Настройка логирования в ADCM

- Файл log4j.properties

Просмотр логов в Flink UI

Логирование в приложениях Flink

Сервис Flink генерирует текстовые логи, которые могут быть полезны при анализе причин различных ошибок, возникающих при работе с сервисом. По умолчанию Flink использует фасад протоколирования SLF4J, а также Log4j 2 в качестве фреймворка логирования. Также допускается использование любого SLF4J-совместимого фреймворка, например Log4j 1, Logback и так далее.

Расположение и формат лог-файлов

По умолчанию Flink хранит логи в локальной директории */var/log/flink* на хостах ADH, где установлены компоненты Flink. Пример содержимого директории */var/log/flink/* показан ниже.

```
$ ls -l /var/log/flink
```

```
-rw-rw-r-- 1 admin admin 37329 Mar 13 11:34 flink-admin-client-ka-adh-6.ru-central1.internal.log
-rw-r--r-- 1 flink flink 77921 Mar 13 11:34 flink-flink-standalonesession-0-ka-adh-6.ru-central1.internal.log
-rw-r--r-- 1 flink flink   457 Mar 13 11:07 flink-flink-standalonesession-0-ka-adh-6.ru-central1.internal.out
-rw-r--r-- 1 flink flink 70599 Mar 13 11:34 flink-flink-taskexecutor-0-ka-adh-6.ru-central1.internal.log
-rw-r--r-- 1 flink flink  3084 Mar 13 11:34 flink-flink-taskexecutor-0-ka-adh-6.ru-central1.internal.out
-rw-r--r-- 1 flink flink    73 Mar 13 11:07 flink-jobmanager.out
-rw-r--r-- 1 root  root  38489 Mar 13 11:34 flink-root-client-ka-adh-6.ru-central1.internal.log
-rw-r--r-- 1 flink flink    68 Mar 13 11:07 flink-taskmanager.out
```

Типы логов Flink

Компоненты Flink генерируют несколько типов лог-файлов, описанных ниже.

Имя лог-файла	Атрибуты	Комментарий
flink-<user_name>-client-<host_name>.log	- `<host_name>` — имя хоста; - `<user_name>` — имя пользователя, запустившего задачу Flink.	Клиентские логи Flink
flink-flink-standalonesession-<n>-<host_name>.log	- `<host_name>` — имя хоста; - `<n>` — индекс сессии.	Логи YARN
flink-flink-taskexecutor-<n>-<host_name>.log	- `<host_name>` — имя хоста; - `<n>` — индекс задачи.	Логи TaskExecutor
flink-jobmanager.out	—	Вывод Flink JobManager
flink-taskmanager.out	—	Вывод Flink TaskManager

Настройка логирования в ADCM

С использованием Log4j настройка процесса логирования в сервисе Flink сводится к модификации конфигурационных файлов *log4j.properties*. Сервис Flink использует два основных файла конфигурации:

- *log4j.properties*. Файл конфигурации, используемый компонентами JobManager/TaskManager.
- *log4j-cli.properties*. Используется для логирования событий, связанных с интерфейсом командной строки Flink.

Данные файлы хранятся в директории */etc/flink/conf* на хостах с установленными компонентами Flink. Однако вместо того, чтобы редактировать файлы вручную на каждом хосте, ADCM предоставляет возможность обновить конфигурацию Log4j на всех хостах ADH одновременно. Для этого:

1. В ADCM UI перейдите на страницу ***Clusters*** → ***<cluster_name>*** → ***Services*** → ***Flink*** → ***Primary Configuration*** и активируйте опцию ***Show advanced***.
2. Выберите раздел ***log4j.properties/log4j-cli.properties*** для редактирования конфигурации. Конфигурация Log4j, указанная через ADCM UI, перезапишет содержимое файлов */etc/flink/conf/log4j*.properties* при рестарте сервиса Flink. Подробная информация о конфигурационных свойствах Log4j доступна в документации Log4j.

> **ПРИМЕЧАНИЕ**
>
> Обновление конфигурационных файлов Log4j не требует перезапуска сервиса Flink. Данные файлы периодически сканируются (по умолчанию каждые 30 секунд) и могут обновляться "на лету".

Ниже приведены параметры конфигурации сервиса Flink, связанные с логированием.

Параметры логирования сервиса Flink

Раздел настроек	Конфигурационный параметр	Значение по умолчанию	Описание
flink-env.sh	FLINK_LOG_DIR	/var/log/flink	Корневая директория лог-файлов Flink
flink-conf.yaml	Logging level	INFO	Рутовый уровень логирования

Файл log4j.properties

Ниже представлен файл *log4j.properties*, используемый Flink по умолчанию. Основные свойства подсвечены комментариями с информацией о назначении конкретного параметра.

```
################################################################################
{% include 'apache_license.txt' %}
################################################################################

# Allows this configuration to be modified at runtime. The file will be checked every 30 seconds.
monitorInterval=30 ①

# This affects logging for both user code and Flink
rootLogger.level = {{ services.flink.config.flink_conf.logging_level }} ②
rootLogger.appenderRef.file.ref = MainAppender

# Uncomment this if you want to _only_ change Flink's logging
#logger.flink.name = org.apache.flink
#logger.flink.level = INFO

# The following lines keep the log level of common libraries/connectors on
# log level INFO. The root logger does not override this. You have to manually
# change the log levels here.
logger.akka.name = akka
logger.akka.level = INFO
logger.hadoop.name = org.apache.hadoop
logger.hadoop.level = INFO
logger.zookeeper.name = org.apache.zookeeper
logger.zookeeper.level = INFO
logger.shaded_zookeeper.name = org.apache.flink.shaded.zookeeper3
logger.shaded_zookeeper.level = INFO
{% raw %}
# Log all infos in the given file
appender.main.name = MainAppender
appender.main.type = RollingFile
appender.main.append = true
appender.main.fileName = ${sys:log.file} ③
appender.main.filePattern = ${sys:log.file}.%i ④
appender.main.layout.type = PatternLayout
appender.main.layout.pattern = %d{yyyy-MM-dd HH:mm:ss,SSS} %-5p %-60c %x - %m%n ⑤
appender.main.policies.type = Policies
appender.main.policies.size.type = SizeBasedTriggeringPolicy ⑥
appender.main.policies.size.size = 100MB
appender.main.policies.startup.type = OnStartupTriggeringPolicy ⑦
appender.main.strategy.type = DefaultRolloverStrategy
appender.main.strategy.max = ${env:MAX_LOG_FILE_NUMBER:-10} ⑧

# Suppress the irrelevant (wrong) warnings from the Netty channel handler
logger.netty.name = org.jboss.netty.channel.DefaultChannelPipeline
logger.netty.level = OFF
{% endraw %}
```

① Интервал в секундах для сканирования данной конфигурации Log4j на предмет изменений.

② Установка рутового уровня логирования. Возможные уровни: `ALL`, `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, `FATAL`, `OFF`. Дефолтное значение берется из свойства ***flink-conf.yaml*** → ***Logging level*** в ADCM.

③ Установка имени лог-файлов. Шаблон по умолчанию генерирует имена файлов типа *flink-flink-taskexecutor-<n>-<host_name>.log*.

④ Шаблон для имен лог-файлов после ротации.

⑤ Шаблон для записи объекта события в лог-файл.

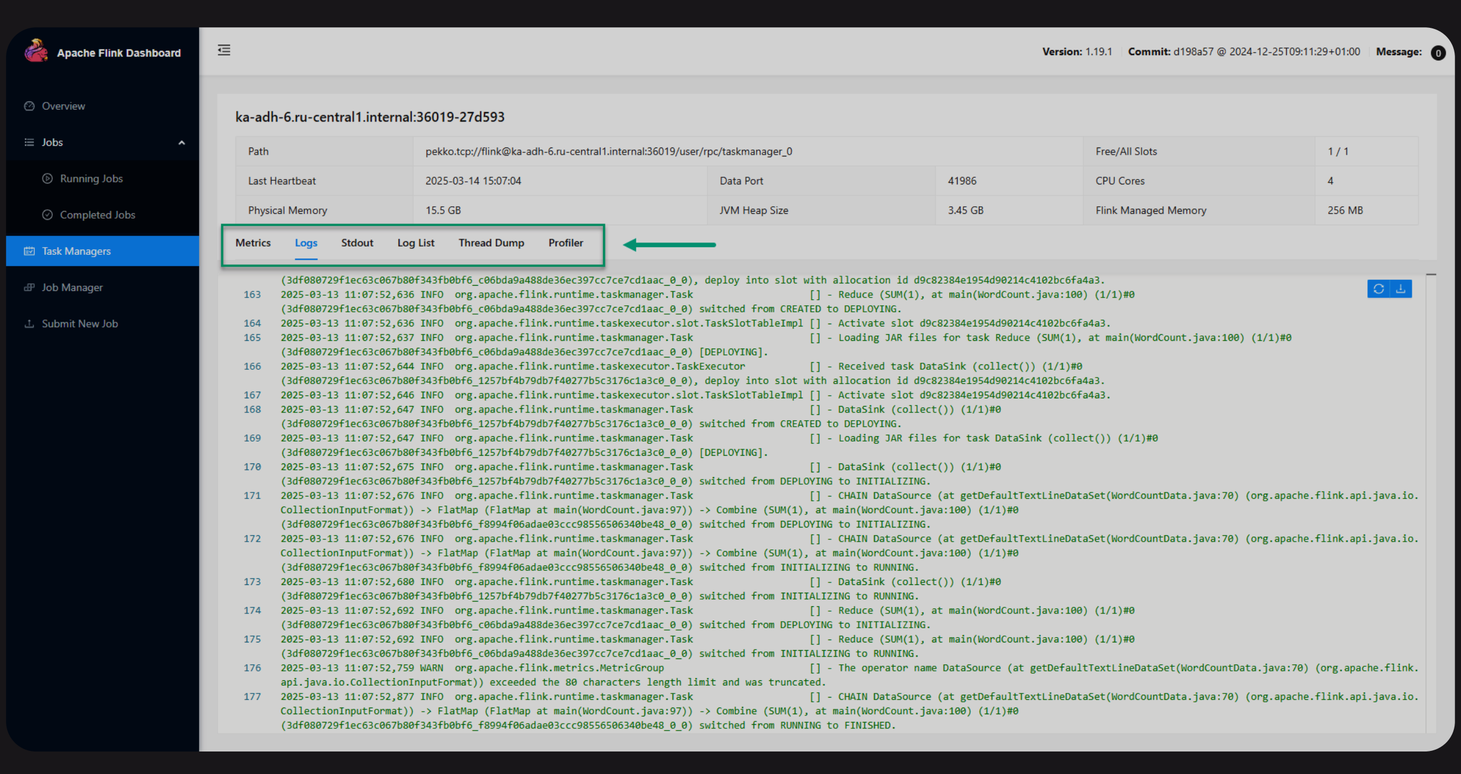
⑥ Политика ротации лог-файлов по размеру. Когда файл достигает размера, указанного в `appender.main.policies.size.size`, аппендер прекращает запись в этот файл и начинает записывать все события в новый.

⑦ Политика, выполняющая ротацию лог-файлов при старте Flink (если лог-файл старше времени запуска JVM).

⑧ Установка максимального количества лог-файлов. При превышении этого значения самые старые файлы удаляются.

Просмотр логов в Flink UI

Логи Flink, генерируемые компонентами TaskManager/JobManager, также доступны в Flink UI. Чтобы увидеть логи TaskManager, в Flink UI откройте страницу ***Task Managers*** и выберите нужный Task Manager. Текстовые логи, системный вывод, а также другие метаданные доступны на вкладках, как показано на рисунке ниже.



Version: 1.19.1 Commit: d198a57 @ 2024-12-25T09:11:29+01:00 Message: 0

ka-adh-6.ru-central1.internal:36019-27d593

Path	pekko.tcp://flink@ka-adh-6.ru-central1.internal:36019/user/rpc/taskmanager_0			Free/All Slots	1 / 1
Last Heartbeat	2025-03-14 15:07:04	Data Port	41986	CPU Cores	4
Physical Memory	15.5 GB	JVM Heap Size	3.45 GB	Flink Managed Memory	256 MB

```
(3df080729f1ec63c067b80f343fb0bf6_c06bda9a488de36ec397cc7ce7cd1aac_0_0), deploy into slot with allocation id d9c82384e1954d90214c4102bc6fa4a3.
2025-03-13 11:07:52,636 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - Reduce (SUM(1), at main(WordCount.java:100) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_c06bda9a488de36ec397cc7ce7cd1aac_0_0) switched from CREATED to DEPLOYING.
2025-03-13 11:07:52,636 INFO  org.apache.flink.runtime.taskexecutor.slot.TaskSlotTableImpl [] - Activate slot d9c82384e1954d90214c4102bc6fa4a3.
2025-03-13 11:07:52,637 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - Loading JAR files for task Reduce (SUM(1), at main(WordCount.java:100) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_c06bda9a488de36ec397cc7ce7cd1aac_0_0) [DEPLOYING].
2025-03-13 11:07:52,644 INFO  org.apache.flink.runtime.taskexecutor.TaskExecutor           [] - Received task DataSink (collect()) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_1257bf4b79db7f40277b5c3176c1a3c0_0_0), deploy into slot with allocation id d9c82384e1954d90214c4102bc6fa4a3.
2025-03-13 11:07:52,646 INFO  org.apache.flink.runtime.taskexecutor.slot.TaskSlotTableImpl [] - Activate slot d9c82384e1954d90214c4102bc6fa4a3.
2025-03-13 11:07:52,647 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - DataSink (collect()) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_1257bf4b79db7f40277b5c3176c1a3c0_0_0) switched from CREATED to DEPLOYING.
2025-03-13 11:07:52,647 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - Loading JAR files for task DataSink (collect()) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_1257bf4b79db7f40277b5c3176c1a3c0_0_0) [DEPLOYING].
2025-03-13 11:07:52,675 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - DataSink (collect()) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_1257bf4b79db7f40277b5c3176c1a3c0_0_0) switched from DEPLOYING to INITIALIZING.
2025-03-13 11:07:52,676 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - CHAIN DataSource (at getDefaultTextLineDataSet(WordCountData.java:70) (org.apache.flink.api.java.io.
CollectionInputFormat)) -> FlatMap (FlatMap at main(WordCount.java:97)) -> Combine (SUM(1), at main(WordCount.java:100) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_f8994f06adae03ccc98556506340be48_0_0) switched from DEPLOYING to INITIALIZING.
2025-03-13 11:07:52,676 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - CHAIN DataSource (at getDefaultTextLineDataSet(WordCountData.java:70) (org.apache.flink.api.java.io.
CollectionInputFormat)) -> FlatMap (FlatMap at main(WordCount.java:97)) -> Combine (SUM(1), at main(WordCount.java:100) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_f8994f06adae03ccc98556506340be48_0_0) switched from INITIALIZING to RUNNING.
2025-03-13 11:07:52,680 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - DataSink (collect()) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_1257bf4b79db7f40277b5c3176c1a3c0_0_0) switched from INITIALIZING to RUNNING.
2025-03-13 11:07:52,692 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - Reduce (SUM(1), at main(WordCount.java:100) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_c06bda9a488de36ec397cc7ce7cd1aac_0_0) switched from DEPLOYING to INITIALIZING.
2025-03-13 11:07:52,692 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - Reduce (SUM(1), at main(WordCount.java:100) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_c06bda9a488de36ec397cc7ce7cd1aac_0_0) switched from INITIALIZING to RUNNING.
2025-03-13 11:07:52,759 WARN  org.apache.flink.metrics.MetricGroup                         [] - The operator name DataSource (at getDefaultTextLineDataSet(WordCountData.java:70) (org.apache.flink.
api.java.io.CollectionInputFormat)) exceeded the 80 characters length limit and was truncated.
2025-03-13 11:07:52,877 INFO  org.apache.flink.runtime.taskmanager.Task                    [] - CHAIN DataSource (at getDefaultTextLineDataSet(WordCountData.java:70) (org.apache.flink.api.java.io.
CollectionInputFormat)) -> FlatMap (FlatMap at main(WordCount.java:97)) -> Combine (SUM(1), at main(WordCount.java:100) (1/1)#0
(3df080729f1ec63c067b80f343fb0bf6_f8994f06adae03ccc98556506340be48_0_0) switched from RUNNING to FINISHED.
```

Логи TaskManager

Логи Job Manager доступны на странице ***Job Manager*** на вкладке ***Logs***.

Логирование в приложениях Flink

В следующем примере показано базовое использование логера SLF4J в Java-приложении Flink.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class MyFlinkApp {
        private static final Logger logger = LoggerFactory.getLogger(MyFlinkApp.class); ①
        public static void main(String[] args) {
        logger.info("Log INFO event"); ②
        try {
            logger.debug("In-try with placeholder: {}", "foobar"); ③
        } catch (Exception e) {
            logger.error("An {} has occurred.", "error", e);
        }
        }
}
```

① Получение объекта логера. Объект логера рекомендуется хранить в поле с модификаторами `private static final`.

② Сохранение произвольного сообщения в лог.

③ Логирование сообщения с плейсхолдером.

Для Flink-приложений на Python пример логирования выглядит следующим образом:

```
import logging

logging.basicConfig(stream=sys.stdout,
                    level=logging.INFO,
                    format="%(message)s")
logging.info("Log this INFO event")
```

Создание стримингового ETL-пайплайна с помощью Flink

Константин Алпашкин

Содержание

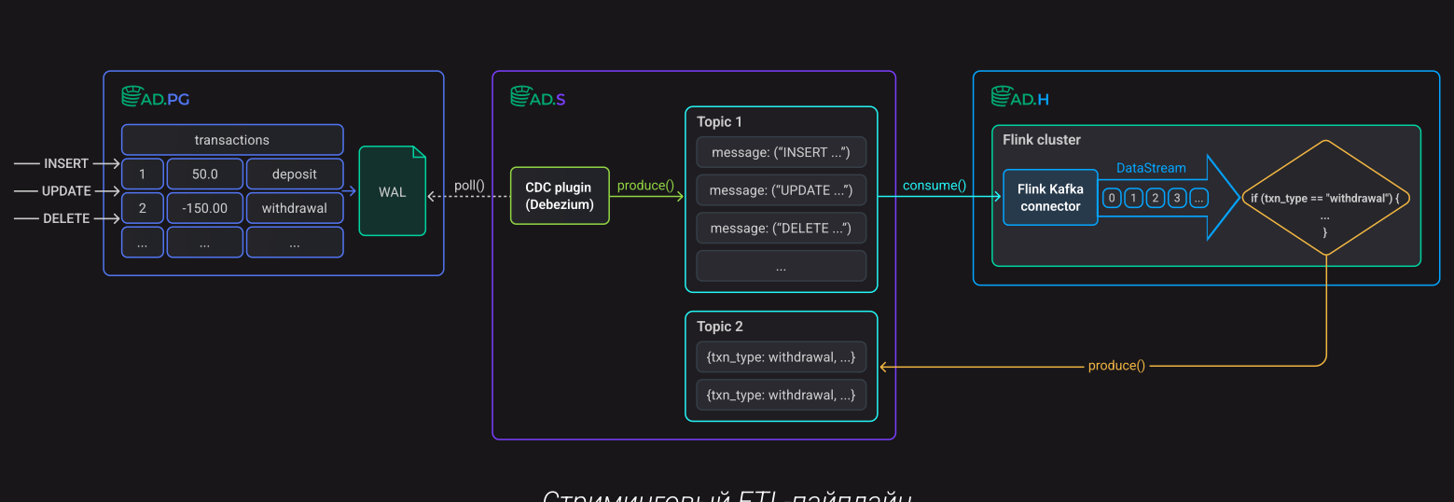
Назад к содержанию

- Обзор
- Подготовка продуктов Arenadata
  - ADPG
  - ADS и ADS Control
  - ADH
- Создание приложения Flink
- Запуск пайплайна

Обзор

В данной статье описан процесс создания стримингового ETL-пайплайна с использованием сервиса Flink (ADH) и других продуктов Arenadata. Ядром демонстрационного пайплайна служит приложение Flink, которое анализирует поток данных (Kafka) в режиме реального времени, извлекает определенные сообщения и направляет их в другой поток при соблюдении условий.

На следующей схеме представлена визуализация пайплайна.



Стриминговый ETL-пайплайн

Основные этапы обработки описаны ниже:

1. В PostgreSQL-сервере (ADPG) хранится таблица `transactions`, которая имеет следующую структуру:

```
 txn_id | acc_id |   txn_value  |      txn_date       |  txn_type
--------+--------+-----------+---------------------+------------
      1 |   1001 |       130.00 | 2025-07-01 10:15:00 | spb
      2 |   1002 |       -50.00 | 2025-07-02 09:45:00 | withdrawal
      3 |   1002 |       -50.25 | 2025-07-01 11:00:00 | withdrawal
     ...|    ... |          ... |                 ... |       ...
```

   Содержимое таблицы постоянно обновляется командами `INSERT`/`UPDATE`/`DELETE` от внешних клиентов.

2. В кластере ADS развернут сервер Kafka. В кластере также установлен CDC-плагин (Debezium), который отслеживает изменения в таблице `transactions`. Плагин сканирует write-ahead log (WAL) ADPG на предмет новых записей и отправляет информацию об изменениях в очередь Kafka в виде отдельных сообщений.

3. В кластере ADH приложение Flink подписывается на Kafka-топик в ADS, используя Flink-коннектор для Kafka. Flink парсит тело каждого сообщения и, если соблюдается определенное условие (`txn_type=withdrawal`), направляет сообщения в специальный топик Kafka.

> **ПРИМЕЧАНИЕ**
>
> В пайплайне, описанном в данной статье, Flink выполняет простую проверку на наличие определенной строки в теле сообщения. В реальных пайплайнах Flink может использоваться для более глубокого анализа потока данных, позволяя выявлять неочевидные закономерности с помощью stateful-вычислений.

Описанный в данной статье пайплайн построен с использованием следующего программного стека Arenadata.

Продукт	Версия
Arenadata Postgres (ADPG)	16.3.4
Arenadata Streaming (ADS)	3.7.2.1
Arenadata Streaming Control (ADS Control)	2.5.0
Arenadata Hyperwave (ADH)	4.0.0

Подготовка продуктов Arenadata

Далее показано, как с нуля установить и настроить продукты Arenadata, задействованные в тестовом пайплайне:

- ADPG
- ADS и ADS Control
- ADH

ADPG

1. Установите ADPG-кластер, следуя инструкциям в статье Online-установка.

2. Убедитесь, что ADPG-сервис использует порт по умолчанию (`5432`) для входящих соединений.

3. Создайте роль `demo_user`. Для этого запустите `psql` на хосте с сервисом ADPG:

```
$ sudo su - postgres
$ psql
```

   И затем выполните команду:

```
CREATE ROLE demo_user WITH LOGIN SUPERUSER PASSWORD 'password';
```

4. В установленном ADPG-кластере создайте базу данных `demo_db`. Для этого выполните команду `psql`:

```
CREATE DATABASE demo_db OWNER demo_user;
```

5. В базе данных `demo_db` создайте таблицу `transactions` в схеме `public`:

```
\c demo_db
```

```
CREATE TABLE transactions (
    txn_id     BIGINT,
    acc_id     INTEGER,
    txn_value  DOUBLE PRECISION,
    txn_date   TIMESTAMP,
    txn_type   TEXT
);
```

6. Отредактируйте конфигурацию pg_hba.conf, чтобы разрешить доступ к ADPG. Для этого добавьте следующую строку в поле ***PG_HBA*** на странице настроек сервиса ADPG:

```
host demo_db demo_user all trust
```

> **ПРИМЕЧАНИЕ**
>
> Данная запись разрешает TCP/IP-соединения к серверу ADPG с любого IP-адреса, что небезопасно в производственной среде. Для лучшей безопасности явно укажите IP-адреса хостов, которые будут подключаться к серверу ADPG.

7. Установите конфигурационный параметр `wal_level=logical`. Это добавляет в лог информацию для логического декодирования, что необходимо для работы Debezium. Чтобы установить параметр, укажите его в поле ***postgresql.conf custom section*** на странице настроек ADPG-сервиса.

Права для Debezium

Чтобы Debezium мог корректно взаимодействовать с ADPG, убедитесь, что роль PostgreSQL, используемая Debezium, имеет следующие права:

- `REPLICATION`
- `LOGIN`
- `CREATE`
- `SELECT`

В этой статье пользователь с правами `SUPERUSER` обладает всеми необходимыми привилегиями, но такую роль следует использовать только для тестирования. Для большей безопасности рекомендуется предоставлять пользователям только необходимые привилегии.

Подробная информация о настройке привилегий пользователей для Debezium доступна в документации Debezium.

ADS и ADS Control

1. Установите кластер ADS, следуя инструкциям в статье Online-установка. Кластер должен включать сервисы ZooKeeper, Kafka и Kafka Connect.

2. Активируйте конфигурационный параметр Kafka ***auto.create.topics.enable*** в секции ***server.properties***. Это необходимо, чтобы Debezium мог автоматически создавать топики Kafka.

3. Установите кластер ADS Control в соответствии с инструкциями.

4. Интегрируйте ADS Control с кластером ADS в соответствии с инструкциями.

5. В веб-интерфейсе ADS Control создайте коннектор Debezium для ADPG согласно инструкциям. Коннектор Debezium, используемый в этом сценарии, имеет следующую JSON-конфигурацию:

```
{
  "connector.class": "io.debezium.connector.postgresql.PostgresConnector",
  "publication.autocreate.mode": "filtered",
  "database.user": "demo_user", ①
  "database.dbname": "demo_db", ②
  "tasks.max": 1,
  "database.port": 5432,
  "plugin.name": "pgoutput",
  "topic.prefix": "postgres",
  "database.hostname": "10.92.43.174", ③
  "database.password": "password", ④
  "name": "PostgresConnector",
  "value.converter": "org.apache.kafka.connect.storage.StringConverter",
  "key.converter": "org.apache.kafka.connect.storage.StringConverter"
}
```

   ① Роль PostgreSQL для подключения Debezium к ADPG.

   ② База данных ADPG для отслеживания изменений.

   ③ Адрес сервера ADPG. Используйте IP-адрес вашего ADPG-сервера.

   ④ Пароль для доступа к ADPG.

В результате Debezium считывает записи WAL-файла ADPG и направляет сообщения в Kafka-топик `postgres.public.transactions`. Пример сообщения:

```
Struct{after=Struct{txn_id=1,acc_id=1001,txn_value=250.0,txn_date=1751364900000000,txn_type=deposit},source=Struct{version=2.7.0.
Final,connector=postgresql,name=postgres,ts_ms=1752485024943,snapshot=first,db=demo_db,sequence=
[null,"29241720"],ts_us=1752485024943896,ts_ns=1752485024943896000,schema=public,table=transactions,txId=758,lsn=29241720},op=r,t
s_ms=1752485025133,ts_us=1752485025133518,ts_ns=1752485025133518357}
```

ADH

Установите кластер ADH, следуя инструкциям в статье Online-установка. Кластер ADH должен включать следующие сервисы:

- Core configuration
- HDFS
- YARN
- ZooKeeper
- Flink

Создание приложения Flink

Ниже описаны шаги по созданию Flink-приложения на Java с помощью Maven.

> **ПРИМЕЧАНИЕ**
>
> Подробная информация о создании приложений Flink на Python доступна в статье Примеры использования PyFlink.

Установите Maven (если он еще не установлен):

1. Скачайте Maven:

```
$ wget https://dlcdn.apache.org/maven/maven-3/3.9.10/binaries/apache-maven-3.9.10-bin.tar.gz -P /tmp
```

2. Распакуйте архив:

```
$ sudo tar -zxf /tmp/apache-maven-3.9.10-bin.tar.gz -C /opt
```

3. Проверьте версию Maven:

```
$ /opt/apache-maven-3.9.10/bin/mvn -version
```

   Пример вывода:

```
Apache Maven 3.9.10 (5f519b97e944483d878815739f519b2eade0a91d)
Maven home: /opt/apache-maven-3.9.10
Java version: 1.8.0_412, vendor: Red Hat, Inc., runtime: /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.412.b08-1.el7_9.x86_64/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "3.10.0-1160.59.1.el7.x86_64", arch: "amd64", family: "unix"
```

Установив Maven, выполните следующие шаги для сборки проекта.

1. Создайте новый Maven-проект со следующей структурой:

```
flink-app-demo
│   pom.xml
└───src
    └───main
        └───java
            └───io
                └───arenadata
                       └─KafkaFlinkApp.java
```

   Пример тестового *pom.xml* представлен ниже.

   pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>io.arenadata</groupId>
  <artifactId>flink-app-demo</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>
  <name>Flink-app-demo</name>

  <properties>
    <flink.version>1.20.1</flink.version>
    <kafka.connector.version>3.3.0-1.20</kafka.connector.version>
    <java.version>1.8</java.version>
    <maven.compiler.source>8</maven.compiler.source>
    <maven.compiler.target>8</maven.compiler.target>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-streaming-java</artifactId>
      <version>${flink.version}</version>
    </dependency>
    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-connector-base</artifactId>
      <version>${flink.version}</version>
    </dependency>
    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-connector-kafka</artifactId>
      <version>${kafka.connector.version}</version>
    </dependency>
    <dependency>
      <groupId>com.fasterxml.jackson.core</groupId>
      <artifactId>jackson-databind</artifactId>
      <version>2.17.1</version>
    </dependency>
  </dependencies>
</project>
```

> **ПРИМЕЧАНИЕ**
>
> В данном примере показано, как собрать и запустить приложение Flink в кластере ADH, который уже содержит базовые библиотеки, необходимые для работы Flink-приложений. В зависимости от версии Flink, запуск проекта в IDE (вне кластера ADH) может привести к исключениям `java.lang.NoClassDefFoundError` из-за отсутствия базовых зависимостей. Для тестирования Flink-приложения в IDE вне кластера ADH добавьте библиотеки Flink в *pom.xml*. Пример такого "толстого" *pom.xml* приведен ниже.

   pom.xml для запуска в IDE

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>io.arenadata</groupId>
  <artifactId>flink-app-demo</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>
  <name>Flink-app-demo</name>

  <properties>
    <flink.version>1.20.1</flink.version>
    <kafka.connector.version>3.3.0-1.20</kafka.connector.version>
    <java.version>1.8</java.version>
    <maven.compiler.source>8</maven.compiler.source>
    <maven.compiler.target>8</maven.compiler.target>
  </properties>


    <!-- This dependency is provided, because it should not be packaged into the JAR file. -->
  <dependencies>
    <!-- Flink Streaming API -->
    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-streaming-java</artifactId>
      <version>${flink.version}</version>
    </dependency>

    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-connector-base</artifactId>
      <version>${flink.version}</version>
    </dependency>
    <!-- Flink Kafka Connector -->
    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-connector-kafka</artifactId>
      <version>${kafka.connector.version}</version>
    </dependency>

    <!-- Kafka client (optional, provided by connector) -->
    <dependency>
      <groupId>org.apache.kafka</groupId>
      <artifactId>kafka-clients</artifactId>
      <version>3.7.2</version>
    </dependency>

    <!-- Flink Core (optional) -->
    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-core</artifactId>
      <version>${flink.version}</version>
    </dependency>

    <!-- Logging (required for output visibility) -->
    <dependency>
      <groupId>org.slf4j</groupId>
      <artifactId>slf4j-log4j12</artifactId>
      <version>1.7.36</version>
    </dependency>

    <dependency>
      <groupId>com.fasterxml.jackson.core</groupId>
      <artifactId>jackson-databind</artifactId>
      <version>2.17.1</version>
    </dependency>

    <!-- To make it work in IntelliJ IDEA: -->

    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-runtime</artifactId>
      <version>${flink.version}</version>
    </dependency>

    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-clients</artifactId>
      <version>${flink.version}</version>
    </dependency>

    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-runtime-web</artifactId>
      <version>${flink.version}</version>
      <optional>true</optional>
    </dependency>

    <dependency>
      <groupId>org.apache.flink</groupId>
      <artifactId>flink-test-utils</artifactId>
      <version>${flink.version}</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

2. Вставьте следующий код в *KafkaFlinkApp.java*:

```
package io.arenadata;

import org.apache.flink.api.common.eventtime.WatermarkStrategy;
import org.apache.flink.api.common.serialization.SimpleStringSchema;
import org.apache.flink.connector.kafka.sink.KafkaSink;
import org.apache.flink.connector.kafka.sink.KafkaRecordSerializationSchema;
import org.apache.flink.connector.kafka.source.KafkaSource;
import org.apache.flink.connector.kafka.source.enumerator.initializer.OffsetsInitializer;
import org.apache.flink.streaming.api.datastream.DataStream;
import org.apache.flink.streaming.api.environment.StreamExecutionEnvironment;

public class KafkaFlinkApp {

    public static void main(String[] args) throws Exception {
        final StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

        KafkaSource<String> src_kafka_transactions = KafkaSource.<String>builder() ①
                .setBootstrapServers("ka-ads-1.ru-central1.internal:9092,ka-ads-2.ru-central1.internal:9092,ka-ads-3.ru-
central1.internal:9092") ②
                .setTopics("postgres.public.transactions") ③
                .setGroupId("flink-consumer-group")
                .setStartingOffsets(OffsetsInitializer.earliest())
                .setValueOnlyDeserializer(new SimpleStringSchema())
                .build();

        DataStream<String> stream = env.fromSource(src_kafka_transactions, ④
                WatermarkStrategy.noWatermarks(),
                "Kafka Source");


        DataStream<String> filtered = stream.filter(msg -> { ⑤
            boolean isWithdrawal = msg.contains("txn_type=withdrawal");
            if (isWithdrawal) {
                System.out.println("Withdrawal detected in message: " + msg);
            }
            return isWithdrawal;
        });

        KafkaSink<String> sink_kafka_withdrawals = KafkaSink.<String>builder() ⑥
                .setBootstrapServers("ka-ads-1.ru-central1.internal:9092,ka-ads-2.ru-central1.internal:9092,ka-ads-3.ru-
central1.internal:9092")
                .setRecordSerializer(
                        KafkaRecordSerializationSchema.<String>builder()
                                .setTopic("postgres.public.transactions.withdrawals")
                                .setValueSerializationSchema(new SimpleStringSchema())
                                .build())
                .build();

        filtered.sinkTo(sink_kafka_withdrawals); ⑦

        env.execute("My Flink demo app"); ⑧
    }
}
```

   ① Объявление KafkaSource в качестве источника (source) входных данных.

   ② Список адресов брокеров Kafka для подключения к кластеру ADS. Актуальная строка подключения доступна в ADCM (***Clusters → <ADS_cluster> → Services → Kafka → Info***).

   ③ Топик Kafka, из которого приложение будет получать сообщения об изменениях в таблице. Debezium автоматически создает данный топик и направляет в него CDC-сообщения об изменениях в таблице ADPG.

   ④ Создание объекта `DataStream` на основе источника Kafka.

   ⑤ Создание еще одного DataStream, в который попадают только сообщения, содержащие `txn_type=withdrawal`. В реальных пайплайнах можно выполнять более глубокий анализ элементов DataStream с помощью продвинутых инструментов Flink API (например, обнаружение последовательности транзакций, совершенных одним аккаунтом за определенный отрезок времени).

   ⑥ Объявление KafkaSink для отправки данных в Kafka.

   ⑦ Отправка содержимого DataStream в ADS.

   ⑧ Запуск приложения.

3. Скомпилируйте проект:

```
$ cd flink-app-demo
$ /opt/apache-maven-3.9.10/bin/mvn clean package
```

Запуск пайплайна

1. Перезапустите кластеры ADPG, ADS, ADS Control и ADH.

2. На любом ADH-хосте, где установлен компонент Flink client, запустите приложение Flink:

```
$ flink run flink-app-demo/target/flink-app-demo-1.0-SNAPSHOT.jar
```

   После этого JAR-файл приложения передается компоненту JobManager, и далее приложение выполняется на хостах TaskManager. Пример вывода:

```
Setting HBASE_CONF_DIR=/etc/hbase/conf because no HBASE_CONF_DIR was set.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/flink/lib/log4j-slf4j-impl-2.17.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Job has been submitted with JobID c917cc83500f4866e3aeb67d1d17dbf7
```

3. Убедитесь, что приложение успешно запустилось, используя команду `flink list` или веб-интерфейс Flink.

   Пример вывода:

```
[konstantin@ka-adh-2 ~]$ flink list
------------------ Running/Restarting Jobs -------------------
14.07.2025 12:36:13 : 57ff8e66a5a76cf80da5b760b44ac0ad : My Flink demo app (RUNNING)
--------------------------------------------------------------
No scheduled jobs.
```

4. Запишите тестовые данные в таблицу ADPG. Для этого запустите `psql` на хосте ADPG-сервера:

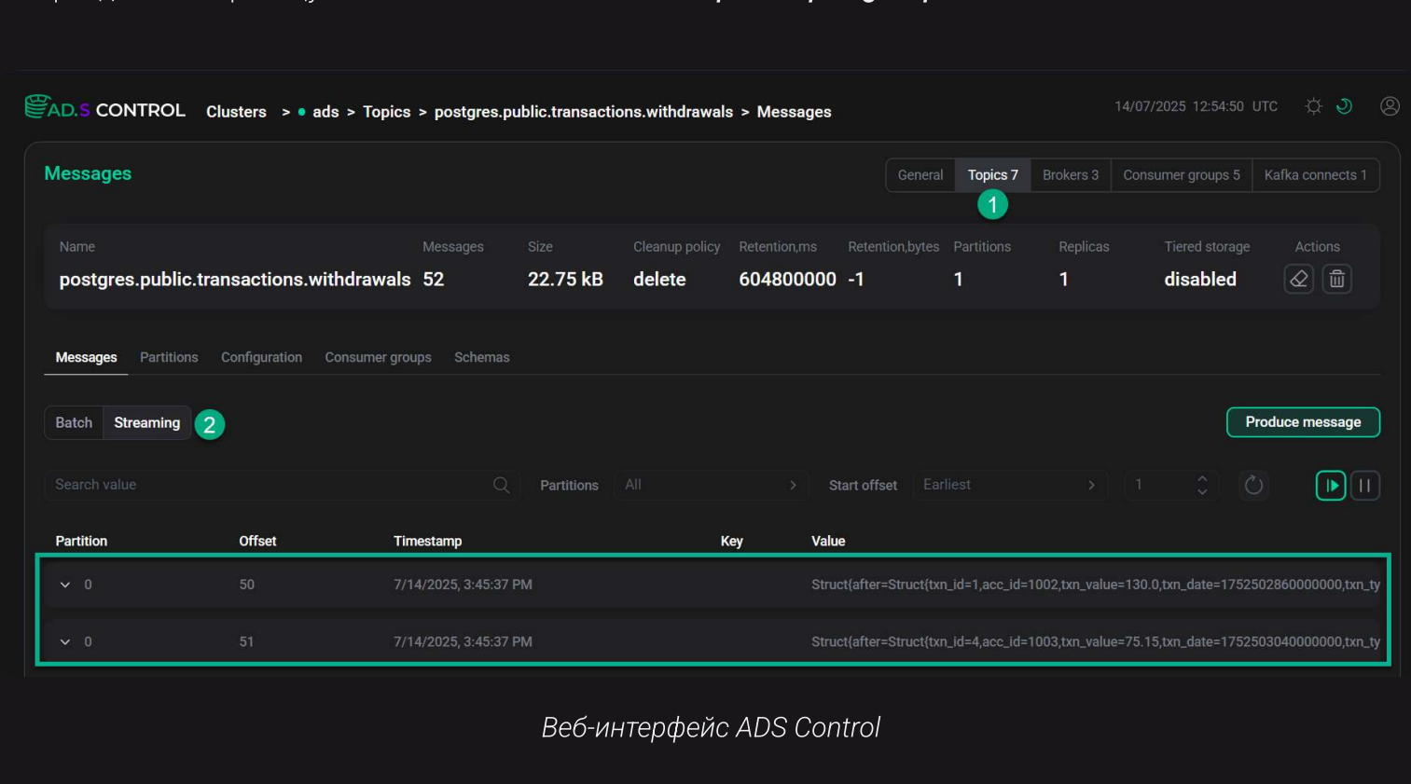
```
$ sudo su - postgres
$ psql
```

   Подключитесь к тестовой БД и выполните операцию `INSERT`:

```
\c demo_db
```

```
INSERT INTO public.transactions VALUES
(1, 1002,  130.00, '2025-07-14 14:21:00', 'withdrawal');
(2, 1001,  50.00, '2025-07-14 14:22:00', 'deposit');
(3, 1002,  50.25, '2025-07-14 14:23:00', 'spb');
(4, 1003,  75.15, '2025-07-14 14:24:00', 'withdrawal');
```

5. В веб-интерфейсе ADS Control перейдите на страницу ***Clusters → <ADS_cluster> → Topics → postgres.public.transactions.withdrawals → Messages***, как показано ниже.



Name	Messages	Size	Cleanup policy	Retention,ms	Retention,bytes	Partitions	Replicas	Tiered storage	Actions
postgres.public.transactions.withdrawals	52	22.75 kB	delete	604800000	-1	1	1	disabled	

Partition	Offset	Timestamp	Key	Value
0	50	7/14/2025, 3:45:37 PM		Struct{after=Struct{txn_id=1,acc_id=1002,txn_value=130.0,txn_date=1752502860000000,txn_ty
0	51	7/14/2025, 3:45:37 PM		Struct{after=Struct{txn_id=4,acc_id=1003,txn_value=75.15,txn_date=1752503040000000,txn_ty

Веб-интерфейс ADS Control

   После небольшой задержки в топике `postgres.public.transactions.withdrawals` появятся 2 сообщения, отфильтрованные Flink по наличию `txn_type=withdrawal`. Проверьте тело сообщений, развернув их в веб-интерфейсе.

6. Остановите приложение Flink вручную. Для этого выполните следующие команды на любом ADH-хосте с компонентом Flink client.

   Получите ID Flink-приложения:

```
$ flink list
```

   Остановите приложение:

```
$ flink cancel <application_id>
```

   Пример вывода:

```
Cancelling job c917cc83500f4866e3aeb67d1d17dbf7.
Cancelled job c917cc83500f4866e3aeb67d1d17dbf7.
```

Управление сервисом Flink через ADCM

Константин Алпашкин

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Flink. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Flink, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Flink
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются

Действия с компонентами Flink

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Управление сервисом Flink2 через ADCM

Константин Алпашкин

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Flink2. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Flink2, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Flink2
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются

Действия с компонентами Flink2

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Подключение к HBase

Евгения Кузина

Содержание

Назад к содержанию

HBase shell

Нативное Java API

Внешнее API

Phoenix

В статье описаны способы подключения к HBase. Вы можете использовать HBase shell, Phoenix, собственное и внешнее API HBase. Также для взаимодействия с HBase предусмотрен веб-интерфейс. См. Обзор HBase UI.

HBase shell

HBase shell представляет собой консоль JRuby, доступную на любом узле кластера HBase. За дополнительной информацией обратитесь к статьям Начало работы с HBase shell и Команды HBase shell.

Нативное Java API

HBase имеет Java API, позволяющее осуществлять программный доступ к языку манипулирования данными (DML). С помощью него можно выполнять CRUD-операции над таблицами HBase. Статья Использование нативного Java API для подключения к HBase содержит пример использования нескольких методов. Полный список API доступен по ссылке: User API Reference.

Внешнее API

Можно получить доступ к Apache HBase через языки программирования, отличные от Java, и через специальные протоколы. ADH включает в себя HBase REST-сервер и HBase Thrift2-сервер, которые можно добавить в кластер в качестве компонентов сервиса HBase. Для получения дополнительной информации обратитесь к статье Использование внешнего API для подключения HBase.

Phoenix

Apache Phoenix — это система запросов SQL для Apache HBase. Доступ к системе осуществляется через драйвер JDBC. Phoenix позволяет создавать запросы и управлять таблицами HBase с помощью SQL. См. Подключение к Phoenix.

Использование нативного Java API для подключения к HBase

Eugenia Kuzina

Назад к содержанию

HBase Java API позволяет использовать язык манипулирования данными (DML), с помощью которого можно выполнять CRUD-операции над таблицами HBase. Эта статья содержит пример использования нескольких методов. Полный список API доступен по ссылке: User API Reference.

Для компиляции приведенного ниже примера можно использовать Maven. Если вы не знакомы с этим инструментом, следующее руководство будет полезно: Maven in 5 Minutes.

В примере показано, как создать, изменить и удалить таблицу HBase в версии ADH 3.1.2.1.

```java
package com.example.hbase.admin;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.HColumnDescriptor;
import org.apache.hadoop.hbase.HConstants;
import org.apache.hadoop.hbase.HTableDescriptor;
import org.apache.hadoop.hbase.TableName;
import org.apache.hadoop.hbase.client.Admin;
import org.apache.hadoop.hbase.client.Connection;
import org.apache.hadoop.hbase.client.ConnectionFactory;
import org.apache.hadoop.hbase.io.compress.Compression.Algorithm;

public class Example {

  private static final String TABLE_NAME = "MY_TABLE";
  private static final String CF_DEFAULT = "MY_COLUMN_FAMILY";


  public static void createOrOverwrite(Admin admin, HTableDescriptor table) throws IOException {
    if (admin.tableExists(table.getTableName())) {
      admin.disableTable(table.getTableName());
      admin.deleteTable(table.getTableName());
    }
    admin.createTable(table);
  }

  // Создание таблицы
  public static void createSchemaTables(Configuration config) throws IOException {
    try (Connection connection = ConnectionFactory.createConnection(config);
         Admin admin = connection.getAdmin()) {

      HTableDescriptor table = new HTableDescriptor(TableName.valueOf(TABLE_NAME));
      table.addFamily(new HColumnDescriptor(CF_DEFAULT).setCompressionType(Algorithm.NONE));

      System.out.print("Creating table.");
      createOrOverwrite(admin, table);
      System.out.println(" Done.");
    }
  }

  public static void modifySchema (Configuration config) throws IOException {
    try (Connection connection = ConnectionFactory.createConnection(config);
         Admin admin = connection.getAdmin()) {

      TableName tableName = TableName.valueOf(TABLE_NAME);
      if (!admin.tableExists(tableName)) {
        System.out.println("Table does not exist.");
        System.exit(-1);
      }

      HTableDescriptor table = admin.getTableDescriptor(tableName);

      // Обновление таблицы
      HColumnDescriptor newColumn = new HColumnDescriptor("NEW_COL_FAMILY");
      newColumn.setCompactionCompressionType(Algorithm.GZ);
      newColumn.setMaxVersions(HConstants.ALL_VERSIONS);
      admin.addColumn(tableName, newColumn);

      // Обновление существующего семейства столбцов
      HColumnDescriptor existingColumn = new HColumnDescriptor(CF_DEFAULT);
      existingColumn.setCompactionCompressionType(Algorithm.GZ);
      existingColumn.setMaxVersions(HConstants.ALL_VERSIONS);
      table.modifyFamily(existingColumn);
      admin.modifyTable(tableName, table);

      // Отключение таблицы
      admin.disableTable(tableName);

      // Удаление семейства столбцов
      admin.deleteColumn(tableName, CF_DEFAULT.getBytes("UTF-8"));

      // Удаление таблицы. Таблицу сначала необходимо отключить.
      admin.deleteTable(tableName);
    }
  }

  public static void main(String... args) throws IOException {
    Configuration config = HBaseConfiguration.create();

    //Добавление конфигурационных файлов (hbase-site.xml, core-site.xml)
    config.addResource(new Path(System.getenv("HBASE_CONF_DIR"), "hbase-site.xml"));
    config.addResource(new Path(System.getenv("HADOOP_CONF_DIR"), "core-site.xml"));
    createSchemaTables(config);
    modifySchema(config);
  }
}
```

Добавьте следующую зависимость в файл *pom.xml*:

```xml
<dependency>
    <groupId>org.apache.hbase</groupId>
    <artifactId>hbase-shaded-client</artifactId>
    <version>2.2.7</version>
</dependency>
```

Использование внешнего API для подключения к HBase

Евгения Кузина

Содержание

Назад к содержанию

HBase REST Server

HBase Thrift2 Server

Вы можете получить доступ к Apache HBase не только с помощью языка Java, но и других языков программирования и специальных протоколов. ADH 3.1.2.1 включает HBase REST Server и HBase Thrift2 Server, настройка которых описана ниже. Также можно использовать C/C++ Apache HBase Client, Java Data Objects (JDO), Scala и Jython для того, чтобы подключиться к сервису ADH HBase теми же способами, что и к нативной версии Apache HBase.

HBase REST Server

ADH включает HBase REST-сервер, который может быть добавлен в кластер во время установки. Чтобы добавить компонент ***HBase REST Server*** в уже установленный кластер, выполните кластерное действие ***Add HBase REST Server(s)***. См. Управление сервисом HBase через ADCM.

По умолчанию HBase REST-сервер работает на порте `60080`. Чтобы изменить это значение, укажите параметр конфигурации `hbase.rest.port` в пользовательском интерфейсе ADCM. См. Конфигурационные параметры.

HTTP-методы для обращения к REST-серверу можно вызывать утилитами curl или wget. Если для конечной точки не указан определенный метод, используйте `GET` для запросов на выборку, `PUT` или `POST` для создания или изменения и `DELETE` для удаления.

В этой статье содержится несколько примеров использования конечных точек. Для получения информации обо всех доступных конечных точках перейдите по ссылке Using REST Endpoints.

По умолчанию вывод команды представляет собой обычный текст. Чтобы изменить формат ответа, добавьте заголовок `Accept` следующим образом:

- для XML — `Accept: text/xml`;
- для JSON — `Accept: application/json`;
- для Protocol Buffers (протокол сериализации данных) — `Accept: application/x-protobuf`.

Следующий запрос возвращает версию кластера в формате XML:

```
$ curl -vi -X GET -H "Accept: text/xml" "http://10.92.40.168:60080/version/cluster"
```

Результат:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><ClusterVersion Version="2.2.7"/>
```

Можно использовать конечные точки для управления пространствами имен. Команда ниже создает пространство имен:

```
$ curl -vi -X POST "http://10.92.40.168:60080/namespaces/test_ns"
```

Чтобы удалить пространство имен `test_ns`, вызовите следующую команду:

```
$ curl -vi -X DELETE "http://10.92.40.168:60080/namespaces/test_ns"
```

Приведенный ниже код создает новую таблицу или заменяет схему существующей таблицы указанной схемой:

```
$ curl -vi -X POST \
-H "Accept: text/xml" \
-H "Content-Type: text/xml" \
-d '<?xml version="1.0" encoding="UTF-8"?><TableSchema name="users"><ColumnSchema name="cf" /></TableSchema>' \
"http://10.92.40.168:60080/users/schema"
```

Добавьте строку в таблицу `users`, созданную выше:

```
$ curl -vi -X PUT \
         -H "Accept: text/xml" \
         -H "Content-Type: text/xml" \
         -d '<?xml version="1.0" encoding="UTF-8" standalone="yes"?><CellSet><Row key="cm93MQ=="><Cell
column="Y2Y6ZQo=">dmFsdWUx</Cell></Row></CellSet>' \
         "http://10.92.40.168:60080/users/testrow"
```

Строка, квалификатор столбца и значение должны быть закодированы Base-64.

Код ниже удаляет таблицу `users`:

```
$ curl -vi -X DELETE -H "Accept: text/xml" "http://10.92.40.168:60080/users/schema"
```

HBase Thrift2 Server

Вы можете добавить компонент ***HBase Thrift2 server*** во время установки кластера на шаге Добавление компонентов. Также можно использовать действие ***Add Thrift2 Server(s)***, чтобы включить HBase Thrift2-сервер в уже установленном кластере. См. Управление сервисом HBase через ADCM.

Используйте параметры конфигурации, перечисленные в таблице ниже, чтобы настроить Thrift для безопасной аутентификации.

Параметр	Описание
hbase.thrift.keytab.file	Kerberos keytab-файл для Thrift
hbase.thrift.kerberos.principal	Принципал Thrift Kerberos
hbase.thrift.security.qop	Устанавливает проверку аутентификации, целостности и приватности. Поддерживаемые значения: ■ `auth-conf` — проверка аутентификации, целостности и приватности; ■ `auth-int` — проверка аутентификации и целостности; ■ `auth` — только проверка аутентификации.

HBase Thrift2-сервер работает на порте `9090`.

Обзор HBase UI

Евгения Кузина

Содержание

Назад к содержанию

- Веб-интерфейс Master Server
  - Home
  - Table Details
  - Procedures & Locks
  - HBCK Report
  - Process Metrics
  - Local Logs
  - Log Level
  - Debug Dump
  - Metrics Dump
  - Profiler
  - HBase Configuration
- Веб-интерфейс Region Server
  - Home
  - Process Metrics
  - Local Logs
  - Log Level
  - Debug Dump
  - Metrics Dump
  - Profiler
  - HBase Configuration
- Веб-интерфейс Thrift2 Server
  - Home
  - Local Logs
  - Log Level
  - Metrics Dump
  - Profiler
  - HBase Configuration

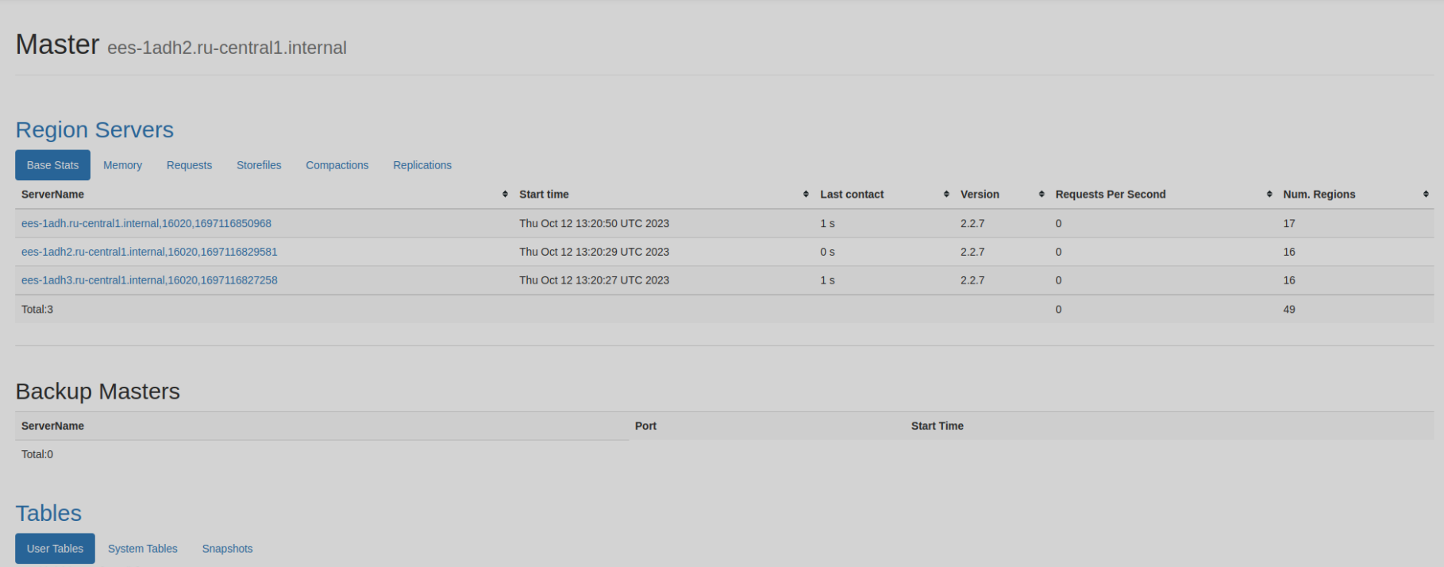
Пользовательские веб-интерфейсы HBase позволяют просматривать текущее состояние кластера и менять log level без перезагрузки сервиса. HBase поддерживает веб-интерфейсы для серверов Master, Region и Thrift2. URL веб-интерфейсов HBase перечислены в таблице ниже.

Формат адресов веб-интерфейсов

Сервер	Адрес
Master Server	http://<Master Server IP>:16010
Region Server	http://<Region Server IP>:16030
HBase Thrift2 Server	http://<HBase Thrift2 Server IP>:9095

Веб-интерфейс Master Server

Пользовательский интерфейс HBase Master Server работает на порте `16010`.



Master ees-1adh2.ru-central1.internal

Region Servers

ServerName	Start time	Last contact	Version	Requests Per Second	Num. Regions
ees-1adh.ru-central1.internal,16020,1697116850968	Thu Oct 12 13:20:50 UTC 2023	1 s	2.2.7	0	17
ees-1adh2.ru-central1.internal,16020,1697116829581	Thu Oct 12 13:20:29 UTC 2023	0 s	2.2.7	0	16
ees-1adh3.ru-central1.internal,16020,1697116827258	Thu Oct 12 13:20:27 UTC 2023	1 s	2.2.7	0	16
Total:3				0	49

Backup Masters

ServerName	Port	Start Time
Total:0		

Tables

13 table(s) in set. [Details]

Домашняя страница веб-интерфейса Master Server

Веб-интерфейс HBase Master содержит следующие вкладки:

- Home
- Table Details
- Procedures & Locks
- HBCK Report
- Process Metrics
- Local Logs
- Log Level
- Debug Dump
- Metrics Dump
- Profiler
- HBase Configuration

Home

Вкладка ***Home*** интерфейса Master Server содержит следующие разделы с общей информацией о кластере HBase:

- ***Region Servers*** — содержит общую статистику, информацию об используемой памяти, включая размер memstore, сведения о репликации и статистику storefile. В таблице метрик также отображаются ссылки на пользовательские интерфейсы соответствующих HBase Region Servers.
- ***Backup Masters*** — информация о резервных серверах Master, если такие имеются в кластере.
- ***Tables*** — пользовательские и системные таблицы и снепшоты.
- ***Peers*** — информация об удаленных кластерах, в которые реплицируются данные.
- ***Tasks*** — информация о выполненных задачах. Можно просматривать отдельно статистику по задачам обработчика RPC и задачам, не связанным с RPC. Также ссылка ***View as JSON*** позволяет получить статистику по задачам в формате JSON.
- ***Software Attributes*** — общая информация о JVM, HBase, Hadoop и ZooKeeper. Раздел включает информацию о версиях, контрольных суммах, датах компиляции, адреса зарегистрированных серверов ZK, корневой узел кластера HBase в ZK, ключ для добавления кластера в качестве узла репликации, расположение домашнего каталога HBase и другие.

Table Details

Вкладка ***Table Details*** содержит таблицу ***User Tables*** с двумя колонками:

- ***Table*** — отображает ссылки на страницы с подробной информацией о таблицах, которая включает атрибуты таблицы, схему, регионы, распределение регионов по серверам Region и статистику таблицы.
- ***Description*** — содержит описания таблиц, аналогичные выводу команды describe.

Procedures & Locks

На вкладке ***Procedures & Locks*** перечислены процедуры, завершенные с ошибками, и блокировки. Вкладка также содержит статистику состояния WAL для процедур.

HBCK Report

Вкладка ***HBCK Report*** отображает два отчёта: ***HBCK Chore Report*** и ***CatalogJanitor Consistency Issues***.

HBCK Chore Report проверяет соответствие информации о регионах в памяти HMaster, памяти RegionServer и HDFS. Он также проверяет, существует ли регион и согласована ли информация о нахождении региона.

CatalogJanitor Consistency Issues Report определяет, является ли информация о регионе в метатаблице полной, а также имеются ли проблемы с наложениями и пропусками.

Запустите команды `hbck_chore_run` и catalogjanitor_run в hbase shell, чтобы сгенерировать новые отчёты. Если проблемы не обнаружены, отображаются только заголовки отчетов.

Process Metrics

Вкладка ***Process Metrics*** отображает лист метрик для процесса HMaster, который включает следующие элементы:

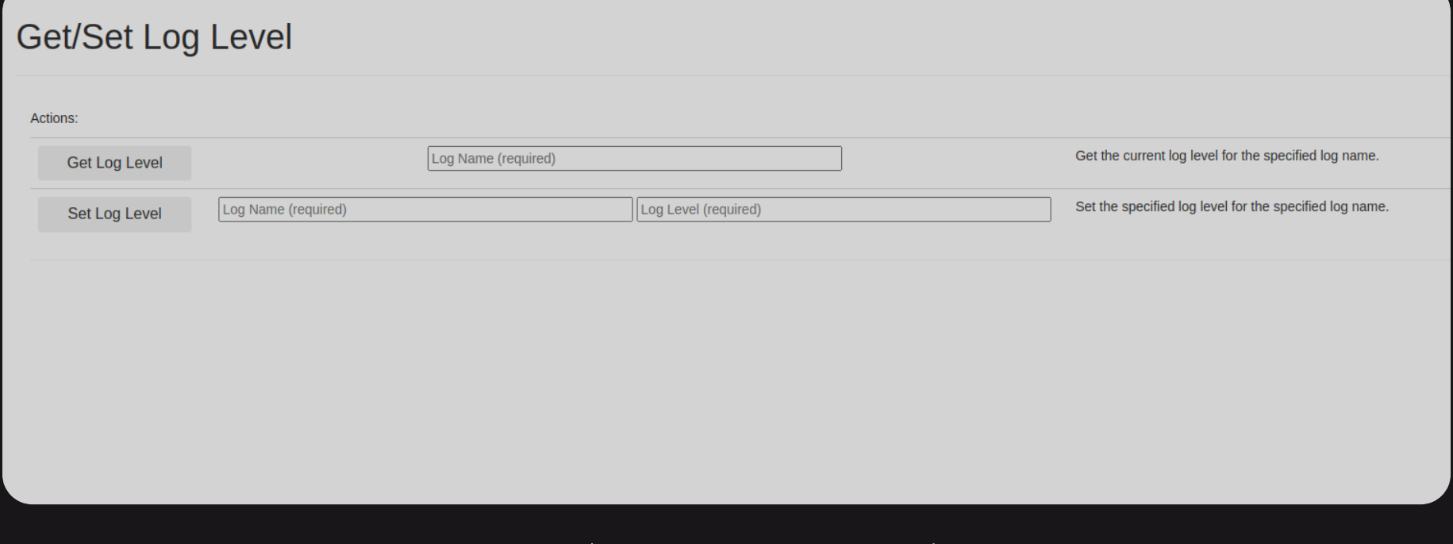
- Threads — статистика потоков процесса.
- GC Collectors — статистика по сборщикам мусора (garbage collectors) Java.
- Metaspace — статистика по `Metaspace`. Область памяти `Metaspace` содержит метаданные классов, загруженных в память, и загрузчики классов.
- Compressed Class Space — статистика по `Compressed Class Space`, области сжатых указателей, которая используется для хранения информации о загруженных классах.
- Par Eden Space — статистика по `Eden Space`, где создается большинство новых объектов Java. Когда пространство памяти, выделенное для этой области, заполняется, доступные объекты из `Eden Space` переносятся в `Survivor Space`.
- Par Survivor Space — статистика по `Survivor Space`. Когда пространство памяти, выделенное для этой области, заполняется, доступные объекты из `Survivor Space` переносятся в `CMS Old Gen`.
- CMS Old Gen — статистика по области памяти `CMS Old Gen`. Эта область Java heap используется для постоянных объектов.

Local Logs

Вкладка ***Local Logs*** содержит список ссылок на файлы журналов, хранящиеся на текущем хосте. При нажатии на ссылку соответствующий файл журнала открывается в окне браузера. В файловой системе файлы журналов хранятся в каталоге */var/log/hbase*. Для получения дополнительной информации обратитесь к статье Логирование в HBase.

Log Level

Вкладка ***Log Level*** позволяет изменить уровень логирования без перезапуска сервиса. Уровень логирования, установленный на этой вкладке, применяется до перезапуска сервиса. После перезапуска используются настройки из файла *log4j.properties*.



Get the current log level for the specified log name.

Set the specified log level for the specified log name.

Вкладка Log Level пользовательского интерфейса HBase

Чтобы проверить уровень логирования, заполните первое поле ***Log Name*** и нажмите кнопку ***Get Log Level***. Имя лога должно содержать иерархию классов, например, `org.apache.hadoop.hbase`.

Чтобы изменить уровень логирования, введите название лога в соответствующее поле. Заполните поле ***Log Level*** одним из следующих значений: `TRACE`, `DEBUG` или `INFO` — и нажмите кнопку ***Set Log Level***.

Debug Dump

На вкладке ***Debug Dump*** отображается отчет о текущем состоянии Master Server. Раздел содержит информацию о версии, задачах, исполнителях (executors), дамп потоков процесса (process thread dump), конфигурацию сервера в формате XML и логи. В случае возникновения проблем в кластере HBase информацию из Debug Dump можно использовать в качестве отправной точки отладки.

Metrics Dump

На вкладке ***Metrics Dump*** отображается отчет с метриками Master Server. Представляет собой дамп JMX bean в формате JSON.

Profiler

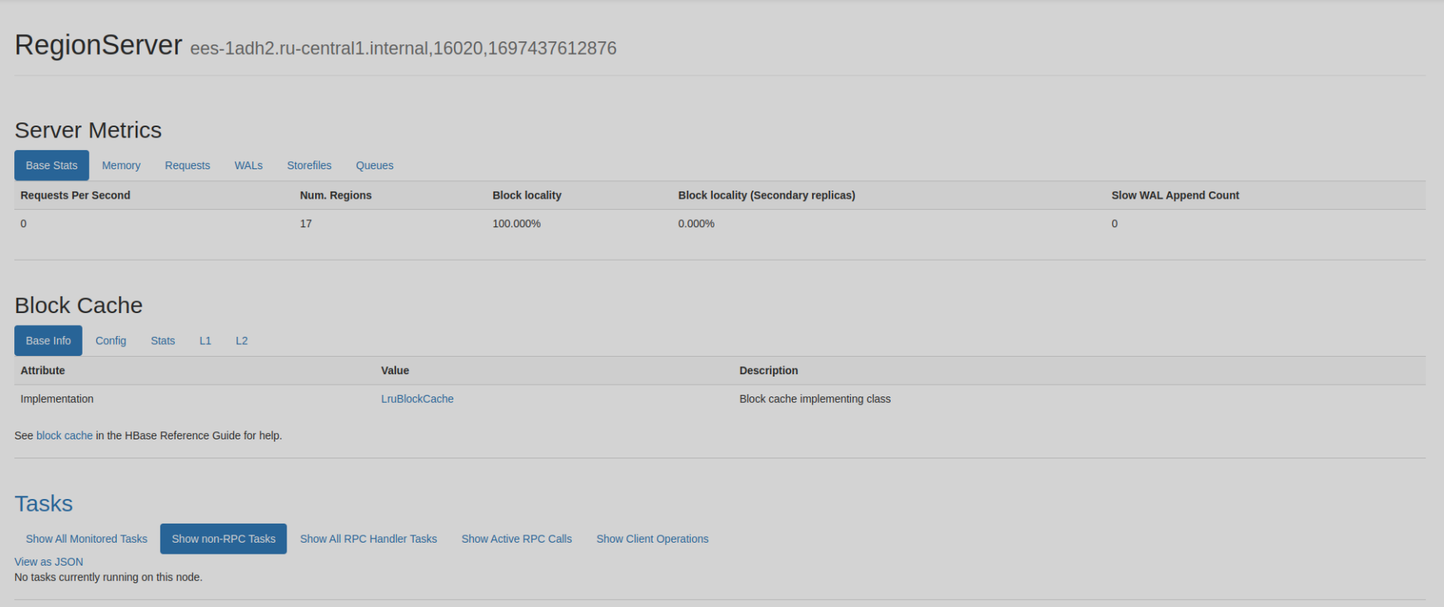
В кластере ADH профилировщик отключен.

HBase Configuration

Вкладка ***HBase Configuration*** содержит настройки конфигурации HBase в формате XML.

Веб-интерфейс Region Server

Пользовательский интерфейс HBase Region Server работает на порте `16030`.



RegionServer ees-1adh2.ru-central1.internal,16020,1697437612876

Server Metrics

Requests Per Second	Num. Regions	Block locality	Block locality (Secondary replicas)	Slow WAL Append Count
0	17	100.000%	0.000%	0

Block Cache

Attribute	Value	Description
Implementation	LruBlockCache	Block cache implementing class

See block cache in the HBase Reference Guide for help.

Tasks

No tasks currently running on this node.

Домашняя страница веб-интерфейса Region Server

Веб-интерфейс HBase Region Server содержит следующие вкладки:

- Home
- Process Metrics
- Local Logs
- Log Level
- Debug Dump
- Metrics Dump
- Profiler
- HBase Configuration

Home

Вкладка ***Home*** интерфейса Region Server содержит следующие разделы с общей информацией о текущем Region Server:

- ***Server Metrics*** — общая статистика текущего сервера, информация об используемой памяти, включая размер memstore, статистика по запросам, WAL, storefiles и размер различных типов очередей.
- ***Block Cache*** — статистика и конфигурация blockcache.
- ***Tasks*** — информация о выполненных задачах. Можно просматривать отдельно статистику по задачам обработчика RPC и задачам, не связанным с RPC. Также ссылка ***View as JSON*** позволяет получить статистику по задачам в формате JSON.
- ***Regions*** — метрики для каждого региона, включая статистику по запросам, storefiles, сжатию (compaction) и размеру memstore. Название региона состоит из имени таблицы, стартового ключа и случайно сгенерированного идентификатора региона, разделенных запятыми. Например: `SYSTEM.LOG,\x01\x00\x00\x00,1695113080532.c4e4b11c5617462e641c9db3538a9c7d.`, `people_ages,F,1694507451861.b53a2f6692e0d4ab5a46a4def96676b0.`
- ***Replication Status*** — информация о репликации и её метрики, если репликация настроена в кластере.
- ***Software Attributes*** — общая информация о JVM, HBase, Hadoop и ZooKeeper. Раздел включает информацию о версиях, контрольных суммах, датах компиляции, адреса зарегистрированных серверов ZK, адрес HBase Master и другие.

Process Metrics

Вкладка ***Process Metrics*** отображает метрики для процесса HRegionServer, который включает следующие элементы:

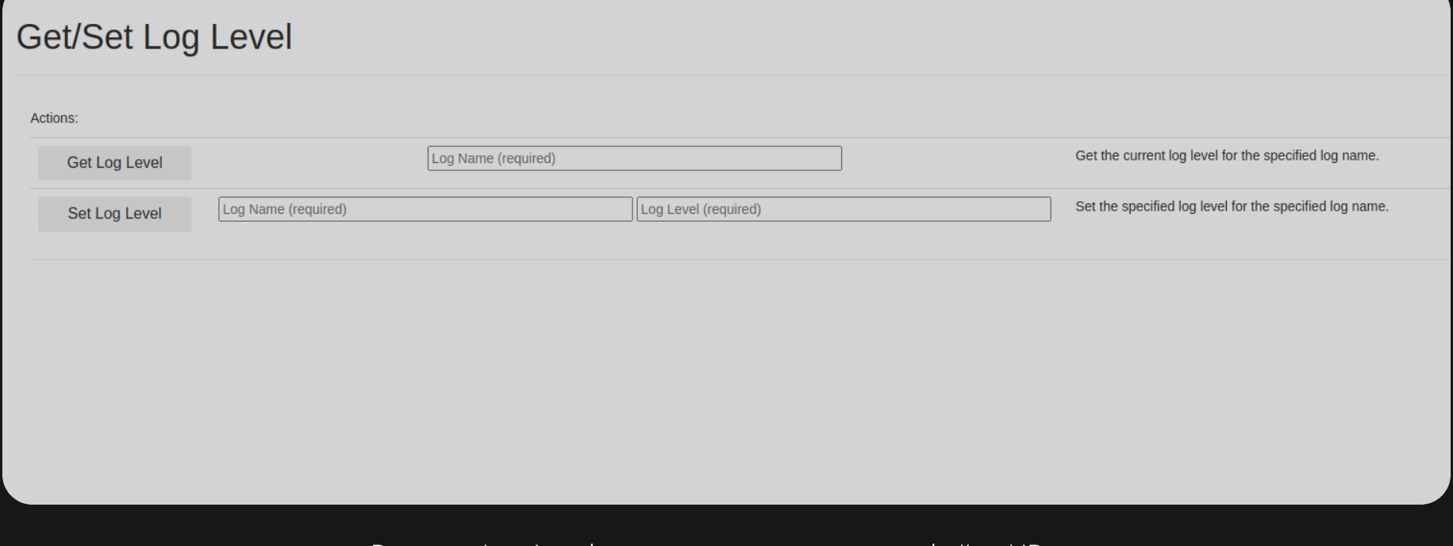
- Threads — статистика потоков процесса.
- GC Collectors — статистика по сборщикам мусора (garbage collectors) Java.
- Metaspace — статистика по `Metaspace`. Область памяти `Metaspace` содержит метаданные классов, загруженных в память, и загрузчики классов.
- Compressed Class Space — статистика по `Compressed Class Space`, области сжатых указателей, которая используется для хранения информации о загруженных классах.
- Par Eden Space — статистика по `Eden Space`, где создается большинство новых объектов Java. Когда пространство памяти, выделенное для этой области, заполняется, доступные объекты из `Eden Space` переносятся в `Survivor Space`.
- Par Survivor Space — статистика по `Survivor Space`. Когда пространство памяти, выделенное для этой области, заполняется, доступные объекты из `Survivor Space` переносятся в `CMS Old Gen`.
- CMS Old Gen — статистика по области памяти `CMS Old Gen`. Эта область Java heap используется для постоянных объектов.

Local Logs

Вкладка ***Local Logs*** содержит список ссылок на файлы журналов, хранящиеся на текущем хосте. При нажатии на ссылку соответствующий файл журнала открывается в окне браузера. В файловой системе файлы журналов хранятся в каталоге */var/log/hbase*. Для получения дополнительной информации обратитесь к статье Логирование в HBase.

Log Level

Вкладка ***Log Level*** позволяет изменить уровень логирования без перезапуска сервиса. Уровень логирования, установленный на этой вкладке, применяется до перезапуска сервиса. После перезапуска используются настройки из файла *log4j.properties*.



Get the current log level for the specified log name.

Set the specified log level for the specified log name.

Вкладка Log Level пользовательского интерфейса HBase

Чтобы проверить уровень логирования, заполните первое поле ***Log Name*** и нажмите кнопку ***Get Log Level***. Имя лога должно содержать иерархию классов, например, `org.apache.hadoop.hbase`.

Чтобы изменить уровень логирования, введите название лога в соответствующее поле. Заполните поле ***Log Level*** одним из следующих значений: `TRACE`, `DEBUG` или `INFO` — и нажмите кнопку ***Set Log Level***.

Debug Dump

На вкладке ***Debug Dump*** отображается отчет о текущем состоянии Region Server.

Metrics Dump

На вкладке ***Metrics Dump*** отображается отчет с метриками Region Server. Представляет собой дамп JMX bean в формате JSON.

Profiler

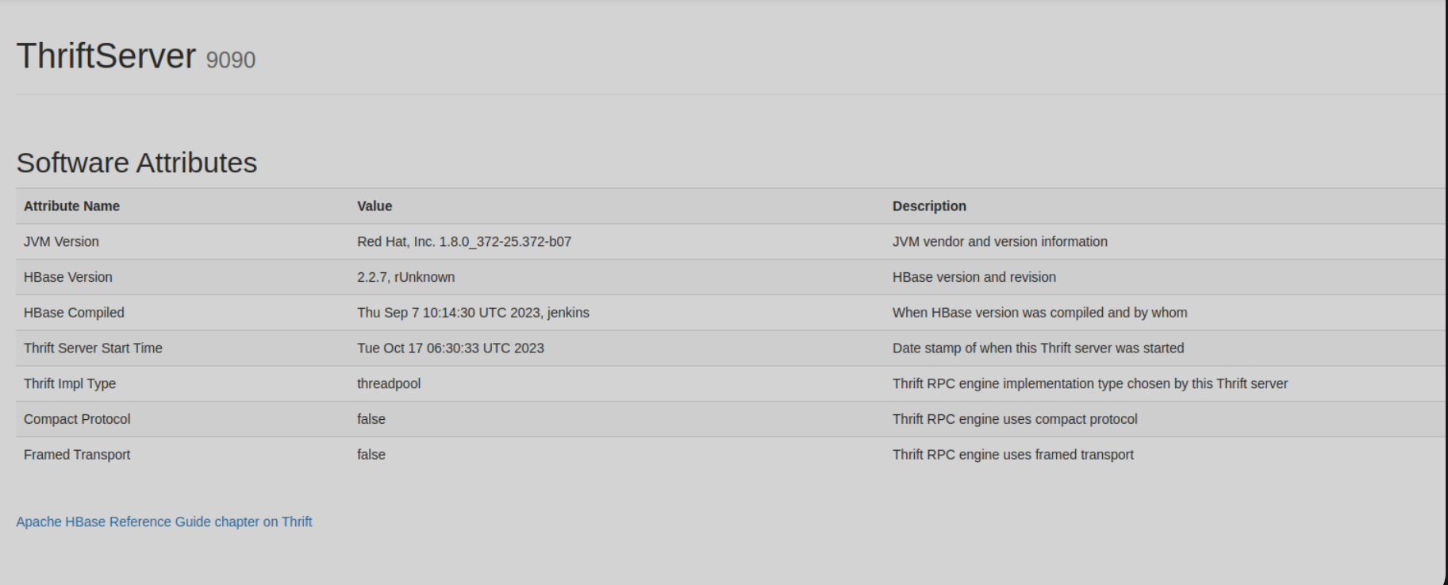
В кластере ADH профилировщик отключен.

HBase Configuration

Вкладка ***HBase Configuration*** содержит настройки конфигурации HBase в формате XML.

Веб-интерфейс Thrift2 Server

Пользовательский интерфейс HBase Thrift2 Server слушает порт `9095`.



ThriftServer 9090

Software Attributes

Attribute Name	Value	Description
JVM Version	Red Hat, Inc. 1.8.0_372-25.372-b07	JVM vendor and version information
HBase Version	2.2.7, rUnknown	HBase version and revision
HBase Compiled	Thu Sep 7 10:14:30 UTC 2023, jenkins	When HBase version was compiled and by whom
Thrift Server Start Time	Tue Oct 17 06:30:33 UTC 2023	Date stamp of when this Thrift server was started
Thrift Impl Type	threadpool	Thrift RPC engine implementation type chosen by this Thrift server
Compact Protocol	false	Thrift RPC engine uses compact protocol
Framed Transport	false	Thrift RPC engine uses framed transport

Apache HBase Reference Guide chapter on Thrift

Домашняя страница веб-интерфейса Thrift2 Server

Веб-интерфейс HBase Thrift2 Server содержит следующие вкладки:

- Home
- Local Logs
- Log Level
- Metrics Dump
- Profiler
- HBase Configuration

Home

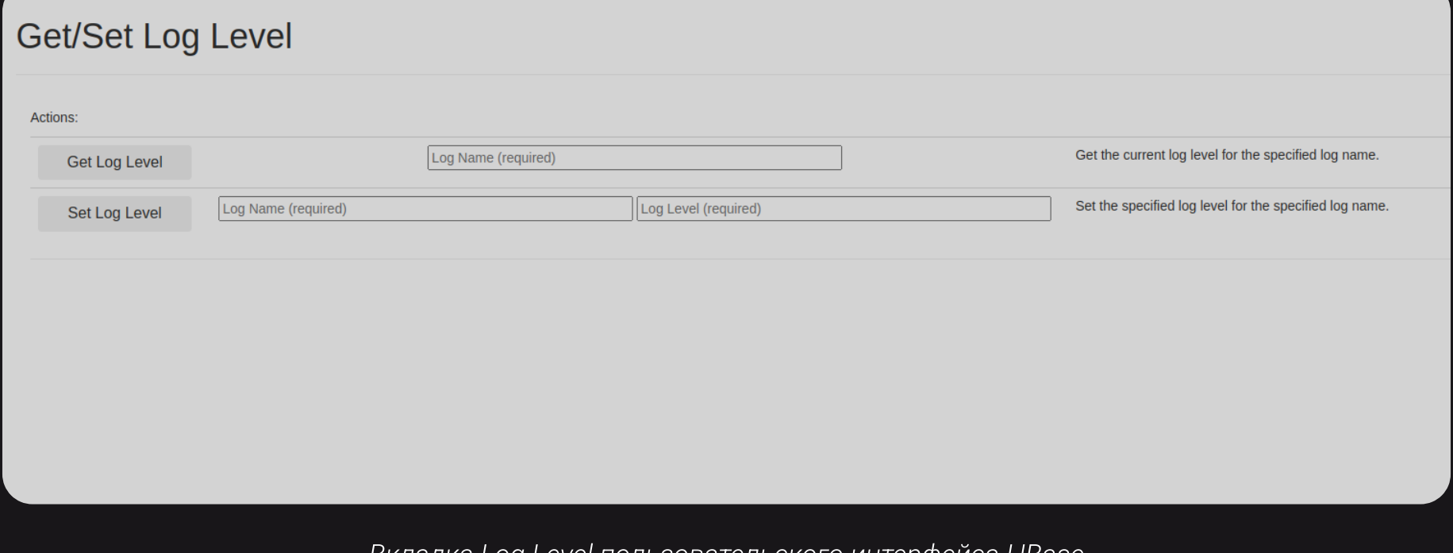
Вкладка ***Home*** отображает таблицу ***Software Attributes***, содержащую версии JVM и HBase, время запуска Thrift2 Server, тип реализации механизма Thrift RPC, выбранный сервером Thrift2, и логические значения, показывающие, использует ли механизм Thrift RPC компактный протокол compact и фреймы (framed transport) при передаче сообщений.

Local Logs

Вкладка ***Local Logs*** содержит список ссылок на файлы журналов, хранящиеся на текущем хосте. При нажатии на ссылку соответствующий файл журнала открывается в окне браузера. В файловой системе файлы журналов хранятся в каталоге */var/log/hbase*. Для получения дополнительной информации обратитесь к статье Логирование в HBase.

Log Level

Вкладка ***Log Level*** позволяет изменить уровень логирования без перезапуска сервиса. Уровень логирования, установленный на этой вкладке, применяется до перезапуска сервиса. После перезапуска используются настройки из файла *log4j.properties*.



Get the current log level for the specified log name.

Set the specified log level for the specified log name.

Вкладка Log Level пользовательского интерфейса HBase

Чтобы проверить уровень логирования, заполните первое поле ***Log Name*** и нажмите кнопку ***Get Log Level***. Имя лога должно содержать иерархию классов, например, `org.apache.hadoop.hbase`.

Чтобы изменить уровень логирования, введите название лога в соответствующее поле. Заполните поле ***Log Level*** одним из следующих значений: `TRACE`, `DEBUG` или `INFO` — и нажмите кнопку ***Set Log Level***.

Metrics Dump

На вкладке ***Metrics Dump*** отображается отчет с метриками Thrift2 Server. Представляет собой дамп JMX bean в формате JSON.

Profiler

В кластере ADH профилировщик отключен.

HBase Configuration

Вкладка ***HBase Configuration*** содержит настройки конфигурации HBase в формате XML.

Начало работы с HBase shell

Альберт Багдасарян

Содержание

Назад к содержанию

Шаг 1. Создание таблицы

Step 2. Получение информации о таблице

Шаг 3. Запись данных в таблицу

Шаг 4. Считывание заданной строки таблицы

- Чтение целой строки
- Чтение заданной колонки

Шаг 5. Сканирование данных таблицы

- Сканирование всех колонок
- Сканирование заданной колонки

Шаг 6. Удаление ячейки из таблицы

Шаг 7. Изменение структуры таблицы

Шаг 8. Удаление таблицы

Простейшим способом начать работу в HBase является использование утилиты HBase shell. Она представляет собой консоль JRuby, доступную на любом узле кластера HBase непосредственно после его размещения. Для перехода в интерактивный режим работы с этой консолью выполните следующую команду:

```
$ hbase shell
```

Командная строка shell позволяет ввести команду после символа приглашения `>`.

Приглашение к вводу команды в HBase shell

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-client.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-hive.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-pig.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-thin-client.jar!/org/slf4j/impl/StaticLoggerBinder.
class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
Version 2.0.2, rUnknown, Thu Apr 15 20:20:26 UTC 2021
Took 0.0024 seconds
hbase(main):001:0>
```

Чтобы завершить сеанс работы с HBase shell, выполните команду выхода:

```
exit
```

Ниже приведены некоторые операции работы с данными. Они основаны на примерах, описанных в разделе Модель данных HBase. Полный список этих и других команд shell с описанием их параметров приведен в разделе Команды HBase shell.

ПРИМЕЧАНИЕ

Прежде чем начать выполнение команд, рекомендуем вам ознакомиться с разделом Модель данных HBase.

Шаг 1. Создание таблицы

Команда создания таблицы начинается с ключевого слова `create`, за которым следует название таблицы в кавычках, а затем информация о семействах колонок (column families) в этой таблице. Для задания семейства колонок используйте один из следующих подходов:

- Ввести только название семейства в кавычках без какой-либо иной информации.
- В фигурных скобках задать несколько атрибутов семейства, используя для каждого атрибута пару ключ/значение.

Используйте следующую команду для создания таблицы `articles` с двумя семействами колонок: `basic` и `tags`. Задайте каждое семейство с помощью двух атрибутов: названия и максимального количества хранимых версий значений. В следующем примере каждая колонка в этих семействах будет хранить до пяти версий значений:

```
create 'articles', {NAME => 'basic', VERSIONS => 5}, {NAME => 'tags', VERSIONS => 5}
```

Вывод на консоль приведен ниже:

```
Created table articles
Took 1.6225 seconds
=> Hbase::Table - articles
```

Step 2. Получение информации о таблице

Чтобы проверить, существует ли таблица с заданным названием, и получить информацию о ней, используйте следующие команды:

- **exists**. Эта команда проверяет, существует ли таблица с заданным названием:

```
exists 'articles'
```

Вывод на консоль приведен ниже:

```
Table articles does exist
Took 0.0705 seconds
=> true
```

Если таблица с заданным названием, например `not_existed`, отсутствует, команда вернет следующее сообщение:

```
Table not_existed does not exist
Took 0.0076 seconds
=> false
```

- **list**. Данная команда выведет на консоль название таблицы, если таблица с заданным названием существует:

```
list 'articles'
```

Вывод на консоль:

```
TABLE
articles
1 row(s)
Took 0.3102 seconds
=> ["articles"]
```

Если таблица с заданным названием отсутствует, команда вернет следующее сообщение:

```
TABLE
0 row(s)
Took 0.0064 seconds
=> []
```

- **describe**. Эта команда проверяет наличие таблицы и, если она существует, отображает информацию о семействах колонок этой таблицы:

```
describe 'articles'
```

Вывод на консоль:

```
Table articles is ENABLED
articles
COLUMN FAMILIES DESCRIPTION
{NAME => 'basic', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => '
false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false'
, IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE
=> 'true', BLOCKSIZE => '6553
6'}
{NAME => 'tags', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'f
alse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
 IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536
'}
2 row(s)
Took 0.1100 seconds
```

Если таблица с заданным названием отсутствует, команда вернет следующее сообщение:

```
ERROR: Table not_existed does not exist.

Describe the named table. For example:
  hbase> describe 't1'
  hbase> describe 'ns1:t1'

Alternatively, you can use the abbreviated 'desc' for the same thing.
  hbase> desc 't1'
  hbase> desc 'ns1:t1'

Took 0.0159 seconds
```

Шаг 3. Запись данных в таблицу

Команда записи данных в таблицу начинается с ключевого слова `put`, за которым следует название таблицы, ключ строки (row key), название колонки и данные. Название колонки состоит из двух частей, разделенных двоеточием — названия семейства и квалификатора колонки (column qualifier). Названия семейств колонок фиксированы. Это означает, что они должны задаваться при создании таблицы. Квалификаторы колонок не фиксированы, так что их можно добавлять в процессе записи новых данных в ранее созданную таблицу.

Приведенные ниже команды записывают данные в таблицу `articles` в строку с ключом `article1`. Они записывают данные во вновь добавленные колонки `author` и `header` в семействе `basic`, а также в колонки `arch`, `concepts` и `tutorials` в семействе `tags`. Записываемые данные занимают последнюю позицию в перечне аргументов команды. HBase автоматически генерирует временные метки для этих новых данных.

```
put 'articles', 'article1', 'basic:author', 'Test author'
put 'articles', 'article1', 'basic:header', 'Test article'
put 'articles', 'article1', 'tags:arch', true
put 'articles', 'article1', 'tags:concepts', true
put 'articles', 'article1', 'tags:tutorials', true
```

Следующие команды также записывают данные в таблицу `articles`, но в строку с другим ключом — `article2`. Как вы заметили, нет необходимости записывать данные только в ячейки с ранее определенными квалификаторами колонок — используйте те, которые необходимы, независимо от того, существуют такие колонки или еще нет:

```
put 'articles', 'article2', 'basic:author', 'Test author2'
put 'articles', 'article2', 'basic:header', 'Test article2'
put 'articles', 'article2', 'tags:ref', true
```

Вывод на консоль:

```
Took 0.0051 seconds
```

Шаг 4. Считывание заданной строки таблицы

HBase предоставляет различные команды для чтения строк таблицы, как описано в следующих секциях.

Чтение целой строки

Для чтения заданной строки из таблицы используйте команду `get`, а в качестве аргументов задайте последовательно название таблицы и ключ строки. Например, следующая команда считывает все ячейки строки с ключом `article1` из таблицы `articles`:

```
get 'articles', 'article1'
```

Команда возвращает все данные из всех колонок, которые использует заданная строка. В выводе на консоль каждая ячейка представлена значением и соответствующей отметкой о времени (timestamp) записи этого значения.

```
COLUMN                                   CELL
 basic:author                            timestamp=1637054560096, value=Test author
 basic:header                            timestamp=1637054560118, value=Test article
 tags:arch                               timestamp=1637054560141, value=true
 tags:concepts                           timestamp=1637054560160, value=true
 tags:tutorials                          timestamp=1637054564066, value=true
1 row(s)
Took 0.0442 seconds
```

Вывод на консоль той же команды для строки с ключом `article2` представлен ниже:

```
COLUMN                                   CELL
 basic:author                            timestamp=1637054576501, value=Test author2
 basic:header                            timestamp=1637054576516, value=Test article2
 tags:ref                                timestamp=1637054577512, value=true
1 row(s)
Took 0.0099 seconds
```

По умолчанию команда `get` показывает только последние версии данных, хотя таблица может хранить до пяти значений с различными версиями, как мы определили при создании таблицы. Если вы добавите новое значение в ячейку, то команда `get` вернет это новое значение, поскольку с ним связана наиболее поздняя временная отметка в данной ячейке.

Следующий пример демонстрирует добавление нового значения в колонку `basic:header` строки с ключом `article1`:

```
put 'articles', 'article1', 'basic:header', 'Test article. Version 2'
get 'articles', 'article1'
```

Последующая команда `get` выведет следующую информацию на экран:

```
COLUMN                                   CELL
 basic:author                            timestamp=1637054560096, value=Test author
 basic:header                            timestamp=1637055836875, value=Test article. Version 2
 tags:arch                               timestamp=1637054560141, value=true
 tags:concepts                           timestamp=1637054560160, value=true
 tags:tutorials                          timestamp=1637054564066, value=true
1 row(s)
Took 0.0066 seconds
```

В следующем примере добавим третью версию значения в колонку `basic:header` строки с ключом `article1`:

```
put 'articles', 'article1', 'basic:header', 'Test article. Version 3'
get 'articles', 'article1'
```

Вывод команды `get` на консоль:

```
COLUMN                                   CELL
 basic:author                            timestamp=1637054560096, value=Test author
 basic:header                            timestamp=1637056832082, value=Test article. Version 3
 tags:arch                               timestamp=1637054560141, value=true
 tags:concepts                           timestamp=1637054560160, value=true
 tags:tutorials                          timestamp=1637054564066, value=true
1 row(s)
Took 0.0064 seconds
```

Чтение заданной колонки

Для чтения заданной колонки в некоторой строке добавьте название этой колонки после названия таблицы и ключа строки в команде `get`. Как обычно, название колонки состоит из названия семейства и квалификатора колонки, разделенных двоеточием.

Следующая команда считывает последнее значение, добавленное в колонку `basic:header` строки с ключом `article1`.

```
get 'articles', 'article1', 'basic:header'
```

Вывод на консоль:

```
COLUMN                                   CELL
 basic:header                            timestamp=1637056832082, value=Test article. Version 3
1 row(s)
Took 0.0227 seconds
```

Из данного результата следует, что команда возвращает последнее добавленное значение. Для получения значения другой версии задайте в фигурных скобках соответствующую временную метку совместно с названием колонки:

```
get 'articles', 'article1', {COLUMN => 'basic:header', TIMESTAMP => 1637054560118}
```

Вывод на консоль:

```
COLUMN                                   CELL
 basic:header                            timestamp=1637054560118, value=Test article
1 row(s)
Took 0.0171 seconds
```

Шаг 5. Сканирование данных таблицы

Команда `scan` позволяет считать все строки таблицы.

Сканирование всех колонок

Для считывания всех колонок всех строк таблицы введите ключевое слово `scan` и название таблицы после него. Следующая команда возвращает полное содержимое таблицы `articles` с последними версиями значений:

```
scan 'articles'
```

Вывод на консоль:

```
ROW                                      COLUMN+CELL
 article1                                column=basic:author, timestamp=1637054560096, value=Test author
 article1                                column=basic:header, timestamp=1637056832082, value=Test article. Version 3
 article1                                column=tags:arch, timestamp=1637054560141, value=true
 article1                                column=tags:concepts, timestamp=1637054560160, value=true
 article1                                column=tags:tutorials, timestamp=1637054564066, value=true
 article2                                column=basic:author, timestamp=1637054576501, value=Test author2
 article2                                column=basic:header, timestamp=1637054576516, value=Test article2
 article2                                column=tags:ref, timestamp=1637054577512, value=true
2 row(s)
Took 0.0204 seconds
```

Для получения значений определенного количества последних версий, не только последней, укажите требуемое количество версий, как в следующей команде:

```
scan 'articles', {VERSIONS => 5}
```

Вывод на консоль:

```
ROW                                      COLUMN+CELL
 article1                                column=basic:author, timestamp=1637054560096, value=Test author
 article1                                column=basic:header, timestamp=1637056832082, value=Test article. Version 3
 article1                                column=basic:header, timestamp=1637055836875, value=Test article. Version 2
 article1                                column=basic:header, timestamp=1637054560118, value=Test article
 article1                                column=tags:arch, timestamp=1637054560141, value=true
 article1                                column=tags:concepts, timestamp=1637054560160, value=true
 article1                                column=tags:tutorials, timestamp=1637054564066, value=true
 article2                                column=basic:author, timestamp=1637054576501, value=Test author2
 article2                                column=basic:header, timestamp=1637054576516, value=Test article2
 article2                                column=tags:ref, timestamp=1637054577512, value=true
2 row(s)
Took 0.0138 seconds
```

Сканирование заданной колонки

Для считывание только одной колонки по всем строкам таблицы задайте в фигурных скобках название требуемой колонки после ключевого слова `scan` и названия таблицы. Следующая команда считывает все значения колонки `basic:author` из таблицы `articles`:

```
scan 'articles', {COLUMN => 'basic:author'}
```

Вывод на консоль:

```
ROW                                      COLUMN+CELL
 article1                                column=basic:author, timestamp=1637054560096, value=Test author
 article2                                column=basic:author, timestamp=1637054576501, value=Test author2
2 row(s)
Took 0.0101 seconds
```

Для считывания нескольких версий значений из заданной колонки укажите требуемое количество версий, как в следующей команде:

```
scan 'articles', {COLUMN => 'basic:header', VERSIONS => 5}
```

Вывод на консоль:

```
ROW                                      COLUMN+CELL
 article1                                column=basic:header, timestamp=1637056832082, value=Test article. Version 3
 article1                                column=basic:header, timestamp=1637055836875, value=Test article. Version 2
 article1                                column=basic:header, timestamp=1637054560118, value=Test article
 article2                                column=basic:header, timestamp=1637054576516, value=Test article2
2 row(s)
Took 0.0298 seconds
```

Шаг 6. Удаление ячейки из таблицы

Для удаления заданной ячейки из таблицы введите ключевое слово `delete` и затем название таблицы, ключ строки, название колонки и, при необходимости, временную метку. Если последний аргумент не указан, то команда удалит последнюю версию ячейки, не затрагивая более ранние версии. Задание временной метки приводит к удалению версии ячейки, соответствующей этой метке.

Следующая команда удаляет последнюю версию ячейки из колонки `basic:header` в строке с ключом `article1`:

```
delete 'articles', 'article1', 'basic:header'
```

Вывод на консоль:

```
Took 0.0122 seconds
```

Выполните команду `get` для проверки:

```
get 'articles', 'article1', {COLUMN => 'basic:header'}
```

Вывод на консоль содержит предыдущую версию ячейки:

```
COLUMN                                   CELL
 basic:header                            timestamp=1637055836875, value=Test article. Version 2
1 row(s)
Took 0.0239 seconds
```

Считывание всех версий ячейки также подтверждает, что значение `Test article. Version 3` было удалено:

```
get 'articles', 'article1', {COLUMN => 'basic:header', VERSIONS => 5}
```

Вывод на консоль:

```
COLUMN                                   CELL
 basic:header                            timestamp=1637055836875, value=Test article. Version 2
 basic:header                            timestamp=1637054560118, value=Test article
1 row(s)
Took 0.0077 seconds
```

Шаг 7. Изменение структуры таблицы

Изменение структуры таблицы выполняют в три стадии:

1. С помощью команды `disable` сделать таблицу недоступной для других операций.
2. Используя команду `alter`, внести необходимые изменения в структуру таблицы.
3. С помощью команда `enable` сделать таблицу вновь доступной для всех операций.

ПРИМЕЧАНИЕ

В последних версиях HBase можно изменять структуру таблиц без предварительного применения команды `disable`. Однако, учитывая проблемы прошлых версий, используйте эту возможность осторожно и тестируйте процедуру прежде, чем применять ее в рабочей (production) базе данных.

Следующая цепочка команд делает таблицу `articles` недоступной, добавляет в нее новое семейство колонок `temp` и затем делает таблицу опять доступной:

```
disable 'articles'
alter 'articles', {NAME => 'temp', VERSIONS => 3}
enable 'articles'
```

Вывод команды `alter` на консоль:

```
Updating all regions with the new schema...
All regions updated.
Done.
Took 1.4006 seconds
```

Убедитесь, что семейство колонок `temp` добавлено в таблицу `articles`, путем выполнения команды `describe`:

```
describe 'articles'
```

Вывод на консоль:

```
Table articles is ENABLED
articles
COLUMN FAMILIES DESCRIPTION
{NAME => 'basic', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => '
false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false'
, IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE
=> 'true', BLOCKSIZE => '6553
6'}
{NAME => 'tags', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'f
alse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
 IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536
'}
{NAME => 'temp', VERSIONS => '3', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'f
alse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
 IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536
'}
3 row(s)
Took 0.0224 seconds
```

Шаг 8. Удаление таблицы

Удаление таблицы требует следующих операций:

1. С помощью команды `disable` сделать таблицу недоступной для других операций.
2. Используя команду `drop`, удалить таблицу.

Следующая цепочка команд делает таблицу недоступной и затем удаляет ее:

```
disable 'articles'
drop 'articles'
```

С помощью команды `exists` убедитесь, что таблица больше не существует:

```
exists 'articles'
```

Вывод на консоль:

```
Table articles does not exist
Took 0.0047 seconds
=> false
```

Использование фильтров в HBase

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
- Синтаксис фильтров HBase
  - Простой фильтр
  - Составной фильтр
- Индивидуальные фильтры в HBase
- Динамическая загрузка кастомных фильтров

Обзор

HBase в ADH предоставляет **язык фильтров Thrift (Thrift filter language)**. Он позволяет фильтровать результаты чтения данных из HBase при использовании команд `get` и `scan`. Фильтрация происходит на стороне сервера и нагрузку на HBase не уменьшает. Однако уменьшается нагрузка на сеть, поскольку клиенту передается меньше данных. Фильтры могут использоваться как при взаимодействии через Java API, так и в консоли HBase shell.

Синтаксис фильтров HBase

Простой фильтр

Простой фильтр задается строкой следующего вида:

```
"<FilterName> (<arg1>, <arg2>, ..., <argN>)"
```

где:

- `<FilterName>` — имя одного из индивидуальных фильтров;
- `<arg1>`, `<arg2>` и т.д. — аргументы этого фильтра.

Пример использования простого фильтра в консоли HBase shell:

```
scan 'articles', { FILTER => "ColumnPaginationFilter (2, 1)"}
```

Полный набор аргументов — это условие фильтра. Различные индивидуальные фильтры могут принимать разное количество аргументов — от нуля и более.

Аргументы могут иметь следующие типы:

- строка;
- целое;
- логический;
- метка времени;
- оператор сравнения;
- компаратор.

Метки времени

В консоли HBase shell метки времени в результатах чтения таблиц отображаются в человекочитаемом формате. Однако, если фильтр требует метку времени в качестве значения аргумента, она должна быть задана в формате эпохи Unix. Для преобразования меток времени из одного формата в другой используйте онлайн-инструменты, например EpochConverter.

ПРИМЕЧАНИЕ

Для меток времени в HBase требуется точность до миллисекунд. Вам потребуется вручную дописывать миллисекунды к меткам времени после преобразования в формат эпохи Unix.

Операторы сравнения и компараторы

Операторы сравнения и компараторы используются в аргументах фильтров для составления условий лексикографических соответствий и сравнений.

Следующие операторы сравнения используются в качестве аргументов фильтра.

Операторы сравнения

Синтаксис	Описание
<	Меньше
>	Больше
=	Равно
<=	Меньше или равно
>=	Больше или равно
!=	Не равно

Следующие компараторы используются в качестве аргументов фильтра.

Компараторы

Название	Описание	Пример
BinaryComparator	Лексикографически сравнивает с заданной строкой	(>, 'binary:arc') Такой компаратор соответствует всему, что лексикографически больше чем `arc`
BinaryPrefixComparator	Лексикографически сравнивает с заданной строкой. Сравнение выполняется только до длины этой строки	(=, 'binaryprefix:bot') Такой компаратор соответствует всему, что начинается с `bot`
RegexStringComparator	Лексикографически сравнивает с заданным регулярным выражением для строки. С этим компаратором можно использовать только операторы сравнения `=` и `!=`	(!=, 'regexstring:be*ng') Такой компаратор соответствует любой строке, кроме тех, которые начинаются с `be` и заканчиваются на `ng`
SubStringComparator	Ищет заданную подстроку без учета регистра. С этим компаратором можно использовать только операторы сравнения `=` и `!=`	(!=, substring:con) Такой компаратор соответствует любой строке, кроме тех, в которых содержится `con`

Правила синтаксиса фильтров

Синтаксис фильтров подчиняется следующим правилам:

- Выражение фильтра должно быть заключено в двойные кавычки (`"`).
- Именем фильтра должно быть одно слово. Допустимы любые ASCII-символы, кроме круглых скобок, одинарных кавычек и пробелов.
- Аргументы должны быть разделены запятыми и все вместе заключены в круглые скобки.
- Строковые аргументы должны быть каждый по отдельности заключены в одинарные кавычки (`'`).
- Аргументы других типов (целое, логический, оператор сравнения и т.д.) не должны заключаться в кавычки.
- Аргументы фильтра могут содержать любые ASCII-символы. Однако если в аргументе присутствуют одинарные кавычки — каждая из них должна быть экранирована дополнительной одинарной кавычкой.

Составной фильтр

Составной фильтр состоит из индивидуальных фильтров, объединенных логическими операторами. Существуют бинарные и унарные логические операторы. Бинарные операторы объединяют фильтры слева и справа от себя:

- `AND` — пара ключ/значение возвращается только если она удовлетворяет обоим фильтрам.
- `OR` — пара ключ/значение возвращается, если она удовлетворяет любому из фильтров.

Унарные операторы предшествуют фильтру и меняют его поведение:

- `SKIP` — если одна из пар ключ/значение не удовлетворяет условию фильтра, то вся строка отбрасывается.
- `WHILE` — для каждой строки пары ключ/значение продолжают извлекаться до тех пор, пока одна из них не перестанет удовлетворять условию фильтра.

Используйте скобки для определения порядка обработки индивидуальных фильтров в составных фильтрах. Пример:

```
(<Filter1> OR <Filter2>) AND (<Filter3> OR <Filter4>)
```

Сначала обрабатывается результат объединения фильтров `<Filter1>` и `<Filter2>` оператором `OR`. Затем то же самое происходит с фильтрами `<Filter3>` и `<Filter4>`. В конце эти два результата объединяются оператором `AND`.

Логические операторы и скобки имеют следующие приоритеты, от высшего к низшему:

1. скобки;
2. `SKIP` и `WHILE` (имеют равный приоритет);
3. `AND`;
4. `OR`.

Пример использования составного фильтра в консоли HBase shell:

```
scan 'articles', { FILTER => "ColumnPaginationFilter (2, 1) AND PrefixFilter ('co')"}
```

Индивидуальные фильтры в HBase

Представленные в HBase индивидуальные фильтры и их синтаксис перечислены ниже. Примеры запросов выполнены для тестовой таблицы, заполненной значениями из тестового файла people.csv. Этот файл является расширенной версией тестового файла, использованного в статье Массовая загрузка с использованием встроенных задач MapReduce. Используйте инструкции из этой статьи, если хотите создать такую же тестовую таблицу, которая используется здесь. В этом случае следует использовать видоизмененную команду для запуска задачи `ImportTsv`: вместо флага `-Dimporttsv.columns=HBASE_ROW_KEY,basic:age` задайте `-Dimporttsv.columns=HBASE_ROW_KEY,basic:age,location:town,location:state,location:country`. При необходимости, соответствующим образом замените имена файла и таблицы в команде.

Ключами строк таблицы являются имена людей. Первое семейство столбцов называется `basic` и имеет единственный квалификатор `age`. Второе семейство столбцов называется `location` и имеет три квалификатора: `country`, `state` и `town`. В таблице 997 строк.

KeyOnlyFilter

Возвращает ключ из каждой пары ключ/значение. Не принимает аргументов.

Синтаксис:

```
"KeyOnlyFilter ()"
```

Пример команды:

```
scan 'people', { FILTER => "KeyOnlyFilter ()" }
```

Результат (последние 8 ключей):

```
 ...
 Zimmerman Gene                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=
 Zimmerman Gene                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=
 Zimmerman Gene                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=
 Zimmerman Gene                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=
 Zimmerman Madge                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=
 Zimmerman Madge                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=
 Zimmerman Madge                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=
 Zimmerman Madge                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=
997 row(s)
Took 0.5022 seconds
```

FirstKeyOnlyFilter

Возвращает первую пару ключ/значение из каждой строки. Не принимает аргументов.

Синтаксис:

```
"FirstKeyOnlyFilter ()"
```

Пример команды:

```
scan 'people', { FILTER => "KeyOnlyFilter ()" }
```

Результат (первые и последние пять пар ключ/значение):

```
ROW                                          COLUMN+CELL
 Abbott Delia                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=62
 Abbott Howard                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Abbott Jack                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Adams Clyde                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Aguilar Myrtie                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=23
 ...
 Young Della                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=21
 Young Josephine                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Young Mattie                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=39
 Zimmerman Gene                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=35
 Zimmerman Madge                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=46
997 row(s)
Took 0.1058 seconds
```

PrefixFilter

Возвращает все пары ключ/значение из тех строк, чьи ключи начинаются с префикса, заданного аргументом. Принимает один аргумент: префикс ключа строки.

Синтаксис:

```
"PrefixFilter ('Roy')"
```

Пример команды:

```
scan 'people', { FILTER => "PrefixFilter ('Roy')" }
```

Результат:

```
ROW                                         COLUMN+CELL
 Roy Alfred                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=55
 Roy Alfred                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Roy Alfred                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=Chihuahua
 Roy Alfred                                 column=location:town, timestamp=2024-07-30T08:10:19.297, value=Juarez
 Roy Belle                                  column=basic:age, timestamp=2024-07-30T08:10:19.297, value=35
 Roy Belle                                  column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Roy Belle                                  column=location:state, timestamp=2024-07-30T08:10:19.297, value=AZ
 Roy Belle                                  column=location:town, timestamp=2024-07-30T08:10:19.297, value=Nogales
 Roy Lora                                   column=basic:age, timestamp=2024-07-30T08:10:19.297, value=52
 Roy Lora                                   column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Roy Lora                                   column=location:state, timestamp=2024-07-30T08:10:19.297, value=Yucatan
 Roy Lora                                   column=location:town, timestamp=2024-07-30T08:10:19.297, value=Uman
 Roy Ronald                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=63
 Roy Ronald                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Roy Ronald                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=California
Baja
 Roy Ronald                                 column=location:town, timestamp=2024-07-30T08:10:19.297, value=Mexicali
4 row(s)
Took 0.0252 seconds
```

ColumnPrefixFilter

Возвращает все пары ключ/значение из тех столбцов, чьи квалификаторы начинаются с префикса, заданного аргументом. Принимает один аргумент: префикс квалификатора столбца.

Синтаксис:

```
"ColumnPrefixFilter ('tow')"
```

Пример команды:

```
scan 'people', { FILTER => "ColumnPrefixFilter ('tow')" }
```

Поскольку все строки тестовой таблицы имеют непустые значения для всех столбцов, такой фильтр вернет либо 0 результатов, если ни один из квалификаторов столбцов не имеет заданного префикса, либо все 997, если такой квалификатор есть. Приведенная выше команда вернет 997 пар ключ/значение, первые и последние пять из которых приведены ниже:

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Abbott Howard                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Baton Rouge
 Abbott Jack                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Juarez
 Adams Clyde                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Lafayette
 Aguilar Myrtie                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Uman
 ...
 Young Della                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Chetumal
 Young Josephine                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Redding
 Young Mattie                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=San Jose
 Zimmerman Gene                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Tijuana
 Zimmerman Madge                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Nogales
997 row(s)
Took 0.1360 seconds
```

MultipleColumnPrefixFilter

Возвращает все пары ключ/значение из тех столбцов, чьи квалификаторы начинаются с любого из префиксов, заданных аргументами. Принимает один или несколько аргументов: префиксы квалификаторов столбцов. В случае единственного аргумента работает как фильтр `ColumnPrefixFilter`.

Синтаксис:

```
"MultipleColumnPrefixFilter ('sta', 'tow')"
```

Пример команды:

```
scan 'people', { FILTER => "MultipleColumnPrefixFilter ('sta', 'tow')" }
```

Здесь происходит то же, что и в предыдущем примере (фильтр `ColumnPrefixFilter`): поскольку все строки тестовой таблицы имеют непустые значения для всех столбцов, такой фильтр вернет либо 0 результатов, если ни один из квалификаторов столбцов не имеет какого-либо из заданных префиксов, либо все 997 на каждый найденный квалификатор. Если один из заданных префиксов является префиксом другого (например, `to` и `tow`), количество результатов не увеличивается. Приведенная выше команда вернет 997 пар ключ/значение, первые и последние пять из которых приведены ниже:

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Abbott Delia                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Abbott Howard                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Abbott Howard                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Baton Rouge
 Abbott Jack                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=Chihuahua
 Abbott Jack                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Juarez
 ...
 Young Mattie                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=CA
 Young Mattie                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=San Jose
 Zimmerman Gene                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=California
Baja
 Zimmerman Gene                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Tijuana
 Zimmerman Madge                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=AZ
 Zimmerman Madge                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Nogales
997 row(s)
Took 0.1360 seconds
```

ColumnCountGetFilter

Возвращает заданное аргументом количество столбцов в строке, начиная с первого. Этот фильтр корректно работает только с отдельными строками, поэтому его следует использовать с командой `get`. Использование этого фильтра с командой `scan` не приводит к ошибке, но возвращает некорректный результат. Принимает один аргумент: количество столбцов.

Синтаксис:

```
"ColumnCountGetFilter (3)"
```

Пример команды:

```
get 'people', 'Abbott Delia', { FILTER => "ColumnCountGetFilter (3)" }
```

Результат:

```
COLUMN                                      CELL
 basic:age                                  timestamp=2024-07-30T08:10:19.297, value=62
 location:country                           timestamp=2024-07-30T08:10:19.297, value=USA
 location:state                             timestamp=2024-07-30T08:10:19.297, value=TX
1 row(s)
Took 0.6243 seconds
```

PageFilter

Возвращает заданное аргументом количество строк таблицы, начиная с первой, из каждого региона. Принимает один аргумент: количество строк.

Синтаксис:

```
"PageFilter (2)"
```

Пример команды:

```
scan 'people', { FILTER => "PageFilter (2)" }
```

Поскольку в тестовой таблице было создано пять регионов, разделенных в точках `F`, `K`, `P` и `W`, эта команда вернет по две строки с четырьмя парами ключ/значение в каждой из каждого региона. Результат:

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=62
 Abbott Delia                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Abbott Delia                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Abbott Delia                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Abbott Howard                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Abbott Howard                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Abbott Howard                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Abbott Howard                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Baton Rouge
 Farmer Alan                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Farmer Alan                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Farmer Alan                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=Quintana Roo
 Farmer Alan                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Cancun
 Farmer Dean                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=65
 Farmer Dean                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Farmer Dean                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Farmer Dean                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Baton Rouge
 Keller Elmer                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Keller Elmer                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Keller Elmer                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=Quintana Roo
 Keller Elmer                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Chetumal
 Keller Flora                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=23
 Keller Flora                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Keller Flora                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Keller Flora                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Houston
 Padilla Ethan                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=61
 Padilla Ethan                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Padilla Ethan                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=NV
 Padilla Ethan                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Novac
 Padilla Scott                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=28
 Padilla Scott                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Padilla Scott                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Padilla Scott                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Beaumont
 Wade Janie                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=25
 Wade Janie                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Wade Janie                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=California
Baja
 Wade Janie                                 column=location:town, timestamp=2024-07-30T08:10:19.297, value=Ensenada
 Wade Jerome                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=27
 Wade Jerome                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Wade Jerome                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Wade Jerome                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Houston
10 row(s)
Took 0.0138 seconds
```

ColumnPaginationFilter

Возвращает заданное первым аргументом (предел) количество столбцов после столбца с номером по счету, заданным вторым аргументом (отступ). Принимает два аргумента: предел и отступ.

Синтаксис:

```
"ColumnPaginationFilter (2, 1)"
```

Пример команды:

```
scan 'people', { FILTER => "ColumnPaginationFilter (2, 1)" }
```

Эта команда возвращает 1994 пары ключ/значение, первые и последние шесть из которых приведены ниже:

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Abbott Delia                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Abbott Howard                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Abbott Howard                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Abbott Jack                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Abbott Jack                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=Chihuahua
 ...
 Young Mattie                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Young Mattie                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=CA
 Zimmerman Gene                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Zimmerman Gene                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=California Baja
 Zimmerman Madge                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Zimmerman Madge                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=AZ
997 row(s)
Took 1.0474 seconds
```

InclusiveStopFilter

Выполняет построчное сканирование таблицы до нахождения заданного ключа строки, затем возвращает все строки вплоть до той, в которой был найден ключ, включительно. Принимает одно значение: ключ строки.

Синтаксис:

```
"InclusiveStopFilter (3)"
```

Пример команды:

```
scan 'people', { FILTER => "InclusiveStopFilter ('Allen Austin')" }
```

Результат:

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=62
 Abbott Delia                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Abbott Delia                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Abbott Delia                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Abbott Howard                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Abbott Howard                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Abbott Howard                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Abbott Howard                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Baton Rouge
 Abbott Jack                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Abbott Jack                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Abbott Jack                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=Chihuahua
 Abbott Jack                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Juarez
 Adams Clyde                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Adams Clyde                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Adams Clyde                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Adams Clyde                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Lafayette
 Aguilar Myrtie                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=23
 Aguilar Myrtie                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Aguilar Myrtie                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=Yucatan
 Aguilar Myrtie                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Uman
 Aguilar Terry                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=65
 Aguilar Terry                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Aguilar Terry                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=CA
 Aguilar Terry                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Fresno
 Alexander Derrick                          column=basic:age, timestamp=2024-07-30T08:10:19.297, value=46
 Alexander Derrick                          column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alexander Derrick                          column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Alexander Derrick                          column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Alexander Gregory                          column=basic:age, timestamp=2024-07-30T08:10:19.297, value=54
 Alexander Gregory                          column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alexander Gregory                          column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Alexander Gregory                          column=location:town, timestamp=2024-07-30T08:10:19.297, value=Lafayette
 Alexander Leon                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=42
 Alexander Leon                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alexander Leon                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=AZ
 Alexander Leon                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Kingman
 Allen Austin                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=34
 Allen Austin                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Allen Austin                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=NV
 Allen Austin                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Primm
10 row(s)
Took 0.7167 seconds
```

TimestampsFilter

Возвращает все пары ключ/значение, чьи метки времени совпадают с любой из заданных. Принимает любое количество аргументов: метки времени.

Синтаксис:

```
"TimestampsFilter (1721203180857, 1721316861863)"
```

Все пары ключ/значение в тестовой таблице имеют одну и ту же метку времени, поэтому для более наглядного результата имеет смысл поправить несколько ключей с помощью команды `put`:

```
put 'people', 'Young Mattie', 'location:town', 'Carlsbad', 1722327020000
put 'people', 'Young Mattie', 'location:state', 'NM', 1722327020000
put 'people', 'Yates Douglas', 'location:state', 'NM', 1722327020000
put 'people', 'Yates Douglas', 'location:town', 'Albuquerque', 1722327020000
```

Пример команды:

```
scan 'people', { FILTER => "TimestampsFilter (1722327020000)" }
```

Результат:

```
ROW                                         COLUMN+CELL
 Yates Douglas                              column=location:state, timestamp=2024-07-30T08:10:20, value=NM
 Yates Douglas                              column=location:town, timestamp=2024-07-30T08:10:20, value=Albuquerque
 Young Mattie                               column=location:state, timestamp=2024-07-30T08:10:20, value=NM
 Young Mattie                               column=location:town, timestamp=2024-07-30T08:10:20, value=Carlsbad
2 row(s)
Took 0.1001 seconds
```

RowFilter

Проверяет все строки и возвращает все пары ключ/значение из строки, если значение ключа этой строки удовлетворяет условию сравнения, заданного аргументами. Этот фильтр не работает с командой `get`. Принимает два аргумента: оператор сравнения и компаратор.

Синтаксис:

```
"RowFilter (>=, 'binaryprefix:B')"
```

Пример команды:

```
scan 'people', { FILTER => "RowFilter (<, 'binaryprefix:B')" }
```

Этот фильтр возвращает все пары ключ/значение из строк, ключи которых начинаются со значений, лексикографически меньших `B`. В случае нашей тестовой таблицы это означает все строки с людьми, чьи фамилии начинаются с `A`. Результат:

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=62
 Abbott Delia                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Abbott Delia                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Abbott Delia                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Abbott Howard                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Abbott Howard                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Abbott Howard                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Abbott Howard                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Baton Rouge
 Abbott Jack                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Abbott Jack                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Abbott Jack                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=Chihuahua
 Abbott Jack                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Juarez
 Adams Clyde                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Adams Clyde                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Adams Clyde                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Adams Clyde                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Lafayette
 Aguilar Myrtie                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=23
 Aguilar Myrtie                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Aguilar Myrtie                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=Yucatan
 Aguilar Myrtie                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Uman
 Aguilar Terry                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=65
 Aguilar Terry                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Aguilar Terry                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=CA
 Aguilar Terry                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Fresno
 Alexander Derrick                          column=basic:age, timestamp=2024-07-30T08:10:19.297, value=46
 Alexander Derrick                          column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alexander Derrick                          column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Alexander Derrick                          column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Alexander Gregory                          column=basic:age, timestamp=2024-07-30T08:10:19.297, value=54
 Alexander Gregory                          column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alexander Gregory                          column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Alexander Gregory                          column=location:town, timestamp=2024-07-30T08:10:19.297, value=Lafayette
 Alexander Leon                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=42
 Alexander Leon                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alexander Leon                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=AZ
 Alexander Leon                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Kingman
 Allen Austin                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=34
 Allen Austin                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Allen Austin                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=NV
 Allen Austin                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Primm
 Allison Dustin                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=54
 Allison Dustin                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Allison Dustin                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=Yucatan
 Allison Dustin                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Merida
 Alvarado Dominic                           column=basic:age, timestamp=2024-07-30T08:10:19.297, value=63
 Alvarado Dominic                           column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Alvarado Dominic                           column=location:state, timestamp=2024-07-30T08:10:19.297, value=California Baja
 Alvarado Dominic                           column=location:town, timestamp=2024-07-30T08:10:19.297, value=Ensenada
 Alvarado Maria                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=58
 Alvarado Maria                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Alvarado Maria                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=Chihuahua
 Alvarado Maria                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Chihuahua
 Alvarado Melvin                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=34
 Alvarado Melvin                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alvarado Melvin                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Alvarado Melvin                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Austin
 Alvarado Timothy                           column=basic:age, timestamp=2024-07-30T08:10:19.297, value=27
 Alvarado Timothy                           column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alvarado Timothy                           column=location:state, timestamp=2024-07-30T08:10:19.297, value=CA
 Alvarado Timothy                           column=location:town, timestamp=2024-07-30T08:10:19.297, value=San Francisco
 Alvarez Bessie                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=34
 Alvarez Bessie                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alvarez Bessie                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=NV
 Alvarez Bessie                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Las Vegas
 Alvarez Bruce                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=60
 Alvarez Bruce                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alvarez Bruce                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=CA
 Alvarez Bruce                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Modesto
 Alvarez Harvey                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=57
 Alvarez Harvey                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Alvarez Harvey                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=NV
 Alvarez Harvey                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Fallon
 Alvarez Jacob                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=49
 Alvarez Jacob                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Alvarez Jacob                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=Yucatan
 Alvarez Jacob                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Merida
 Anderson Lester                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=62
 Anderson Lester                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Anderson Lester                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=Sonora
 Anderson Lester                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Hermosillo
 Anderson Lucile                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=33
 Anderson Lucile                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Anderson Lucile                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=Sonora
 Anderson Lucile                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Hermosillo
 Anderson Ora                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=26
 Anderson Ora                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Anderson Ora                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=Sonora
 Anderson Ora                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Nogales
 Andrews Caleb                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=22
 Andrews Caleb                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Andrews Caleb                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=Quintana Roo
 Andrews Caleb                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Cancun
 Andrews Lucy                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=33
 Andrews Lucy                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Andrews Lucy                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=AZ
 Andrews Lucy                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Kingman
 Andrews Noah                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=63
 Andrews Noah                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Andrews Noah                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=California Baja
 Andrews Noah                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Ensenada
 Andrews Susan                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=48
 Andrews Susan                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Andrews Susan                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=Yucatan
 Andrews Susan                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Merida
 Armstrong Isabella                         column=basic:age, timestamp=2024-07-30T08:10:19.297, value=49
 Armstrong Isabella                         column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Armstrong Isabella                         column=location:state, timestamp=2024-07-30T08:10:19.297, value=Sonora
 Armstrong Isabella                         column=location:town, timestamp=2024-07-30T08:10:19.297, value=Hermosillo
 Armstrong Marie                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=25
 Armstrong Marie                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Armstrong Marie                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Armstrong Marie                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Santa Fe
 Arnold Bettie                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=18
 Arnold Bettie                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Arnold Bettie                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=NV
 Arnold Bettie                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Primm
 Atkins Daisy                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=51
 Atkins Daisy                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Atkins Daisy                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=Quintana Roo
 Atkins Daisy                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Chetumal
 Atkins Gene                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=37
 Atkins Gene                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Atkins Gene                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Atkins Gene                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Austin Bertie                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=57
 Austin Bertie                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=MEX
 Austin Bertie                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=California Baja
 Austin Bertie                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Tijuana
 Austin Eugene                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=64
 Austin Eugene                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Austin Eugene                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=LA
 Austin Eugene                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Lafayette
 Austin Travis                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=53
 Austin Travis                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Austin Travis                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Austin Travis                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=El Paso
34 row(s)
Took 0.6767 seconds
```

FamilyFilter

Проверяет все столбцы и возвращает все пары ключ/значение из столбца, если имя семейства этого столбца удовлетворяет условию сравнения, заданного аргументами. Принимает два аргумента: оператор сравнения и компаратор.

Синтаксис:

```
"FamilyFilter (\<=, 'binaryprefix:c')"
```

Пример команды:

```
scan 'people', { FILTER => "FamilyFilter (<, 'binaryprefix:c')" }
```

Этот фильтр возвращает все пары ключ/значение, семейства столбцов которых начинаются со значений, лексикографически меньших `c`. В случае нашей тестовой таблицы это означает весь столбец `basic:age`. Результат (первые и последние пять строк):

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=62
 Abbott Howard                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Abbott Jack                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Adams Clyde                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Aguilar Myrtie                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=23
 ...
 Young Della                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=21
 Young Josephine                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Young Mattie                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=39
 Zimmerman Gene                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=35
 Zimmerman Madge                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=46
997 row(s)
Took 0.4836 seconds
```

QualifierFilter

Проверяет все столбцы и возвращает все пары ключ/значение из столбца, если имя квалификатора этого столбца удовлетворяет условию сравнения, заданного аргументами. Принимает два аргумента: оператор сравнения и компаратор.

Синтаксис:

```
"QualifierFilter (=, 'binary:town')"
```

Пример команды:

```
scan 'people', { FILTER => "QualifierFilter (=, 'binary:town')" }
```

Этот фильтр возвращает все пары ключ/значение, квалификатор столбца которых строго равен `town`. Результат (первые и последние пять строк):

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Dallas
 Abbott Howard                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Baton
Rouge
 Abbott Jack                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Juarez
 Adams Clyde                                column=location:town, timestamp=2024-07-30T08:10:19.297,
value=Lafayette
 Aguilar Myrtie                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Uman
 ...
 Young Della                                column=location:town, timestamp=2024-07-30T08:10:19.297,
value=Chetumal
 Young Josephine                            column=location:town, timestamp=2024-07-30T08:10:19.297,
value=Redding
 Young Mattie                               column=location:town, timestamp=2024-07-30T08:10:20, value=Carlsbad
 Zimmerman Gene                             column=location:town, timestamp=2024-07-30T08:10:19.297,
value=Tijuana
 Zimmerman Madge                            column=location:town, timestamp=2024-07-30T08:10:19.297,
value=Nogales
997 row(s)
Took 0.2112 seconds
```

ValueFilter

Возвращает все пары ключ/значение, значения которых удовлетворяют условию сравнения, заданного аргументами. Принимает два значения: оператор сравнения и компаратор.

Синтаксис:

```
"ValueFilter (=, 'binary:TX')"
```

Пример команды:

```
scan 'people', { FILTER => "ValueFilter (=, 'binary:TX')" }
```

Этот фильтр возвращает список техасцев из тестовой таблицы, то есть все пары ключ/значение, строго равные `TX`. Результат (первые и последние пять строк):

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Alexander Derrick                          column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Alvarado Melvin                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Atkins Gene                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Austin Travis                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 ...
 Watkins Julian                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Welch Lela                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Willis Travis                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Wilson Grace                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
 Wilson Nellie                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=TX
112 row(s)
Took 0.0535 seconds
```

DependentColumnFilter

Ищет в каждой строке столбец, определенный двумя обязательными аргументами: семейство столбцов и квалификатор столбца. Если такой столбец найден (**референтный столбец**), то возвращает все пары ключ/значение из этой строки, имеющие такую же метку времени как и референтный столбец. Если не найден — не возвращает ничего.

Если задан необязательный третий аргумент (`dropDependentColumn`) логического типа, то референтный столбец либо возвращается (при значении `false`), либо нет (при значении `true`).

Можно задать еще два аргумента: оператор сравнения и компаратор. В этом случае значение столбца также должно пройти проверку сравнения, заданного этими аргументами, чтобы столбец считался референтным.

Принимает два, три или пять аргументов: семейство столбцов (обязательно), квалификатор столбца (обязательно), флаг `dropDependentColumn` (логический, необязательно), оператор сравнения и компаратор (последние два аргумента могут быть заданы только совместно).

Синтаксис:

```
"DependentColumnFilter ('location', 'town')"
```

```
"DependentColumnFilter ('location', 'town', true)"
```

```
"DependentColumnFilter ('location', 'town', true, =, 'binary:Carlsbad')"
```

Пример команды:

```
scan 'people', { FILTER => "DependentColumnFilter ('location', 'town', true, =, 'binary:Carlsbad')" }
```

Этот фильтр ищет в каждой строке точное значение `Carlsbad` в столбце с точным именем `location:town` и в случае успеха возвращает остальные пары ключ/значение, имеющие ту же метку времени, опуская саму найденную пару ключ/значение. Результат:

```
ROW                                         COLUMN+CELL
 Young Mattie                               column=location:state, timestamp=2024-07-30T08:10:20, value=NM
1 row(s)
Took 0.0269 seconds
```

Этот пример опирается на поправки, внесенные в примере работы фильтра `TimestampsFilter`. Без них фильтр ничего не вернет, поскольку условие сравнения не будет выполнено.

SingleColumnValueFilter

Ищет в каждой строке референтный столбец, определяемый четырьмя обязательными аргументами: семейство столбцов, квалификатор столбца, оператор сравнения и компаратор. Если найден столбец с заданным семейством и квалификатором и его значение удовлетворяет условию сравнения — фильтр возвращает все пары ключ/значение из этой строки. Если найден столбец с заданным семейством и квалификатором, но его значение не удовлетворяет условию сравнения — никакие пары ключ/значение из этой строки не возвращаются. Если референтный столбец не найден — по умолчанию фильтр возвращает все пары ключ/значение из этой строки.

Можно дополнительно совместно задать два аргумента: если первый аргумент (`setFilterIfMissing`) имеет значение `true` (по умолчанию `false`), а референтный столбец не найден — фильтр не возвращает никакие пары ключ/значение из этой строки. Если второй аргумент (`setLatestVersionOnly`) имеет значение `false` (по умолчанию `true`) — проверяются все версии значения референтного столбца на соответствие условию сравнения, а не только последняя.

Принимает четыре или шесть аргументов: семейство столбцов (обязательно), квалификатор столбца (обязательно), оператор сравнения (обязательно), значение для сравнения (обязательно), флаги `setFilterIfMissing` и `setLatestVersionOnly` (необязательно, совместно).

Синтаксис:

```
"SingleColumnValueFilter ('location', 'town', =, 'binaryprefix:Al')"
```

```
"SingleColumnValueFilter ('location', 'town', =, 'binaryprefix:Al', true, false)"
```

Пример команды:

```
scan 'people', { FILTER => "SingleColumnValueFilter ('location', 'town', =, 'binaryprefix:Al', true, false)" }
```

Этот фильтр ищет в каждой строке значения, начинающиеся с `Al`, в столбце с точным именем `location:town`, проверяя все версии этих значений. Только в случае успеха он возвращает все пары ключ/значение из этой строки. Результат:

```
ROW                                         COLUMN+CELL
 Ball Nelle                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=43
 Ball Nelle                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Ball Nelle                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Ball Nelle                                 column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Bell Leila                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=58
 Bell Leila                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Bell Leila                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Bell Leila                                 column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Cohen John                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=28
 Cohen John                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Cohen John                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Cohen John                                 column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Diaz Anne                                  column=basic:age, timestamp=2024-07-30T08:10:19.297, value=57
 Diaz Anne                                  column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Diaz Anne                                  column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Diaz Anne                                  column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Howard Florence                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=37
 Howard Florence                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Howard Florence                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Howard Florence                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Ingram Barbara                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=55
 Ingram Barbara                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Ingram Barbara                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Ingram Barbara                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Jefferson Charlie                          column=basic:age, timestamp=2024-07-30T08:10:19.297, value=46
 Jefferson Charlie                          column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Jefferson Charlie                          column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Jefferson Charlie                          column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Kennedy Todd                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=44
 Kennedy Todd                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Kennedy Todd                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Kennedy Todd                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 McGee Isabelle                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=48
 McGee Isabelle                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 McGee Isabelle                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 McGee Isabelle                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Page Victoria                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=30
 Page Victoria                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Page Victoria                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Page Victoria                              column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Phelps Lida                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=43
 Phelps Lida                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Phelps Lida                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Phelps Lida                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Phillips Helen                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=61
 Phillips Helen                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Phillips Helen                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Phillips Helen                             column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Reyes Marc                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=25
 Reyes Marc                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Reyes Marc                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Reyes Marc                                 column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Roberts Clayton                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=52
 Roberts Clayton                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Roberts Clayton                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Roberts Clayton                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Ryan Curtis                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=58
 Ryan Curtis                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Ryan Curtis                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Ryan Curtis                                column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Spencer Lucinda                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Spencer Lucinda                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Spencer Lucinda                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Spencer Lucinda                            column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Woods Bessie                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=47
 Woods Bessie                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Woods Bessie                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Woods Bessie                               column=location:town, timestamp=2024-07-30T08:10:19.297, value=Albuquerque
 Yates Douglas                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=35
 Yates Douglas                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Yates Douglas                              column=location:state, timestamp=2024-07-30T08:10:20, value=NM
 Yates Douglas                              column=location:town, timestamp=2024-07-30T08:10:20, value=Albuquerque
18 row(s)
Took 0.0263 seconds
```

SingleColumnValueExcludeFilter

Работает так же, как и фильтр `SingleColumnValueFilter`, за исключением того, что значение референтного столбца никогда не возвращается.

Синтаксис:

```
"SingleColumnValueExcludeFilter ('location', 'town', =, 'binaryprefix:Al')"
```

```
"SingleColumnValueExcludeFilter ('location', 'town', =, 'binaryprefix:Al', true, false)"
```

Пример команды:

```
scan 'people', { FILTER => "SingleColumnValueExcludeFilter ('location', 'town', =, 'binaryprefix:Al', true, false)" }
```

Этот фильтр ищет в каждой строке значения, начинающиеся с `Al`, в столбце с точным именем `location:town`, проверяя все версии этих значений. Только в случае успеха он возвращает все пары ключ/значение из этой строки, кроме найденной. Результат:

```
ROW                                         COLUMN+CELL
 Ball Nelle                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=43
 Ball Nelle                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Ball Nelle                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Bell Leila                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=58
 Bell Leila                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Bell Leila                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Cohen John                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=28
 Cohen John                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Cohen John                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Diaz Anne                                  column=basic:age, timestamp=2024-07-30T08:10:19.297, value=57
 Diaz Anne                                  column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Diaz Anne                                  column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Howard Florence                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=37
 Howard Florence                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Howard Florence                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Ingram Barbara                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=55
 Ingram Barbara                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Ingram Barbara                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Jefferson Charlie                          column=basic:age, timestamp=2024-07-30T08:10:19.297, value=46
 Jefferson Charlie                          column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Jefferson Charlie                          column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Kennedy Todd                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=44
 Kennedy Todd                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Kennedy Todd                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 McGee Isabelle                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=48
 McGee Isabelle                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 McGee Isabelle                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Page Victoria                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=30
 Page Victoria                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Page Victoria                              column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Phelps Lida                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=43
 Phelps Lida                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Phelps Lida                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Phillips Helen                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=61
 Phillips Helen                             column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Phillips Helen                             column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Reyes Marc                                 column=basic:age, timestamp=2024-07-30T08:10:19.297, value=25
 Reyes Marc                                 column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Reyes Marc                                 column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Roberts Clayton                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=52
 Roberts Clayton                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Roberts Clayton                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Ryan Curtis                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=58
 Ryan Curtis                                column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Ryan Curtis                                column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Spencer Lucinda                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Spencer Lucinda                            column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Spencer Lucinda                            column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Woods Bessie                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=47
 Woods Bessie                               column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Woods Bessie                               column=location:state, timestamp=2024-07-30T08:10:19.297, value=NM
 Yates Douglas                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=35
 Yates Douglas                              column=location:country, timestamp=2024-07-30T08:10:19.297, value=USA
 Yates Douglas                              column=location:state, timestamp=2024-07-30T08:10:20, value=NM
18 row(s)
Took 0.0646 seconds
```

ColumnRangeFilter

Возвращает только те пары ключ/значение, у которых имя квалификатора столбца находится в диапазоне, заданном аргументами. Края диапазона могут быть определенными или пустыми. После каждого края задается логический аргумент, определяющий, включать ли значение края в диапазон.

Принимает четыре аргумента: левый край диапазона, флаг левого включения, правый край диапазона, флаг правого включения. Если край диапазона пуст, то значение соответствующего ему флага включения не имеет значения.

Синтаксис:

```
"ColumnRangeFilter ('', true, 'c', false)"
```

Пример команды:

```
scan 'people', { FILTER => "ColumnRangeFilter ('', true, 'c', false)" }
```

Этот фильтр вернет только столбец `basic:age`, поскольку следующий по алфавиту квалификатор столбца — это `country`, и он не попадает в заданный диапазон. Результат (первые и последние пять строк):

```
ROW                                         COLUMN+CELL
 Abbott Delia                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=62
 Abbott Howard                              column=basic:age, timestamp=2024-07-30T08:10:19.297, value=24
 Abbott Jack                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Adams Clyde                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Aguilar Myrtie                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=23
 ...
 Young Della                                column=basic:age, timestamp=2024-07-30T08:10:19.297, value=21
 Young Josephine                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=29
 Young Mattie                               column=basic:age, timestamp=2024-07-30T08:10:19.297, value=39
 Zimmerman Gene                             column=basic:age, timestamp=2024-07-30T08:10:19.297, value=35
 Zimmerman Madge                            column=basic:age, timestamp=2024-07-30T08:10:19.297, value=46
997 row(s)
Took 0.1788 seconds
```

Динамическая загрузка кастомных фильтров

HBase в составе ADH поддерживает динамическую загрузку кастомных фильтров. Для использования этой возможности следует задать директорию, содержащую JAR-файлы с кастомными фильтрами:

1. Откройте веб-интерфейс ADCM и выберите ваш кластер ADH.
2. Перейдите в раздел ***Services → HBase → Primary configuration*** и включите переключатель ***Show advanced***.
3. Откройте секцию настроек ***Custom hbase-site.xml*** и нажмите ***Add property***.
4. В поле имени параметра введите `hbase.dynamic.jars.dir`. В качестве значения параметра укажите путь на ваше усмотрение, например, `${hbase.rootdir}/lib`. Нажмите ***Apply***.
5. Сохраните конфигурацию, нажав ***Save → Create***, и перезапустите сервис, нажав ***Actions → Reconfig and graceful restart***.

При наличии JAR-файлов с кастомными фильтрами по заданному пути вы сможете использовать их как в консоли HBase shell, так и в Java-приложениях с использованием API HBase.

Использование сопроцессоров в HBase

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
  - Сопроцессор observer
  - Сопроцессор endpoint
- Загрузка и выгрузка сопроцессоров
  - Статическая загрузка сопроцессора
  - Статическая выгрузка сопроцессора
  - Динамическая загрузка сопроцессора
  - Динамическая выгрузка сопроцессора

Обзор

Сопроцессоры HBase — это механизм расширения (фреймворк), позволяющий выполнять распределенные вычисления с помощью запуска пользовательского кода непосредственно в месте размещения данных.

Сопроцессор observer

Сопроцессор observer (или **observer**) предназначен для создания хуков, позволяющих следить за табличными операциями с клиентской стороны. В числе таких операций `get`, `put`, `scan`, `delete` и другие. В HBase представлены три вида сопроцессоров observer:

- **RegionObserver** — предоставляет хуки для запросов обработки данных, таких как `get`, `put`, `scan`, `delete` и других. Экземпляр RegionObserver размещается на каждом регионе таблицы. Область действия каждого экземпляра ограничена его регионом.
- **WALObserver** — предоставляет хуки для операций write-ahead log (WAL). WALObserver может наблюдать или перехватывать события записи и реконструкции, выполняемые WAL. На каждом сервере регионов имеется один контекст обработки WAL с одним WALObserver.
- **MasterObserver** — предоставляет хуки для операций, связанных с описанием данных, таких как `create table`, `delete table`, `modify table` и других. MasterObserver располагается в контексте HBase master server.

Сопроцессор observer может наблюдать за действиями, выполняемыми клиентом в регионе, и вмешиваться в них. Следующие функции доступны для сопроцессоров observer.

Функции сопроцессоров observer

Функция	Описание
preGet, postGet	Вызываются до и после того как клиент выполнит запрос `get`, соответственно
preExists, postExists	Вызываются до и после того как клиент проверит существование данных с помощью запроса `get`, соответственно
prePut, postPut	Вызываются до и после того как клиент запишет данные с помощью запроса `put`, соответственно
preDelete, postDelete	Вызываются до и после того как клиент удалит данные с помощью запроса `delete`, соответственно
preOpen, postOpen	Вызываются до и после того как состояние региона online передается на HBase master server, соответственно
preFlush, postFlush	Вызываются до и после того как содержимое MemStore записывается в новый HFile, соответственно
preScannerOpen, postScannerOpen	Вызываются до и после того как клиент вызовет новый сканер с помощью запроса `scan`, соответственно
preScannerNext, postScannerNext	Вызываются до и после того как следующая строка будет запрошена клиентом в открытом сканере, соответственно
preScannerClose, postScannerClose	Вызываются до и после того как клиент закроет сканер, соответственно
preCheckAndPut, postCheckAndPut	Вызываются до и после того как клиент вызовет функцию `checkAndPut()`, соответственно
preCheckAndDelete, postCheckAndDelete	Вызываются до и после того как клиент вызовет функцию `checkAndDelete()`, соответственно

Ниже приведен код для простого сопроцессора observer. Он проверяет информацию о пользователе для запроса `get` с помощью инъекции кода в определенных хуках `preGet`. Он выдаст исключение CoprocessorException, если у пользователя нет прав доступа к ресурсу, что в свою очередь приведет к отказу в запросе пользователя.

```
package org.apache.hadoop.hbase.coprocessor;

import org.apache.hadoop.hbase.client.Get;

public class AccessControlCoprocessor extends BaseRegionObserverCoprocessor {

  public Get preGet(CoprocessorEnvironment e, Get get)
      throws CoprocessorException {

    if (access_not_allowed)  {
      throw new AccessDeniedException("User access denied");
    }
    return get;
  }
}
```

Сопроцессор endpoint

Сопроцессор endpoint (или **endpoint**) позволяет выполнять заданный код в регионе по триггеру. Например, можно выполнять агрегирование столбцов на любом сервере регионов.

Сопроцессор endpoint позволяет задавать кастомный динамический протокол RPC для коммуникации между клиентом и сервером регионов. Например, можно создать метод, который определяет специальные параметры запроса и типы ответов. Методы RPC, предоставляемые сопроцессором endpoint, могут выполняться при вызове динамических RPC-функций на стороне клиента.

Загрузка и выгрузка сопроцессоров

Для использования сопроцессора в HBase его сначала следует загрузить. Статическая загрузка выполняется через конфигурацию HBase, а динамическая — через консоль HBase shell или Java API.

Статическая загрузка сопроцессора

Чтобы статически загрузить сопроцессор, выполните следующее:

1. Поместите JAR-файл с кодом сопроцессора в директорию *lib/* инсталляции HBase на всех узлах кластера, где требуется использование сопроцессора.
2. Откройте веб-интерфейс ADCM и выберите ваш кластер ADH.
3. Перейдите в раздел ***Services*** → ***HBase*** → ***Primary configuration*** и включите переключатель ***Show advanced***.
4. Откройте секцию настроек ***Custom hbase-site.xml*** и нажмите ***Add property***.
5. Укажите одно из следующих имен:
   - `hbase.coprocessor.region.classes` — для сопроцессора RegionObserver или endpoint;
   - `hbase.coprocessor.wal.classes` — для сопроцессора WALObserver;
   - `hbase.coprocessor.master.classes` — для сопроцессора MasterObserver.
6. В качестве значения параметра укажите полное имя класса для класса реализации сопроцессора. Если таких классов несколько — их имена следует разделить запятыми.
7. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.

Статически загруженные сопроцессоры активны на всех регионах всех таблиц. Первый сопроцессор имеет приоритет `Coprocessor.Priority.SYSTEM`. Все остальные имеют целочисленные значения приоритета в порядке их объявления. Можно переопределить значение приоритета, добавив новое значение к имени класса, отделив его символом `|`. Пример: `org.myname.hbase.coprocessor.endpoint.SumEndPoint|21`.

Статическая выгрузка сопроцессора

Чтобы статически выгрузить сопроцессор, выполните следующее:

1. Удалите параметр из секции настроек ***Custom hbase-site.xml***, который был создан для сопроцессора.
2. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.
3. (Необязательно) Удалите JAR-файл из директории *lib/* инсталляции HBase.

Динамическая загрузка сопроцессора

Чтобы динамически загрузить сопроцессор, перейдите в консоль HBase shell и выполните команду следующего вида:

```
alter '<table_name>', METHOD => 'table_att', 'coprocessor'=>'hdfs://<namenode>:<port>/<coprocessor.jar>|<class_name>|<priority>|
<args>'
```

где:

- `<table_name>` — имя таблицы, для которой загружается сопроцессор;
- `<namenode>:<port>` — сетевой адрес узла, содержащего JAR-файл с кодом сопроцессора. Если файл расположен на этом же узле, эту часть можно опустить, обращая внимание на тройной символ `/`: `hdfs:///`;
- `<coprocessor.jar>` — полный путь к JAR-файлу в файловой системе узла;
- `<class_name>` — полное имя класса сопроцессора;
- `<priority>` — целочисленное значение приоритета сопроцессора;
- `<args>` — аргументы, принимаемые реализацией сопроцессора.

Пример:

```
alter 't1', METHOD => 'table_att', 'coprocessor'=>'hdfs://<namenode>:<port>/user/<hadoop-user>/coprocessor.jar|org.myname.hbase.
Coprocessor.RegionObserverExample|1037|arg1=1,arg2=2'
```

Чтобы убедиться, что сопроцессор загружен, используйте команду `describe`. Информация о сопроцессоре должна появиться в выводе команды сразу после имени таблицы.

Динамическая выгрузка сопроцессора

Чтобы динамически выгрузить сопроцессор, перейдите в консоль HBase shell и выполните команду следующего вида:

```
alter '<table_name>', METHOD => 'table_att_unset', NAME => 'coprocessor$1'
```

Сканирование снепшотов

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
  - Класс TableSnapshotScanner
  - Класс TableSnapshotInputFormat
- Конфигурация

Обзор

Сканирование снепшотов — это функция, позволяющая сканировать данные непосредственно из HFile без использования HBase. Она работает независимо от того, включен ли HBase, и помогает экономить ресурсы на серверной стороне HBase.

Сканирование снепшотов обеспечивается классами `TableSnapshotScanner` и `TableSnapshotInputFormat`.

Класс TableSnapshotScanner

Класс `TableSnapshotScanner` предназначен для однократного сканирования HFile, запускаемого с клиентской стороны. Эти HFile копируются во временную директорию и удаляются после закрытия сканера. Эта директория не должна принадлежать пути, заданному параметром `hbase.rootdir` в конфигурационном файле *hbase-site.xml*. Пользователь на клиентской стороне также должен иметь права на запись для этой директории.

Ниже приведен пример использования класса `TableSnapshotScanner`:

```
Path restoreDir = new Path("<path>");
Scan scan = new Scan();
try (TableSnapshotScanner scanner = new TableSnapshotScanner(conf, restoreDir, snapshotName, scan)) {
    Result result = scanner.next();
    while (result != null) {
        ...
        result = scanner.next();
    }
}
```

где `<path>` — путь ко временной директории.

Класс TableSnapshotInputFormat

Класс `TableSnapshotInputFormat` предназначен для сканирования HFile и использования результатов в задачах MapReduce.

Ниже приведен пример использования класса `TableSnapshotInputFormat`:

```
Job job = new Job(conf);
Path restoreDir = new Path(",<path>");
Scan scan = new Scan();
TableMapReduceUtil.initTableSnapshotMapperJob(snapshotName, scan, MyTableMapper.class, MyMapKeyOutput.class,
MyMapOutputValueWritable.class, job, true, restoreDir);
```

Конфигурация

Перед использованием функции сканирования снепшотов выполните следующую настройку:

1. Откройте веб-интерфейс ADCM и выберите ваш кластер ADH.
2. Перейдите в раздел ***Services*** → ***HDFS*** → ***Primary configuration*** и включите переключатель ***Show advanced***.
3. Откройте секцию настроек ***Custom hdfs-site.xml*** и нажмите ***Add property***.
4. В качестве названия параметра введите `dfs.namenode.acls.enabled` и установите для него значение `true`.
5. Откройте секцию настроек ***Custom core-site.xml*** и нажмите ***Add property***.
6. В качестве названия параметра введите `fs.permissions.umask-mode` и установите для него значение `027`.
7. Сохраните конфигурацию, нажав ***Save*** → ***Create***.
8. Перейдите в раздел ***Services*** → ***HBase*** → ***Primary configuration*** и включите переключатель ***Show advanced***.
9. Откройте секцию настроек ***Custom hbase-site.xml*** и нажмите ***Add property***.
10. В качестве названия параметра введите `hbase.coprocessor.master.classes` и установите для него значение `"org.apache.hadoop.hbase.security.access.AccessController,org.apache.hadoop.hbase.security.access.SnapshotScannerHDFSAclController"`.
11. Добавьте ещё один параметр. В качестве названия параметра введите `hbase.acl.sync.to.hdfs.enable` и установите для него значение `true`.
12. Сохраните конфигурацию, нажав ***Save*** → ***Create*** и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.
13. Перейдите в консоль HBase shell и измените схему нужной таблицы для использования функции сканирования снепшотов, выполнив команду следующего вида:

```
alter '<table>', CONFIGURATION => {'hbase.acl.sync.to.hdfs.enable' => 'true'}
```

где `<table>` — название таблицы.

Использование объектов MOB в HBase

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
  - Введение
  - Архитектура
  - Обслуживание
  - Сжатие
- Использование
  - Конфигурация столбцов MOB
  - Конфигурация политики сжатия
  - Конфигурация политики архивации
- Оптимизация
  - Конфигурация кеша MOB
  - Ручное сжатие

Обзор

Введение

Объекты среднего размера (Medium-sized object, MOB) — это тип данных, хранящихся в таблицах HBase, с порядком размера от 100 КБ до 10 МБ и более. Примерами таких объектов являются файлы изображений, документов, архивов и другие бинарные файлы.

Технически, HBase способен работать со значениями любых размеров, но объекты крупнее 100 КБ могут и будут отрицательно влиять на производительность во время записи данных из MemStore в HFile: разделения регионов (region split) и сжатия (HFile compaction) неизбежно приведут к дублированию информации (write amplification). Чтобы избежать этого, можно определить семейства столбцов, содержащие крупные объекты в своих ячейках, как столбцы MOB. Ячейки, которые должны хранить MOB, будут хранить лишь указатели на настоящие файлы. Эти файлы, в свою очередь, хранятся в специальных MOB HFile, которые могут располагаться в любом заданном месте (HDFS или нет).

Ячейки обычных столбцов хранят данные как есть, независимо от их размера. Ячейки столбцов MOB различают данные по размеру согласно заданному порогу: данные меньшего размера хранятся как есть, данные большего размера хранятся как MOB.

Архитектура

Изначально все данные, записываеме в HBase впервые, хранятся в WAL и MemStore. Когда MemStore, соответствующий семейству столбцов MOB, наполняется — он создает два HFile. Значения ячеек, меньшие заданного порога, записываются в обычный HFile. Значения ячеек большего или равного размера записываются в специальный MOB HFile, и одновременно с этим указатели на расположение этих значений записываются в соответствующие ячейки (ячейки указателей MOB) обычного HFile. После окончания записи из MemStore и закрытия соответствующего обычного HFile сервер регионов добавляет к нему метаданные. Эти метаданные содержат список всех MOB HFile, на которые имеются указатели в обычном HFile.

Значение ячейки указателя MOB состоит из двух частей: размер MOB и имя соответствующего MOB HFile. Она также имеет две метки: первая сообщает о том, что ячейка содержит указатель MOB, а вторая содержит пространство имен и название таблицы для MOB HFile. Эти метки могут помочь при поиске ячеек указателей MOB и для оптимизации способа обнаружения значений в MOB HFile, особенно после нескольких процедур снятия снепшота HBase. Эти метки по умолчанию не пересылаются с помощью RPC, поскольку они доступны лишь со стороны серверов HBase.

После создания или сжатия MOB HFile размещаются в специальном месте внутри корневой директории HBase в соответствии с их пространством имен, названием таблицы, логическим регионом MOB и названием семейства столбцов. Пример такого пути: */hbase/mobdir/data/default/table_with_mobs/463b0c18d2eb1a65b4b940837d6d0a08/mob_column/*, где:

- `hbase` — корневая директория HBase;
- `mobdir/data` — подкаталог по умолчанию для MOB HFile;
- `default` — название пространства имен;
- `table_with_mobs` — имя таблицы;
- `463b0c18d2eb1a65b4b940837d6d0a08` — идентификатор логического региона;
- `mob_column` — имя семейства столбцов.

Обслуживание

Обслуживание MOB HFile выполняется регулярными рутинами (chores) на master-сервере HBase и серверах регионов. Они отвечают за соблюдение срока жизни (TTL) семейств столбцов и за сжатие HFile. Когда MOB HFile становится неактуальным после процедуры сжатия, рутина перемещает его в архив точно так же, как это происходит с обычными HFile. Регионы MOB независимы от обычных регионов, поэтому они используют то же расположение хранения архива. Пример пути к такому расположению: */hbase/archive/data/default/table_with_mobs/463b0c18d2eb1a65b4b940837d6d0a08/mob_column/*.

Если существует снепшот таблицы, для которой включено использование MOB (или клон такого снепшота), то MOB HFile будут оставаться в архиве столько, сколько от них потребуют эти снепшоты. Рутины очистки удаляют MOB HFile из архива когда необходимость в них исчезает.

Сжатие

Во время записи данных из MemStore, связанного с семейством столбцов MOB, HBase записывает значения размером выше порога в MOB HFile. Во время процедуры сжатия региона сервер регионов перезаписывает данные в обычных HFile, но оставляет указатели на MOB HFile без изменения. Клиент может осуществлять прозрачный доступ к исходным данным по указателю, поскольку внутренние рутины сервера регионов отслеживают эти данные и извлекают значения из MOB HFile. Это позволяет накапливать большое количество MOB HFile без снижения эффективности извлечения значений из ячеек MOB. Сжатие MOB HFile требуется гораздо реже.

Если значения в ячейках MOB часто подвергаются изменениям или удалению, то это может привести к избыточному расходу пространства. Чтобы удалить MOB HFile и освободить пространство, необходимо убедиться, что никакие ячейки не содержат указателей на него, а его текущие значения должны быть записаны в новый MOB HFile. Также, если файловая система накладывает ограничение на максимальное количество содержащихся в ней файлов (например, HDFS), то даже если значения в ячейках MOB не подвергаются изменениям или удалению вовсе, MOB HFile будут накапливаться и потребуют сжатия.

Использование

Конфигурация столбцов MOB

В командной оболочке HBase shell за обозначение семейства столбцов MOB отвечают два параметра:

- `IS_MOB` — логический параметр, определяющий, могут ли значения в ячейках столбцов являться MOB.
- `MOB_THRESHOLD` — целочисленный параметр, указывающий количество байт, начиная с которого значение считается MOB.

Параметр `MOB_THRESHOLD` является необязательным. Если его не задавать — используется значение по умолчанию `102400` (100 КБ).

Пример команды в HBase shell для создания новой таблицы с семейством столбцов, предназначенным для хранения MOB:

```
create 'table_with_mobs', {NAME => 'mob_column', IS_MOB => true, MOB_THRESHOLD => 204800}
```

В этом примере создается таблица `table_with_mobs` с семейством столбцов `mob_column` для хранения MOB. Все значения размером более 200 КБ в этом семействе столбцов будут обрабатываться как MOB.

Пример команды в HBase shell для добавления нового семейства столбцов, предназначенного для хранения MOB:

```
alter 'table_with_mobs', {NAME => 'anoter_mob_column', IS_MOB => true, MOB_THRESHOLD => 153600}
```

В этом примере к таблице `table_with_mobs` добавляется семейство столбцов `anoter_mob_column`. Оно также предназначено для хранения MOB, а в качестве порога устанавливается размер 150 КБ.

Пример фрагмента кода для Java API, задающего семейство столбцов MOB:

```
HColumnDescriptor mob_column = new HColumnDescriptor("f");
mob_column.setMobEnabled(true);
mob_column.setMobThreshold(102400L);
```

В этом примере создается семейство столбцов `mob_column` для хранения MOB с порогом по умолчанию 100 КБ.

Конфигурация политики сжатия

Специальная рутина на master-сервере HBase периодически запускает процедуру полного сжатия как для обычных HFile, так и для MOB HFile. По умолчанию она выполняется раз в неделю, но это расписание можно изменить:

1. Откройте веб-интерфейс ADCM и выберите ваш кластер ADH.
2. Перейдите в раздел ***Services*** → ***HBase*** → ***Primary configuration*** и активируйте переключатель ***Show advanced***.
3. Откройте секцию настроек ***Custom hbase-site.xml*** и нажмите ***Add property***.
4. В поле имени параметра введите `hbase.mob.compaction.chore.period`. В качестве значения параметра укажите периодичность в секундах. Например, сутки — `86400`, месяц — `2592000`. Нажмите ***Apply***.
5. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.

По умолчанию эта рутина указывает всем регионам выполнять сжатие одновременно и максимально загружает кластер. Можно сбалансировать нагрузку, установив ограничение на количество одновременно выполняемых сжатий в разных регионах:

1. Откройте веб-интерфейс ADCM и выберите ваш кластер ADH.
2. Перейдите в раздел ***Services*** → ***HBase*** → ***Primary configuration*** и активируйте переключатель ***Show advanced***.
3. Откройте секцию настроек ***Custom hbase-site.xml*** и нажмите ***Add property***.
4. В поле имени параметра введите `hbase.mob.major.compaction.region.batch.size`. В качестве значения параметра укажите целочисленное ограничение. Ноль означает поведение рутины по умолчанию (все регионы работают одновременно). Нажмите ***Apply***.
5. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.

Конфигурация политики архивации

Специальная рутина на master-сервере HBase периодически запускает процедуру архивации для MOB HFile, когда необходимость в них исчезает. MOB HFile отправляется в архив в двух случаях:

- он старше времени жизни своего семейства столбцов;
- нет обычных HFile ни в одном регионе, которые бы содержали указатели на него.

По умолчанию эта рутина выполняется раз в день. Этого достаточно в большинстве случаев. Однако если семейства столбцов MOB обновляются достаточно часто или имеют короткое время жизни, что приводит к частым сжатиям, может потребоваться изменение расписания этой рутины:

1. Откройте веб-интерфейс ADCM и выберите ваш кластер ADH.
2. Перейдите в раздел ***Services*** → ***HBase*** → ***Primary configuration*** и активируйте переключатель ***Show advanced***.
3. Откройте секцию настроек ***Custom hbase-site.xml*** и нажмите ***Add property***.
4. В поле имени параметра введите `hbase.master.mob.cleaner.period`. В качестве значения параметра укажите периодичность в секундах. Нажмите ***Apply***.
5. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.

Оптимизация

Конфигурация кеша MOB

Количество MOB HFile может быть велико по сравнению с обычными HFile, и поэтому они не всегда бывают открыты для чтения. Кеш открытых MOB HFile держит открытыми те из них, доступ к которым выполнялся позднее всего (кеш LRU). Можно настроить следующие параметры:

- **Размер кеша** — ограничение на количество кешируемых обработчиков открытых файлов. Большой размер кеша может улучшить производительность, уменьшая количество повторных открытий и закрытий файлов. Слишком большой размер кеша может привести к ошибке.
- **Период вытеснения** — тайм-аут для неиспользуемого открытого файла, после которого он закрывается и удаляется из кеша.
- **Фактор вытеснения** — коэффициент от 0 до 1, определяющий, сколько файлов должно остаться в кеше при превышении его размера. При достижении ограничения на размер кеша общее количество обработчиков в нем умножается на этот коэффициент, и полученное количество наиболее используемых остается в кеше, а остальные вытесняются.

Чтобы добавить эти параметры, выполните следующее:

1. Откройте веб-интерфейс ADCM и выберите ваш кластер ADH.
2. Перейдите в раздел ***Services*** → ***HBase*** → ***Primary configuration*** и активируйте переключатель ***Show advanced***.
3. Откройте секцию настроек ***Custom hbase-site.xml*** и нажмите ***Add property***.
   Для задания одного или нескольких требуемых параметров используйте следующие значения:
   - Размер кеша — в поле имени параметра введите `hbase.mob.file.cache.size`. В качестве значения параметра укажите целочисленное ограничение. Значение по умолчанию — `1000`.
   - Период вытеснения — в поле имени параметра введите `hbase.mob.cache.evict.period`. В качестве значения параметра укажите целочисленное значение в секундах. Значение по умолчанию — `3600` (один час).
   - Фактор вытеснения — в поле имени параметра введите `hbase.mob.cache.evict.remain.ratio`. В качестве значения параметра укажите дробный коэффициент от `0.0f` до `1.0f`. Значение по умолчанию — `0.5f`.
4. Нажмите ***Apply***.
5. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.

Ручное сжатие

Чтобы не ждать, пока рутина сжатия запустится по расписанию, можно запустить процедуру сжатия вручную. Перейдите в командную оболочку HBase shell и используйте одну из команд следующего вида:

```
major_compact 'table_with_mobs'
major_compact 'table_with_mobs', 'mob_column'
```

Первая команда подвергнет процедуре сжатия все семейства столбцов указанной таблицы. Вторая запустит процедуру только для указанного семейства столбцов.

Пример того же запроса в Java API:

```
Admin.majorCompact(table_with_mobs, mob_column);
```

Массовая загрузка данных в HBase

Альберт Багдасарян

Назад к содержанию

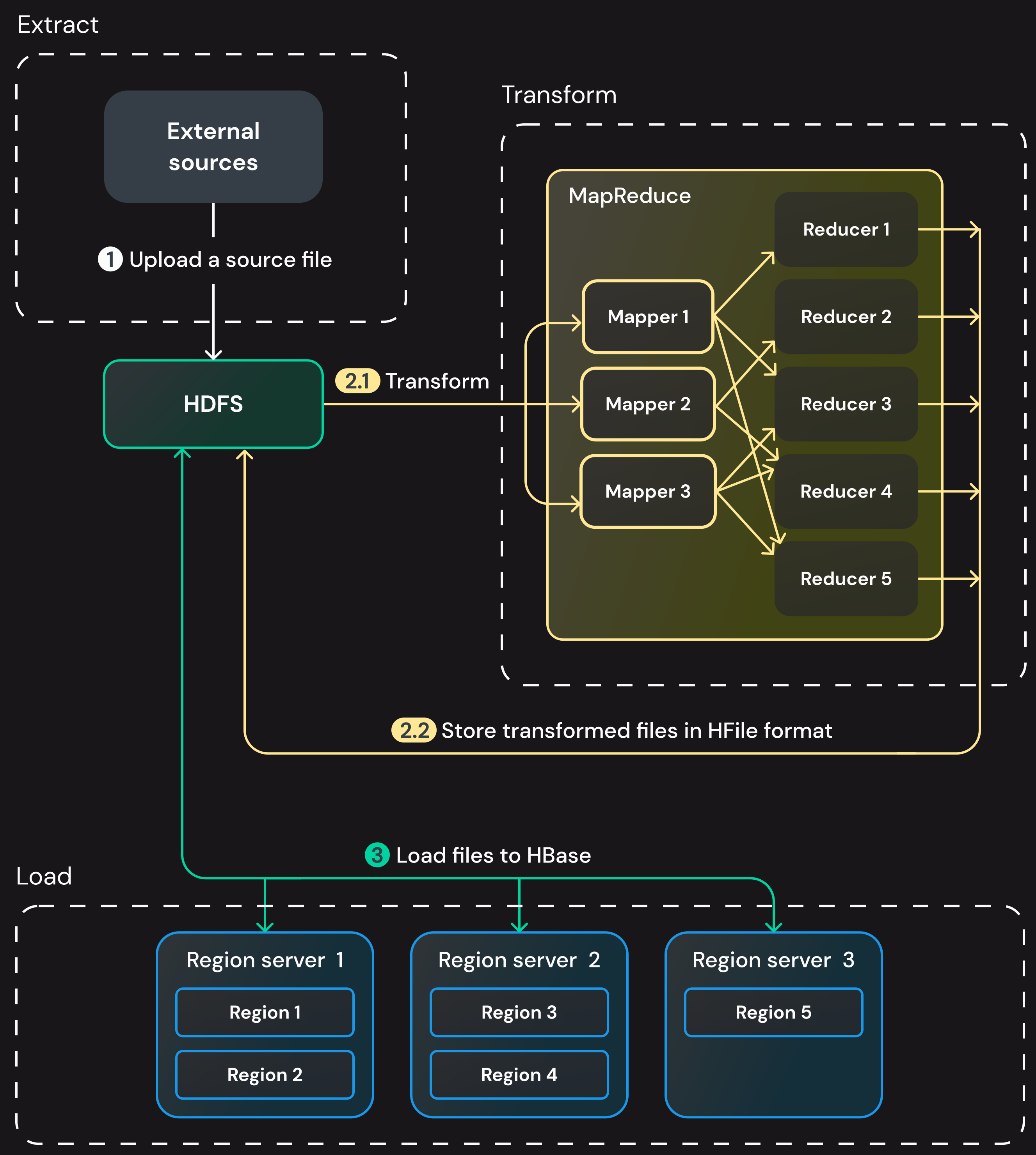
В HBase типовая последовательность записи данных включает следующие шаги: *WAL → MemStore → HFiles*. Это работает хорошо в обычном режиме, но при загрузке очень большого объема данных такая последовательность ведет к значительному снижению производительности. К основным проблемами относятся следующие:

- Повышенные требования к объему области памяти MemStore и журнала WAL.
- Увеличение очередей на операции сжатия (compaction) и очистки (flush).
- Как следствие, увеличенная задержка, приводящая к нарушению договоренностей о качестве обслуживания (SLA).

Массовая загрузка (bulk loading) направлена на решение этих проблем. Она представляет собой процесс подготовки и загрузки данных напрямую в файлы HFiles в системе HDFS, избегая стандартные стадии записи. Такой способ позволяет вносить в базу большие объемы данных гораздо быстрее. Он может быть полезен при загрузке данных из внешних источников или для последовательной загрузки партий данных, образуемых в процессе выполнения длительных автоматических ночных задач.

Массовая загрузка — это процесс ETL, включающий следующие три основные стадии:

1. **Извлечение (Extract)**. На этой стадии процесс извлекает данные из внешнего источника и сохраняет его в HDFS. Источниками могут быть содержимое другой базы данных (database dump) или файлы журналов в формате простого текста, CSV или другого формата. HBase не управляет этой частью процесса.
2. **Преобразование (Transform)**. На этой стадии процесс преобразует данные в формат HFile в соответствие со структурой предварительно созданной таблицы HBase: в каждом регионе он создает один HFile. Данные отсортированы в алфавитном порядке и разделены по файлам с помощью заданных точек деления (split points). Этот шаг обычно требует запуска задания MapReduce, которое и выполняет разделение данных на партиции, сортировку, генерацию класса вида ключ/значение, обеспечение требований к формату HFile и другие операции.
3. **Загрузка (Load)**. На этой стадии процесс перемещает данные в таблицу HBase, обеспечивая загрузку каждого файла в соответствующий регион с помощью Region Server, который обслуживает этот регион.


Extract
External sources
① Upload a source file
HDFS
2.1 Transform
Transform
MapReduce
Mapper 1
Mapper 2
Mapper 3
Reducer 1
Reducer 2
Reducer 3
Reducer 4
Reducer 5
2.2 Store transformed files in HFile format
③ Load files to HBase
Load
Region server 1
Region 1
Region 2
Region server 2
Region 3
Region 4
Region server 3
Region 5

Типовая схема массовой загрузки данных

Существует много способов настройки массовой загрузки данных в HBase, включая использование встроенных задач MapReduce, собственных (custom) задач MapReduce, Spark и других.

Массовая загрузка с использованием встроенных задач MapReduce

Альберт Багдасарян

Содержание

Назад к содержанию

Шаг 1. Извлечение данных

Шаг 2. Преобразование

Шаг 3. Загрузка

В статье рассмотрен простейший способ массовой загрузки данных (bulk loading) в HBase — на основе встроенных утилит ImportTsv и LoadIncrementalHFiles (известной также как CompleteBulkLoad).

Шаг 1. Извлечение данных

На первом шаге необходимо подготовить образец данных и загрузить его в HDFS. Для примера используйте тестовый файл people_ages.csv. Этот файл содержит сгенерированные случайным образом имена и возрасты одной тысячи людей. Обратите внимание, что три имени дублируются — каждое из имен `McGee Isabelle`, `Reid Janie` и `Summers Blanche` встречаются дважды. Дубликаты добавлены, чтобы продемонстрировать поведение системы в таком случае.

Выполните следующие операции:

1. Загрузите файл в свой домашний каталог на один из серверов кластера HBase. Убедитесь, что файл загружен, с помощью следующей команды:

```
$ ls -la ~
```

Вывод на экран должен содержать название загруженного файла:

```
total 56
drwx------. 5 dasha dasha   251 Nov 26 07:46 .
drwxr-xr-x. 3 root  root     19 Aug 31 11:54 ..
drwx------. 3 dasha dasha    17 Aug 31 15:06 .ansible
-rw-------. 1 dasha dasha 14913 Nov 25 16:07 .bash_history
-rw-r--r--. 1 dasha dasha    18 Apr  1  2020 .bash_logout
-rw-r--r--. 1 dasha dasha   193 Apr  1  2020 .bash_profile
-rw-r--r--. 1 dasha dasha   231 Apr  1  2020 .bashrc
drwxrwxrwx. 2 dasha dasha    64 Nov 26 07:45 dasha
-rw-rw-r--. 1 dasha dasha 17651 Nov 26 07:46 people_ages.csv
drwx------. 2 dasha dasha    29 Sep 23 17:35 .ssh
```

2. Создайте в HDFS каталог для своего пользователя, если такого каталога еще нет:

```
$ sudo -u hdfs hdfs dfs -mkdir /user/dasha
$ hdfs dfs -ls /user
```

Пример вывода на экран:

```
Found 5 items
drwxr-xr-x   - hdfs   hadoop          0 2021-11-26 09:56 /user/dasha
drwx------   - hdfs   hadoop          0 2021-08-31 16:15 /user/hdfs
drwxr-xr-x   - mapred hadoop          0 2021-08-31 16:22 /user/history
drwxr-xr-x   - mapred mapred          0 2021-08-31 16:21 /user/mapred
drwxr-xr-x   - yarn   yarn            0 2021-09-01 06:57 /user/yarn
```

3. Расширьте права доступа на ваш каталог с помощью следующей команды (более подробная информация о защите файлов приведена в разделе Защита файлов в HDFS):

```
$ sudo -u hdfs hdfs dfs -chmod 777 /user/dasha
$ hdfs dfs -ls /user
```

Пример вывода на экран:

```
Found 5 items
drwxrwxrwx   - hdfs   hadoop          0 2021-11-26 09:57 /user/dasha
drwx------   - hdfs   hadoop          0 2021-08-31 16:15 /user/hdfs
drwxr-xr-x   - mapred hadoop          0 2021-08-31 16:22 /user/history
drwxr-xr-x   - mapred mapred          0 2021-08-31 16:21 /user/mapred
drwxr-xr-x   - yarn   yarn            0 2021-09-01 06:57 /user/yarn
```

4. Скопируйте этот файл из локальной файловой системы в HDFS:

```
$ hdfs dfs -copyFromLocal ~/people_ages.csv /user/dasha
```

5. Убедитесь, что файл появился в HDFS:

```
$ hdfs dfs -ls /user/dasha
```

Пример вывода на экран:

```
Found 1 items
-rw-r--r--   3 dasha hadoop      17651 2021-11-26 09:57 /user/dasha/people_ages.csv
```

Шаг 2. Преобразование

На этом шаге необходимо создать таблицу в HBase и затем применить встроенное задание MapReduce под названием ImportTsv для преобразования загруженного файла в множество файлов формата HFile согласно структуре предварительно созданной таблицы:

1. Перейдите в интерактивный режим HBase shell:

```
$ hbase shell
```

2. Создайте таблицу `people_ages` с одним семейством колонок `basic`. Задайте четыре точки разделения данных (splits) по ключам строк на пять регионов:

```
create 'people_ages', {NAME => 'basic', VERSIONS => 5}, {SPLITS => ['F', 'K', 'P', 'W']}
```

Вывод на экран:

```
Created table people_ages
Took 1.6675 seconds
=> Hbase::Table - people_ages
```

3. Выйдите из HBase shell и запустите задание MapReduce ImportTsv:

```
$ hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator=',' -Dimporttsv.columns=HBASE_ROW_KEY,basic:age -
Dimporttsv.bulk.output=test_output people_ages /user/dasha/people_ages.csv
```

Аргументы этой утилиты поясняет следующая таблица.

Аргументы

Dimporttsv.separator	Разделитель данных в исходных данных. В этом примере им является запятая
Dimporttsv.columns	Полный перечень колонок, содержащихся в импортированном тексте. В нашем примере использование `HBASE_ROW_KEY,basic:age` означает, что первая колонка исходного текста состоит из уникальных ключей строк, а вторая содержит значения колонки `basic:age`
Dimporttsv.bulk.output	Относительный путь к каталогу HDFS, в котором требуется создать файлы в формате HFile. В нашем примере, использование `test_output` означает полный путь */user/dasha/test_output*

Последние позиции в перечне аргументов команды содержат название таблицы (`people_ages`) и путь к исходному файлу в HDFS (`/user/dasha/people_ages.csv`).

РЕКОМЕНДАЦИЯ

Входной аргумент `Dimporttsv.bulk.output` не является обязательным. Если вы не используете его, то HBase загружает данные, следуя типовой последовательности записи. Таким образом, вы можете использовать задание MapReduce ImportTsv просто для чтения файлов CSV (без выполнения последующих шагов). Однако, если необходимо загрузить данные в HFiles, надо задать этот аргумент.

4. Проверьте содержимое каталога *test_output*:

```
$ hdfs dfs -ls /user/dasha/test_output
```

Он содержит папку *basic*, названную так согласно названию семейства колонок в таблице:

```
Found 2 items
-rw-r--r--   3 dasha hadoop          0 2021-11-26 09:59 /user/dasha/test_output/_SUCCESS
drwxr-xr-x   - dasha hadoop          0 2021-11-26 09:59 /user/dasha/test_output/basic
```

5. Проверьте содержимое папки *basic*:

```
$ hdfs dfs -ls /user/dasha/test_output/basic
```

Она содержит пять файлов формата HFile — один файл на регион. Эти файлы, созданные утилитой ImportTsv, содержат данные из исходного файла, отсортированные в алфавитном порядке и сохраненные в формате HFile:

```
Found 5 items
-rw-r--r--   3 dasha hadoop      16383 2021-11-26 09:59 /user/dasha/test_output/basic/0c17ccdbda7a465281ee063fa806ee44
-rw-r--r--   3 dasha hadoop      15511 2021-11-26 09:59 /user/dasha/test_output/basic/13d31777206148dab3fbf3bf3c2c3ad0
-rw-r--r--   3 dasha hadoop      14111 2021-11-26 09:59 /user/dasha/test_output/basic/5018aca2586840b5a983eff7a9b58e3f
-rw-r--r--   3 dasha hadoop      17110 2021-11-26 09:59 /user/dasha/test_output/basic/c5f4176da2764378826652cb24b6cb05
-rw-r--r--   3 dasha hadoop       9313 2021-11-26 09:59 /user/dasha/test_output/basic/de11b176c73a4aa29133c0064c1e5d52
```

ПРИМЕЧАНИЕ

На этом шаге данные не загружены в таблицу. Если в среде HBase shell вы исполните команду `scan` по отношению к данной таблице, она вернет пустой результат.

Шаг 3. Загрузка

На этом шаге необходимо загрузить подготовленные HFiles в таблицу HBase с помощью встроенной утилиты LoadIncrementalHFiles:

1. Предоставьте HBase право на перемещение подготовленных файлов. Для этого замените их владельца на `hbase`:

```
$ sudo -u hdfs hdfs dfs -chown -R hbase:hbase /user/dasha/test_output
```

Убедитесь, что владелец каталога *test_output* и всех файлов в нем изменен:

```
$ hdfs dfs -ls /user/dasha/
```

Пример вывода на экран:

```
Found 4 items
drwx------   - dasha hadoop          0 2021-11-26 09:59 /user/dasha/.staging
drwxrwxrwx   - dasha hadoop          0 2021-11-26 09:59 /user/dasha/hbase-staging
-rw-r--r--   3 dasha hadoop      17651 2021-11-26 09:57 /user/dasha/people_ages.csv
drwxr-xr-x   - hbase hbase           0 2021-11-26 09:59 /user/dasha/test_output
```

Владелец папки изменен, как и запланировано. Теперь проверим владельца файлов в этой папке:

```
$ hdfs dfs -ls /user/dasha/test_output/basic
```

Пример вывода на экран:

```
Found 5 items
-rw-r--r--   3 hbase hbase      16383 2021-11-26 09:59 /user/dasha/test_output/basic/0c17ccdbda7a465281ee063fa806ee44
-rw-r--r--   3 hbase hbase      15511 2021-11-26 09:59 /user/dasha/test_output/basic/13d31777206148dab3fbf3bf3c2c3ad0
-rw-r--r--   3 hbase hbase      14111 2021-11-26 09:59 /user/dasha/test_output/basic/5018aca2586840b5a983eff7a9b58e3f
-rw-r--r--   3 hbase hbase      17110 2021-11-26 09:59 /user/dasha/test_output/basic/c5f4176da2764378826652cb24b6cb05
-rw-r--r--   3 hbase hbase       9313 2021-11-26 09:59 /user/dasha/test_output/basic/de11b176c73a4aa29133c0064c1e5d52
```

2. Запустите утилиту LoadIncrementalHFiles с правами пользователя `hbase`. Аргументами должны быть путь к созданному каталогу (`test_output`) и название таблицы (`people_ages`):

```
$ sudo -u hbase hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles /user/dasha/test_output people_ages
```

3. В среде HBase shell определите количество строк в таблице `people_ages`:

```
count "people_ages"
```

Как видите, в таблице только 997 строк вместо 1000 строк в исходном файле. Причиной этого является наличие трех дублированных имен, как мы обсуждали ранее.

```
997 row(s)
Took 0.3057 seconds
=> 997
```

ВНИМАНИЕ

Будьте внимательны к выбору поля исходного файла для формирования ключей строк. Значения в этом поле должны быть уникальными. Дублирование приведет к потере строк, несмотря на поддержку нескольких версий в семействе колонок.

4. Просканируйте таблицу для получения ее содержимого:

```
scan 'people_ages'
```

Первые десять строк, выведенные командой, приведены ниже:

```
ROW                            COLUMN+CELL
 Abbott Delia                  column=basic:age, timestamp=1637918350575, value=62
 Abbott Howard                 column=basic:age, timestamp=1637918350575, value=24
 Abbott Jack                   column=basic:age, timestamp=1637918350575, value=29
 Adams Clyde                   column=basic:age, timestamp=1637918350575, value=29
 Aguilar Myrtie                column=basic:age, timestamp=1637918350575, value=23
 Aguilar Terry                 column=basic:age, timestamp=1637918350575, value=65
 Alexander Derrick             column=basic:age, timestamp=1637918350575, value=46
 Alexander Gregory             column=basic:age, timestamp=1637918350575, value=54
 Alexander Leon                column=basic:age, timestamp=1637918350575, value=42
 Allen Austin                  column=basic:age, timestamp=1637918350575, value=34
```

5. Проверьте распределение данных таблицы по пяти регионам, выполнив следующую команду:

```
list_regions 'people_ages'
```

Пример вывода на экран:

```
                                         SERVER_NAME |                                                    REGION_NAME |  START_KEY
|    END_KEY |  SIZE |   REQ |   LOCALITY |
 ----------------------------------------------- | ------------------------------------------------------------ | ----------
| ---------- | ----- | ----- | ---------- |
 bds-adh-1.ru-central1.internal,16020,1637912629824 |  people_ages,,1637920713477.bf509f13aa0dd59a5f4e9bcfc6cfbfab. |            |
F |     0 |   953 |        1.0 |
 bds-adh-3.ru-central1.internal,16020,1637912629738 | people_ages,F,1637920713477.b6693f3894a0feddba80ff15c06e2fbf. |          F |
K |     0 |   873 |        1.0 |
 bds-adh-3.ru-central1.internal,16020,1637912629738 | people_ages,K,1637920713477.50dcc6b90c412dab1f830a1b963d21e1. |          K |
P |     0 |   781 |        1.0 |
 bds-adh-2.ru-central1.internal,16020,1637912628845 | people_ages,P,1637920713477.80f4f1f471c9129f00d8f73701159145. |          P |
W |     0 |  1019 |        1.0 |
 bds-adh-2.ru-central1.internal,16020,1637912628845 | people_ages,W,1637920713477.ef6dfb19328842ce5c4ce883378329b0. |          W |
|     0 |   369 |        1.0 |
 5 rows
Took 0.2080 seconds
hbase(main):012:0> 'people_ages'
=> "people_ages"
```

6. Для проверки наличия HFiles в HBase выйдите из HBase shell и запустите следующую команду:

```
$ hdfs dfs -ls /hbase/data/default/people_ages
```

Вывод на экран содержит пять папок, содержащих файлы в формате HFile, и две служебные папки:

```
Found 7 items
drwxr-xr-x   - hbase hbase          0 2021-11-26 09:58 /hbase/data/default/people_ages/.tabledesc
drwxr-xr-x   - hbase hbase          0 2021-11-26 09:58 /hbase/data/default/people_ages/.tmp
drwxr-xr-x   - hbase hbase          0 2021-11-26 09:58 /hbase/data/default/people_ages/50dcc6b90c412dab1f830a1b963d21e1
drwxr-xr-x   - hbase hbase          0 2021-11-26 09:58 /hbase/data/default/people_ages/80f4f1f471c9129f00d8f73701159145
drwxr-xr-x   - hbase hbase          0 2021-11-26 09:58 /hbase/data/default/people_ages/b6693f3894a0feddba80ff15c06e2fbf
drwxr-xr-x   - hbase hbase          0 2021-11-26 09:58 /hbase/data/default/people_ages/bf509f13aa0dd59a5f4e9bcfc6cfbfab
drwxr-xr-x   - hbase hbase          0 2021-11-26 09:58 /hbase/data/default/people_ages/ef6dfb19328842ce5c4ce883378329b0
```

Все данные HBase хранятся в каталоге HDFS */hbase/data/*. Путь к файлу HFile содержит название пространства (namespace), в данном случае — `default`, и название таблицы (`people_ages`).

Использование снепшотов для резервного копирования и восстановления данных в HBase

Владимир Адаменков

Содержание

Назад к содержанию

Обзор

Использование

- Создание снепшота
- Восстановление данных из снепшота
- Экспорт снепшота
- Ротация снепшотов

Обзор

Снепшоты в HBase — это функциональность, позволяющая делать резервное копирование данных с минимально возможным влиянием на производительность HBase. При создании снепшота таблицы не выполняется копирование данных, а сама таблица остается доступной для операций с данными. Экспорт снепшота в другой кластер выполняется без участия серверов регионов и Master-сервера с помощью утилиты, основанной на distcp. При необходимости таблица может быть быстро восстановлена из снепшота.

Снепшот по сути представляет собой набор метаданных: схема таблицы и указатели на HFile, содержащие данные таблицы. HFile после создания не может быть изменен никаким образом, кроме удаления. HFile, содержащие данные таблицы, становятся связанными со снепшотом в момент его создания. Если HFile, связанный со снепшотом, подвергается сжатию, то вместо удаления он помещается в архив. HFile, связанные со снепшотом, существуют все время существования этого снепшота для возможности извлечения информации и восстановления соответствующей таблицы. Тот же принцип относится к клонам снепшотов, которые являются таблицами, восстановленными из снепшота, но независимыми от оригинальных таблиц: пока существует клон — связанные с его снепшотом HFile также существуют. Поэтому необходимо использование политики ротации для снепшотов и их клонов во избежание исчерпания пространства на дисках.

ПРИМЕЧАНИЕ

Более подробную информацию о командах HBase shell, используемых для операций со снепшотами, можно получить в разделе Команды HBase shell → Snapshots commands.

Использование

Для показанных ниже примеров будет использоваться небольшая тестовая таблица `articles`, процесс создания которой описан в статье Начало работы с HBase shell. Выполните Шаг 1 и Шаг 3 из неё, если вы хотите попробовать выполнить эти примеры самостоятельно.

Приведенные примеры соответствуют следующему жизненному циклу снепшота:

1. Создание снепшота и проверка его существования.
2. Восстановление таблицы из снепшота после внесения в неё изменений.
3. Экспорт снепшота в другой кластер HBase и восстановление таблицы из него.
4. Удаление снепшота.

Все команды выполняются в консоли HBase shell, если не указано иное.

Создание снепшота

Используйте команду snapshot, чтобы создать снепшот таблицы. Пример:

```
snapshot 'articles', 'articles_snp'
```

Используйте команду list_snapshots, если вы хотите просмотреть все снепшоты, или команду list_table_snapshots, если вы хотите просмотреть снепшоты заданной таблицы. Пример:

```
list_table_snapshots 'articles'
```

Вывод команды будет выглядеть следующим образом:

```
SNAPSHOT                                     TABLE + CREATION TIME
 articles_snp                                articles (2024-11-01 08:11:25 UTC)
1 row(s)
Took 0.0528 seconds
=> ["articles_snp"]
```

Восстановление данных из снепшота

После создания снепшота измените данные в таблице любым образом. Например, можно полностью удалить одну строку и добавить другую. Пример:

```
deleteall 'articles', 'article2'
put 'articles', 'article3', 'basic:author', 'King'
put 'articles', 'article3', 'basic:header', 'Test article3'
put 'articles', 'article3', 'tags:ref', true
```

Считайте данные из таблицы, чтобы убедиться, что данные изменились. Пример:

```
scan 'articles'
```

Вывод команды должен выглядеть следующим образом:

```
ROW                                          COLUMN+CELL
 article1                                    column=basic:author, timestamp=2024-11-01T08:03:55.730, value=Test author
 article1                                    column=basic:header, timestamp=2024-11-01T08:03:55.758, value=Test article
 article1                                    column=tags:arch, timestamp=2024-11-01T08:03:55.776, value=true
 article1                                    column=tags:concepts, timestamp=2024-11-01T08:03:55.794, value=true
 article1                                    column=tags:tutorials, timestamp=2024-11-01T08:03:58.771, value=true
 article3                                    column=basic:author, timestamp=2024-11-01T08:19:36.078, value=King
 article3                                    column=basic:header, timestamp=2024-11-01T08:19:36.093, value=Test article3
 article3                                    column=tags:ref, timestamp=2024-11-01T08:19:37.856, value=true
2 row(s)
Took 0.0638 seconds
```

Используйте команду restore_snapshot для восстановления таблицы, но сперва сделайте её недоступной для операций с данными. Пример:

```
disable 'articles'
restore_snapshot 'articles_snp'
```

Сделайте таблицу вновь доступной и считайте данные из неё, чтобы убедиться, что восстановление сработало. Пример:

```
enable 'articles'
scan 'articles'
```

Вывод команды должен выглядеть следующим образом:

```
ROW                                          COLUMN+CELL
 article1                                    column=basic:author, timestamp=2024-11-01T08:03:55.730, value=Test author
 article1                                    column=basic:header, timestamp=2024-11-01T08:03:55.758, value=Test article
 article1                                    column=tags:arch, timestamp=2024-11-01T08:03:55.776, value=true
 article1                                    column=tags:concepts, timestamp=2024-11-01T08:03:55.794, value=true
 article1                                    column=tags:tutorials, timestamp=2024-11-01T08:03:58.771, value=true
 article2                                    column=basic:author, timestamp=2024-11-01T08:04:19.531, value=Test author2
 article2                                    column=basic:header, timestamp=2024-11-01T08:04:19.554, value=Test article2
 article2                                    column=tags:ref, timestamp=2024-11-01T08:04:20.802, value=true
2 row(s)
Took 0.0387 seconds
```

Экспорт снепшота

Снепшот можно экспортировать в другой кластер и восстановить таблицу в нем. Для этого сначала подключитесь к узлу HBase исходного кластера через SSH и переключитесь на пользователя `hbase`, а затем выполните экспорт. Команда экспорта имеет следующий синтаксис:

```
$ hbase org.apache.hadoop.hbase.snapshot.ExportSnapshot -snapshot <snapshot_name> -copy-to hdfs://<namenode_active>:<port>
<hbase_dir> -mappers <map_number>
```

где:

- `<snapshot_name>` — имя экспортируемого снепшота;
- `<namenode_active>` — сетевой адрес активного сервера NameNode в кластере назначения (используйте команду `hdfs haadmin -getAllServiceState`, чтобы узнать, какой сервер является активным);
- `<port>` — номер порта сервера NameNode (`8020` по умолчанию);
- `<hbase_dir>` — домашняя директория HBase: значение параметра `zookeeper.znode.parent` в конфигурационном файле *hbase-site.xml* кластера назначения (`/hbase` по умолчанию);
- `<map_number>` — количество экземпляров метода `map` (мапперов) для выполнения задачи экспорта.

Пример:

```
$ sudo -u hbase bash
$ hbase org.apache.hadoop.hbase.snapshot.ExportSnapshot -snapshot articles_snp -copy-to hdfs://av-adh-backup-1.ru-central1.
internal:8020/hbase -mappers 4
```

Перейдите в консоль HBase shell кластера назначения и восстановите таблицу из снепшота:

```
restore_snapshot 'articles_snp'
```

ПРИМЕЧАНИЕ

В этом случае восстановленная таблица уже доступна для операций с данными.

Ротация снепшотов

Важно, чтобы старые и ненужные снепшоты не накапливались и своевременно удалялись. HFile, связанные со снепшотами (или их клонами), не могут быть удалены, пока эти снепшоты существуют, даже если HFile находятся в архиве. Если тот или иной снепшот вам больше не нужен, удалите его с помощью команды delete_snapshot.
Пример:

```
delete_snapshot 'articles_snp'
```

Также можно удалить все снепшоты заданной таблицы с помощью команды delete_table_snapshots или удалить все снепшоты, чьи имена соответствуют заданному регулярному выражению, с помощью команды delete_all_snapshot.

Репликация HBase

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
- Настройка репликации
  - Из одного кластера в другой
  - Из одного кластера в несколько других

Обзор

Репликация в HBase — это механизм, позволяющий копировать содержимое таблицы (или её отдельных семейств столбцов) в другой кластер ADH. Это может служить целям резервирования и восстановления данных, географического распределения данных, агрегации данных и другим.

Кластер, содержащий исходные данные — это кластер источника, а кластер, принимающий данные — это кластер назначения. Любой кластер может быть источником, назначением, или и тем, и другим. Кластер источника может отправлять данные любому числу кластеров назначения. Процесс репликации является инкрементальным: когда в исходные данные вносятся какие-либо изменения, они передаются кластеру назначения. Если кластер назначения по какой-то причине недоступен, то изменения, внесенные с момента последней успешной передачи, хранятся на стороне источника до момента, когда кластер назначения станет доступен. Для работы репликации необходимо, чтобы совпадали имена таблиц источника и назначения, а также их соответствующих семейств столбцов.

> **ПРИМЕЧАНИЕ**
>
> Более подробную информацию о командах HBase shell, используемых для настройки репликации, можно получить в разделе Команды HBase shell → Replication commands.

Настройка репликации

Для репликации таблицы число кластеров может быть любым, при этом один из них будет кластером источника, а остальные — кластерами назначения. Когда все кластеры настроены и работают, выполните следующее:

1. Откройте веб-интерфейс ADCM и выберите ваш кластер источника ADH.
2. Перейдите в раздел ***Services*** → ***HBase*** → ***Primary configuration*** и активируйте переключатель ***Show advanced***.
3. Откройте секцию настроек ***Custom hbase-site.xml*** и нажмите ***Add property***.
4. В поле имени параметра введите `hbase.replication`. В качестве значения параметра введите `true`.
5. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.

Теперь вы можете настроить репликацию в различных сценариях.

Из одного кластера в другой

Тестовая таблица, используемая для демонстрации процесса настройки репликации, получена с помощью импорта файла people.csv. Процедура импорта описана в статье Массовая загрузка с использованием встроенных задач MapReduce.

Для настройки репликации таблицы из кластера источника в кластер назначения выполните следующее:

1. Перейдите в консоль HBase обоих кластеров.
2. Создайте таблицу с той же структурой и свойствами, что и у исходной таблицы, в кластере назначения:

```
create 'people', {NAME => 'basic', VERSIONS => 5}, {SPLITS => ['F', 'K', 'P', 'W']}, {NAME => 'location', VERSIONS => 5}
```

   Для получения свойств таблицы можно воспользоваться командой `describe` в консоли HBase кластера источника.

3. В кластере источника создайте запись отношения репликации с помощью команды add_peer:

```
add_peer '1', CLUSTER_KEY => 'av-adh-backup-1.ru-central1.internal,av-adh-backup-2.ru-central1.internal,av-adh-backup-3.ru-
central1.internal:2181:/hbase'
```

4. Сделайте таблицу недоступной для операций с данными с помощью команды disable:

```
disable 'people'
```

5. Включите репликацию для семейств столбцов с помощью команды alter:

```
alter 'people', {NAME => 'basic', REPLICATION_SCOPE => '1'}, {NAME => 'location', REPLICATION_SCOPE => '1'}
```

6. Сделайте таблицу вновь доступной для операций с данными с помощью команды enable:

```
enable 'people'
```

Этого должно быть достаточно для запуска процесса репликации. Вы можете внести какие-нибудь изменения в таблицу (например, добавить значение) и затем проверить статус процесса репликации с помощью команды status:

```
status 'replication'
```

Вывод команды будет выглядеть подобным образом:

```
version 2.5.10
3 live servers
    av-adh-1.ru-central1.internal:
       SOURCE: PeerID=1
         Normal Queue: 1
           No Ops shipped since last restart, SizeOfLogQueue=1, EditsReadFromLogQueue=0, OpsShippedToTarget=0, No edits for this
source since it started, Replication Lag=0
      SINK: TimeStampStarted=1729231941314, Waiting for OPs...
    av-adh-2.ru-central1.internal:
       SOURCE: PeerID=1
         Normal Queue: 1
           No Ops shipped since last restart, SizeOfLogQueue=1, EditsReadFromLogQueue=0, OpsShippedToTarget=0, No edits for this
source since it started, Replication Lag=0
      SINK: TimeStampStarted=1729231977710, Waiting for OPs...
    av-adh-3.ru-central1.internal:
       SOURCE: PeerID=1
         Normal Queue: 1
           AgeOfLastShippedOp=799, TimeStampOfLastShippedOp=Fri Oct 18 12:27:53 UTC 2024, SizeOfLogQueue=1,
EditsReadFromLogQueue=1, OpsShippedToTarget=1, TimeStampOfNextToReplicate=Fri Oct 18 12:27:53 UTC 2024, Replication Lag=0
```

Свойство `OpsShippedToTarget` показывает, сколько операций изменения было передано в таблицу в кластере назначения. Ненулевое значение хотя бы на одном сервере означает, что репликация происходит.

Если таблица в кластере источника была непустой перед настройкой репликации, возможно, понадобится скопировать её содержимое в кластер назначения. Чтобы это сделать, подключитесь к узлу HBase кластера источника по SSH и выполните команды следующего вида:

```
$ cd /bin
./hbase org.apache.hadoop.hbase.mapreduce.CopyTable --peer.adr=av-adh-backup-1.ru-central1.internal,av-adh-backup-2.ru-central1.
internal,av-adh-backup-3.ru-central1.internal:2181:/hbase people
```

где:

- `--peer.adr` — значение `CLUSTER_KEY`, использованное ранее при составлении команды `add_peer`;
- `people` — название копируемой таблицы.

Из одного кластера в несколько других

Для настройки репликации в несколько кластеров назначения выполните те же действия, что и для одного кластера, но при добавлении отношений репликации создайте отдельное отношение для каждого кластера назначения, не забывая указывать различные идентификаторы.

Пример:

```
add_peer '1', CLUSTER_KEY => 'av-adh-backup1-1.ru-central1.internal,av-adh-backup1-2.ru-central1.internal,av-adh-backup1-3.ru-
central1.internal:2181:/hbase'
add_peer '2', CLUSTER_KEY => 'av-adh-backup2-1.ru-central1.internal,av-adh-backup2-2.ru-central1.internal,av-adh-backup2-3.ru-
central1.internal:2181:/hbase'
add_peer '3', CLUSTER_KEY => 'av-adh-backup3-1.ru-central1.internal,av-adh-backup3-2.ru-central1.internal,av-adh-backup3-3.ru-
central1.internal:2181:/hbase'
```

Резервное копирование и восстановление данных в HBase

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
  - Ключевые понятия
  - Архитектурные стратегии развертывания
- Конфигурация
- Ограничения
- Основные команды
- Примеры
  - HBase
  - Phoenix

Обзор

Функциональность резервного копирования в HBase была добавлена в ADH версии 4.1.0 (HBase версии 2.6).

Эта функциональность предназначена для создания "холодных" копий данных, что является важнейшим элементом стратегии обеспечения бесперебойной работы и восстановления после сбоев. В отличие от репликации, которая создает "горячие" копии, резервные копии хранятся во внешнем хранилище и требуют ручных процедур восстановления, что обеспечивает защиту от катастрофических сбоев.

Ключевые понятия

Функциональность резервного копирования в HBase использует следующие термины и понятия:

- **Полная резервная копия** — содержит полное состояние таблицы на момент создания. Служит основой для последующих инкрементальных резервных копий.
- **Инкрементальная резервная копия** — содержит только изменения из WAL относительно последней полной или инкрементальной резервной копии. Эффективна с точки зрения объема и позволяет восстановить данные на определенный момент времени.
- **Набор резервных копий** — именованная группа таблиц для упрощения управления операциями резервного копирования и восстановления. Рекомендуется сохранять резервные копии каждого набора по выделенному пути.
- **Идентификатор резервной копии** — уникальный идентификатор (основанный на времени эпохи Unix), присваиваемый каждому сеансу резервного копирования.
- **Слияние инкрементов** — возможность объединения нескольких успешных резервных копий в одну.

Архитектурные стратегии развертывания

Для резервного копирования данных в HBase можно использовать следующие архитектурные стратегии развертывания:

- **Внутри кластера** — резервные копии хранятся в том же HDFS-кластере. Подходит только для тестирования и не обеспечивает отказоустойчивости.
- **Выделенный кластер** — резервные копии создаются в отдельном HDFS-кластере, часто с более дешевой аппаратной конфигурацией. Рекомендуется для производственных сред, обеспечивает географическое разделение.
- **Облачное хранилище** — резервные копии сохраняются в облаке (например, S3) или в HDFS-совместимой системе хранения (через S3A, WebHDFS). Наиболее отказоустойчивый вариант.

Конфигурация

Чтобы настроить функцию резервного копирования в HBase, выполните следующее:

1. Откройте веб-интерфейс ADCM и выберите ваш кластер ADH.
2. Перейдите в раздел ***Services*** → ***HBase*** → ***Primary configuration*** и активируйте переключатель ***Show advanced***.
3. Откройте секцию настроек ***hbase-site.xml*** и установите значение `true` для параметра ***hbase.backup.enable***.
4. Проверьте следующие параметры и убедитесь, что они имеют соответствующие значения:
   - ***hbase.master.logcleaner.plugins*** — `org.apache.hadoop.hbase.backup.master.BackupLogCleaner`. Вы также можете добавить к этому параметру другие классы менеджера процедур резервного копирования, разделяя их запятыми. То же самое можно сделать для всех следующих параметров в этом списке.
   - ***hbase.procedure.master.classes*** — `org.apache.hadoop.hbase.backup.master.LogRollMasterProcedureManager`.
   - ***hbase.procedure.regionserver.classes*** — `org.apache.hadoop.hbase.backup.regionserver.LogRollRegionServerProcedureManager`.
   - ***hbase.coprocessor.region.classes*** — `org.apache.hadoop.hbase.backup.BackupObserver`.
   - ***hbase.coprocessor.master.classes*** — `org.apache.hadoop.hbase.backup.BackupMasterObserver`.
   - ***hbase.master.hfilecleaner.plugins*** — `org.apache.hadoop.hbase.backup.BackupHFileCleaner`.
5. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Reconfig and graceful restart***.

Ограничения

Функциональность резервного копирования в HBase имеет следующие ограничения:

- **Последовательное выполнение**. Невозможно запустить несколько операций резервного копирования или восстановления одновременно.
- **Невозможность отмены**. Запущенные операции резервного копирования или восстановления нельзя отменить.
- **Права суперпользователя**. Все операции выполняются от имени суперпользователя HBase.
- **Восстановление только в онлайн-кластер**. Целевой кластер для восстановления должен работать и быть доступен.
- **Единое место назначения**. Резервная копия может быть сохранена только в одном месте. Дублирование резервной копии требует ручных действий.
- **Увеличение WAL**. Инкрементальные резервные копии предотвращают удаление файлов WAL перед следующим резервным копированием, что может привести к увеличению объема данных в HDFS. Необходимо внимательно следить за расписанием и хранилищем.
- **Прозрачное шифрование данных (Transparent Data Encryption, TDE)**. Функциональность не тестировалась в кластерах с включенным шифрованием данных.
- **Восстановление в другом кластере**. При копировании резервной копии на другой кластер восстановление будет возможным, но после этого вам понадобится полная резервная копия для создания истории.

Основные команды

Функциональность резервного копирования HBase использует следующие основные команды:

1. Создание набора таблиц для гранулярной конфигурации резервного копирования:

```
$ hbase backup set add <set_name1> <table1>,<table2>
$ hbase backup set add <set_name2> <table3>
```

2. Создание полной и инкрементальной резервной копии для разных наборов:

```
$ hbase backup create full hdfs://<nn>:8020/tmp/backups-path1/ -w 3 -s <set_name1> -d
$ hbase backup create full hdfs://<nn>:8020/tmp/backups-path2/ -w 3 -s <set_name2> -d
$ hbase backup create incremental hdfs://<nn>:8020/tmp/backups-path1/ -w 3 -s <set_name1> -d
$ hbase backup create incremental hdfs://<nn>:8020/tmp/backups-path2/ -w 3 -s <set_name2> -d
```

3. Просмотр истории резервного копирования:

```
$ hbase backup history
```

4. Слияние резервных копий:

```
$ hbase backup merge <backup_increment_id1>,<backup_increment_id2>,<backup_increment_id3>
```

   Идентификаторы можно найти в истории или в корневой директории резервных копий в файловой системе.

5. Восстановление:

```
$ hbase restore hdfs://<nn>:8020/tmp/backups-path2/ <backup_id> -o -s <set_name2> -d
```

Примеры

HBase

Этот пример показывает функциональность резервного копирования в HBase без Phoenix.

1. Создайте исходную таблицу:

```
$ hbase shell
```

```
create 'user_activity', 'cf1', 'cf2'
put 'user_activity', 'user1', 'cf1:name', 'Alice'
put 'user_activity', 'user1', 'cf1:email', 'alice@email.com'
put 'user_activity', 'user1', 'cf2:last_login', '2024-01-15'
put 'user_activity', 'user1', 'cf2:login_count', '5'
put 'user_activity', 'user2', 'cf1:name', 'Bob'
put 'user_activity', 'user2', 'cf1:email', 'bob@email.com'
put 'user_activity', 'user2', 'cf2:last_login', '2024-01-14'
put 'user_activity', 'user2', 'cf2:login_count', '3'
put 'user_activity', 'user3', 'cf1:name', 'Charlie'
put 'user_activity', 'user3', 'cf1:email', 'charlie@email.com'
put 'user_activity', 'user3', 'cf2:last_login', '2024-01-13'
put 'user_activity', 'user3', 'cf2:login_count', '7'
```

2. Выйдите из командной оболочки `hbase` и создайте набор резервных копий:

```
exit
```

```
$ hbase backup set add user_backup_set user_activity
```

3. Создайте полную резервную копию:

```
$ hbase backup create full hdfs://tmp/hbase-backup -s user_backup_set -w 3
```

4. Добавьте новые данные:

```
$ hbase shell
```

```
put 'user_activity', 'user4', 'cf1:name', 'Diana'
put 'user_activity', 'user4', 'cf1:email', 'diana@email.com'
put 'user_activity', 'user4', 'cf2:last_login', '2024-01-16'
put 'user_activity', 'user4', 'login_count', '2'
put 'user_activity', 'user1', 'cf2:login_count', '6'
put 'user_activity', 'user2', 'cf2:last_login', '2024-01-16'
```

5. Выйдите из командной оболочки `hbase` и создайте инкрементальную резервную копию:

```
exit
```

```
$ hbase backup create incremental hdfs://tmp/hbase-backup -s user_backup_set -w 3
```

6. Добавьте еще новых данных и создайте вторую инкрементальную резервную копию:

```
$ hbase shell
```

```
put 'user_activity', 'user5', 'cf1:name', 'Eve'
put 'user_activity', 'user5', 'cf1:email', 'eve@email.com'
deleteall 'user_activity', 'user3'
put 'user_activity', 'user1', 'cf2:login_count', '7'
exit
```

```
$ hbase backup create incremental hdfs://tmp/hbase-backup -s user_backup_set -w 3
```

7. Проверьте историю и описание:

```
$ hbase backup history -s user_backup_set
$ hbase backup describe <backup_id>
```

8. Имитируйте потерю данных:

```
$ hbase shell
```

```
disable 'user_activity'
drop 'user_activity'
```

9. Выйдите из командной оболочки `hbase` и восстановите данные:

```
exit
```

```
$ hbase restore hdfs://tmp/hbase-backup <backup_id> -s user_backup_set
```

   Данные можно восстановить на момент любой точки во времени, в которой была создана одна из предыдущих резервных копий.

Phoenix

Этот пример показывает функциональность резервного копирования в Phoenix.

1. Запустите скрипт */usr/lib/phoenix/bin/sqlline.py* и создайте исходную таблицу:

```
CREATE TABLE USER_ACTIVITY (
    USER_ID VARCHAR PRIMARY KEY,
    NAME VARCHAR,
    EMAIL VARCHAR,
    LAST_LOGIN DATE,
    LOGIN_COUNT INTEGER
) COMPRESSION='SNAPPY', SALT_BUCKETS=4;
```

2. Загрузите данные в таблицу:

```
UPSERT INTO USER_ACTIVITY VALUES ('user1', 'Alice', 'alice@email.com', TO_DATE('2024-01-15'), 5);
UPSERT INTO USER_ACTIVITY VALUES ('user2', 'Bob', 'bob@email.com', TO_DATE('2024-01-14'), 3);
UPSERT INTO USER_ACTIVITY VALUES ('user3', 'Charlie', 'charlie@email.com', TO_DATE('2024-01-13'), 7);
```

3. Нажмите `Ctrl+D` и создайте набор резервных копий. Для таблиц Phoenix требуются дополнительные системные таблицы. Вам может понадобиться большее их количество, например, если используются индексы. Поэтому резервные копии таблиц Phoenix лучше хранить в одном наборе:

```
$ hbase backup set add phoenix_backup_set USER_ACTIVITY,SYSTEM.CATALOG,SYSTEM.SEQUENCE,SYSTEM.STATS
```

4. Создайте полную резервную копию:

```
$ hbase backup create full hdfs://tmp/hbase-backup -s phoenix_backup_set -w 3
```

5. Снова запустите скрипт */usr/lib/phoenix/bin/sqlline.py* и добавьте новые данные:

```
UPSERT INTO USER_ACTIVITY VALUES ('user4', 'Alice', 'alice@email.com', TO_DATE('2024-01-15'), 5);
UPSERT INTO USER_ACTIVITY VALUES ('user5', 'Bob', 'bob@email.com', TO_DATE('2024-01-14'), 3);
UPSERT INTO USER_ACTIVITY VALUES ('user6', 'Charlie', 'charlie@email.com', TO_DATE('2024-01-13'), 7);
```

6. Нажмите `Ctrl+D` и создайте инкрементальную копию:

```
$ hbase backup create incremental hdfs://tmp/hbase-backup -s phoenix_backup_set -w 3
```

7. Снова запустите скрипт */usr/lib/phoenix/bin/sqlline.py* и добавьте еще новых данных:

```
UPSERT INTO USER_ACTIVITY VALUES ('user7', 'Alice', 'alice@email.com', TO_DATE('2024-01-15'), 5);
UPSERT INTO USER_ACTIVITY VALUES ('user8', 'Bob', 'bob@email.com', TO_DATE('2024-01-14'), 3);
UPSERT INTO USER_ACTIVITY VALUES ('user9', 'Charlie', 'charlie@email.com', TO_DATE('2024-01-13'), 7);
```

8. Нажмите `Ctrl+D` и создайте вторую инкрементальную копию:

```
$ hbase backup create incremental hdfs://tmp/hbase-backup -s phoenix_backup_set -w 3
```

9. Проверьте историю:

```
$ hbase backup history -s user_backup_set
$ hbase backup describe <backup_id>
```

10. Снова запустите скрипт */usr/lib/phoenix/bin/sqlline.py* и имитируйте потерю данных:

```
DROP TABLE USER_ACTIVITY;
```

11. Нажмите `Ctrl+D` и восстановите данные:

```
$ hbase restore hdfs://tmp/hbase-backup <backup_id> -s phoenix_backup_set
```

   Данные можно восстановить на момент любой точки во времени, в которой была создана одна из предыдущих резервных копий.

Оптимизация производительности в HBase

Владимир Адаменков

Содержание

Назад к содержанию

- Настройки HDFS
- Настройки HBase
- Прочее
  - Схема таблиц
  - Дизайн ключей строк
  - Распределение полного сжатия
  - Компрессия
  - Фильтры Блума
  - Массовая загрузка

Эта статья содержит рекомендации по улучшению производительности кластера HBase. Некоторые из описанных приемов нельзя точно сформулировать для всех случаев — их нужно подстраивать под каждую конкретную конфигурацию.

Настройки HDFS

Параметры HDFS

Параметр	Значение по умолчанию и описание	Рекомендация
dfs.datanode.max.transfer.threads	Значение по умолчанию: `4096`. Этот параметр определяет верхний предел количества файлов, которые узел HDFS DataNode может обслуживать одновременно	Установите для этого параметра значение `8192`. Этот параметр нужно добавить вручную в секции ***Custom hdfs-site.xml*** на конфигурационной странице сервиса HDFS

Настройки HBase

Настройки RegionServer

Параметр	Значение по умолчанию и описание	Рекомендация
zookeeper.session.timeout	Значение по умолчанию: `90000` (в миллисекундах). Этот параметр определяет тайм-аут, после которого Master-сервер считает сервер регионов неактивным. Значение по умолчанию не влияет на работу системы, если в конфигурации ZooKeeper остается значение по умолчанию для tick time (2 секунды). Сервис ZooKeeper имеет собственные тайм-ауты для неактивных сервисов: от двух до двадцати tick time. По умолчанию это означает, что сервис ZooKeeper будет ожидать не более 40 секунд до удаления эфемерного znode неактивного сервера регионов, пока Master-сервер все еще будет ожидать ответ от сервера регионов	Понизьте значение параметра тайм-аута HBase или увеличьте значение tick time в ZooKeeper: будет использовано наименьшее значение. Учитывайте следующее: - Не понижайте значение параметра `zookeeper.session.timeout` слишком сильно — всегда существуют задержки и паузы из-за сетевого трафика, работы сборщика мусора и других временных проблем. Серверы регионов должны выдерживать эти колебания, не отключаясь по тайм-ауту. - Не увеличивайте значение tick time в ZooKeeper до более чем 4 или 5 секунд — это может плохо повлиять на состояние кворума ZooKeeper. Параметр `zookeeper.session.timeout` расположен в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase, значение tick time задается параметром `tickTime` в секции ***zoo.cfg*** на конфигурационной странице сервиса ZooKeeper. Оба параметра указываются в миллисекундах
hbase.regionserver.handler.count	Значение по умолчанию: `30`. Этот параметр определяет количество открытых потоков для ответов на входящие запросы к пользовательским таблицам	Каждый поток потребляет процессорное время и оперативную память, поэтому перед изменением значения этого параметра следует оценить среднюю величину нагрузки запросов. Если запросы имеют тяжелую нагрузку, то для этого параметра лучше задать небольшое значение. Но если запросы в основном имеют легкую нагрузку, то можно увеличить значение этого параметра: стоит начать с удвоенного количества процессорных ядер на узле и затем увеличивать его до достижения наилучшей производительности. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.master.wait.on.regionservers.mintostart	Значение по умолчанию: `1`. Этот параметр определяет минимальное количество серверов регионов, которые должны быть зарегистрированы в Master-сервере до распределения им регионов по серверам. Значение по умолчанию `1` означает, что Master-сервер начнет распределять регионы на первом зарегистрированном сервере регионов	Если в вашем кластере много регионов (например, несколько тысяч), то рекомендуется увеличить значение этого параметра, чтобы назначение регионов начиналось не сразу для их более равномерного распределения. В противном случае первый зарегистрированный сервер регионов может стать узким местом, замедлить запуск кластера и даже привести к потере доступа к серверу регионов. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

Настройки Heap memory

Параметр	Значение по умолчанию и описание	Рекомендация
HBASE_MASTER_OPTS	Значение по умолчанию: `-Xms700m -Xmx9G`. Этот параметр определяет количество оперативной памяти, выделяемой для Master-сервера	Значения по умолчанию достаточно в большинстве случаев, так как Master-сервер не требует большого количества памяти для работы. Тем не менее, если ваш кластер достаточно большой и имеет огромное количество таблиц и/или регионов, может понадобиться увеличить верхний предел памяти, поскольку Master-сервер хранит состояния регионов в своей памяти. Этот параметр находится в секции ***hbase-env.sh*** на конфигурационной странице сервиса HBase
HBASE_REGIONSERVER_OPTS	Значение по умолчанию: `-Xms700m -Xmx9G`. Этот параметр определяет размер heap-памяти сервера регионов, в которой хранятся MemStore и Block cache	В общем случае, чем больше значение этого параметра — тем лучше. Придерживайтесь следующих соображений: - Размер heap-памяти сервера регионов должен быть пропорционален количеству регионов на узле. - Если пиковое потребление памяти достигает 60 или 70 процентов — увеличьте объем heap-памяти. - Если JVM используют алгоритм сборщика мусора CMS, который предоставляется по умолчанию в Arenadata HBase, heap-память сервера регионов не должна превышать 36 ГБ. В противном случае паузы сборщика мусора "stop-the-world" могут сделать HBase непригодным для использования и сильно осложнить сохранение данных. Если JVM используют алгоритм сборщика мусора G1 (garbage first), то ограничений на размер heap-памяти сервера регионов нет. Этот параметр находится в секции ***hbase-regionserver-env.sh*** на конфигурационной странице сервиса HBase

Настройки Block cache

Параметр	Значение по умолчанию и описание	Рекомендация
hfile.block.cache.size	Значение по умолчанию: `0.4`. Этот параметр определяет долю Block cache в heap-памяти сервера регионов	Значение по умолчанию подходит для ситуации, когда из таблиц постоянно читается большое количество примерно одних и тех же данных. Если ситуация отличается (например, запрашиваемые данные все время разные, или размер запрашиваемых данных в среднем достаточно невелик), то стоит уменьшить значение этого параметра до `0.2` или даже до `0.1`. Общее правило таково: этот параметр не должен быть больше, чем действительно необходимо. Точное значение следует подбирать по производительности, методом проб и ошибок. Также оно должно быть сбалансировано со значением параметра `hbase.regionserver.global.memstore.size`. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

Настройки MemStore и записи в HFile

Параметр	Значение по умолчанию и описание	Рекомендация
hbase.regionserver.global.memstore.size	Значение по умолчанию: `0.4`. Этот параметр определяет долю MemStore в heap-памяти сервера регионов	Нужно задать такое значение, чтобы оно в сумме со значением параметра `hfile.block.cache.size` было равным `0.7` или `0.75`. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.regionserver.global.memstore.size.lower.limit	Значение по умолчанию: `0.95`. Этот параметр задает пороговую долю размера MemStore региона. Когда размер данных в MemStore достигает этого порога, запись в них блокируется и выполняется принудительная запись MemStore в HFile	Менять значение параметра не рекомендуется
hbase.hregion.memstore.flush.size	Значение по умолчанию: `134217728` (128 МБ). Этот параметр определяет размер данных в MemStore перед тем, как они будут записаны в HFile. Когда выполняется операция записи в HFile из одного MemStore, она выполняется и для всех остальных MemStore в том же регионе. Чем меньше это значение, тем чаще происходит запись в HFile, тем больше операций чтения и записи, тем больше количество создаваемых HFile и тем чаще запускается процесс сжатия	Рекомендуется увеличить значение этого параметра вдвое (до `268435456`, то есть 256 МБ). Если в вашем кластере HBase большая загрузка операциями записи, можно увеличить это значение до `536870912` (512 МБ), но это не всегда возможно, особенно если у вас достаточно небольшой размер heap-памяти сервера регионов и большое количество регионов и семейств столбцов. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase. Возможные значения: `134217728`, `268435456`, `536870912`
hbase.hstore.flusher.count	Значение по умолчанию: `2`. Этот параметр определяет количество потоков операции записи MemStore в HFile. Чем больше это значение, тем эффективнее процесс записи, но в то же время больше нагрузка на HDFS и, потенциально, больше количество операций сжатия	Можно начать с увеличения значения этого параметра до `4`, а затем увеличивать его понемногу, проверяя влияние на производительность. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.hregion.memstore.block.multiplier	Значение по умолчанию: `4`. Этот параметр определяет множитель для возможного увеличения размера MemStore (определяемого параметром `hbase.hregion.memstore.flush.size`) в ситуациях интенсивного трафика записи в него	Рекомендуется увеличить его до `8`, если размер оперативной памяти позволяет. Когда размер MemStore достигает максимального значения, определяемого произведением значения этого параметра и параметра `hbase.hregion.memstore.flush.size`, соответствующее семейство столбцов блокируется для операций записи до тех пор, пока содержимое MemStore не будет полностью записано в HFile. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

Настройки разделения регионов

Параметр	Значение по умолчанию и описание	Рекомендация
hbase.regionserver.regionSplitLimit	Значение по умолчанию: `1000`. Этот параметр определяет максимальное количество регионов на узле, после которого разделение регионов больше не выполняется. Значение параметра не является ограничением на количество регионов самих по себе, но скорее точкой прекращения создания новых регионов	Значение по умолчанию, равное `1000`, на практике означает отсутствие ограничений, поскольку в подавляющем большинстве случаев на серверах регионов в кластерах HBase гораздо меньше регионов, чем 1000. Можно самостоятельно рассчитать это ограничение для конкретной ситуации. Например, ваш кластер имеет следующие характеристики: - `HBASE_REGIONSERVER_OPTS` — `-Xms700m -Xmx32G`; - `hbase.regionserver.global.memstore.size` — `0.55`; - `hbase.hregion.memstore.flush.size` — `268435456` (256 МБ); - 2 семейства столбцов в таблице. Чтобы найти количество регионов, нужно умножить размер heap-памяти сервера регионов на долю MemStore в нем (первые два параметра), а затем последовательно разделить результат на значение параметра `hbase.hregion.memstore.flush.size` и на количество семейств столбцов. В данном случае получается следующая формула: $\frac{32\mathrm{GB} \cdot 0.55}{256\mathrm{MB} \cdot 2} = 35.2$ Округлите результат в большую сторону, если он дробный, и полученное значение будет оптимальным количеством регионов на узле. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.hregion.max.filesize	Значение по умолчанию: `10737418240` (10 ГБ). Этот параметр определяет максимальный общий размер HFile в регионе перед его разделением	Значение по умолчанию подходит в большинстве случаев. Его можно увеличить, если производительность позволяет (например, до 20 ГБ), но не сильно: более высокие значения могут плохо повлиять на процесс сжатия. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase

Настройки сжатия

Параметр	Значение по умолчанию и описание	Рекомендация
hbase.hstore.compactionThreshold	Значение по умолчанию: `3`. Этот параметр определяет количество HFile в регионе перед тем, как они будут подвергнуты частичному сжатию	Рекомендуется увеличить значение до `5`. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.hstore.blockingStoreFiles	Значение по умолчанию: `16`. Этот параметр определяет максимальное количество HFile в регионе перед блокировкой обновлений для этого региона. Блокировка сохраняется, пока не будет принудительно выполнено частичное сжатие или пока не пройдет тайм-аут, заданный параметром `hbase.hstore.blockingWaitTime`	Если кластер имеет высокую нагрузку на запись и из-за слишком большого количества HFile не происходит запись новых HFile из MemStore, то увеличьте это значение. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.hstore.blockingWaitTime	Значение по умолчанию: `90000` (в миллисекундах). Этот параметр определяет тайм-аут блокировки обновлений для региона из-за слишком большого количества HFile (см. предыдущий параметр). После истечения этого времени блокировка снимается независимо от того, было ли выполнено частичное сжатие	Этот параметр нужно настраивать в зависимости от требований кластера. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.hstore.compaction.min	Значение по умолчанию: `3`. Этот параметр определяет минимальное количество HFile, предназначенных для сжатия, прежде чем это сжатие будет выполнено	Рекомендуется увеличить это значение до `5`. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.hstore.compaction.max	Значение по умолчанию: `10`. Этот параметр определяет максимальное количество HFile, которые могут участвовать в одной операции частичного сжатия	Значение по умолчанию подходит для большинства случаев, но если кластер имеет высокую нагрузку на запись, увеличьте это значение. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.regionserver.thread.compaction.small	Значение по умолчанию: `1`. Этот параметр определяет количество потоков, используемых для процесса частичного сжатия	Если кластер имеет высокую нагрузку на запись, а операции частичного сжатия выполняются слишком часто и занимают слишком много времени, увеличьте это значение. При этом оно не должно превышать половины количества дисков, доступных для HBase. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.regionserver.thread.compaction.large	Значение по умолчанию: `1`. Этот параметр определяет количество потоков, используемых для процесса полного сжатия	Если кластер имеет высокую нагрузку на запись, можно увеличить это значение, но не рекомендуется делать его выше значения параметра `hbase.regionserver.thread.compaction.small`. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.hstore.compaction.min.size	Значение по умолчанию: `134217728` (128 МБ). Этот параметр определяет максимальный размер HFile, которые автоматически предназначаются для сжатия	Если кластер имеет высокую нагрузку на запись, в нем может оказаться много HFile небольших размеров, которые после частичного сжатия образуют HFile, чей размер все еще меньше указанного значения (особенно в случае небольшого значения параметра `hbase.hstore.compaction.max`) и который вновь потребует сжатия. Понижение значения этого параметра исправляет эту проблему: следует задать значение чуть меньше, чем результат произведения среднего объема MemStore перед выгрузкой в HFile и значения параметра `hbase.hstore.compaction.max`. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.hstore.compaction.ratio	Значение по умолчанию: `1.2F`. Этот параметр определяет коэффициент, применяемый к предыдущему параметру (`hbase.hstore.compaction.min.size`). Произведение значений этих двух параметров определяет верхний предел размера HFile, который может быть предназначен для сжатия	Чем выше это значение, тем чаще выполняется сжатие, но тем меньше HFile при этом будет прочитано, и наоборот. Более низкие значения предпочтительны для кластеров с высокой нагрузкой на запись, а более высокие — для кластеров с высокой нагрузкой на чтение. Значение по умолчанию подходит для большинства случаев; если вы захотите его изменить — рекомендуется выбирать значение в пределах `1.0F` — `1.4F`. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase

Настройки WAL

Параметр	Значение по умолчанию и описание	Рекомендация
hbase.regionserver.logroll.multiplier	Значение по умолчанию: `0.95`. Этот параметр определяет размер файла WAL относительно размера блока HDFS (который задается параметром `dfs.blocksize` в секции ***hdfs-site.xml*** на конфигурационной странице сервиса HDFS). Например, если размер блока HDFS равен 134217728 (128 МБ), то размер файла WAL будет равен 127506842 (приблизительно 121.6 МБ)	Рекомендуется оставить значение этого параметра неизменным во всех случаях. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

hbase.regionserver.maxlogs	Значение по умолчанию: `32`.
	Этот параметр определяет максимальное количество файлов WAL на сервере регионов
		Значение этого параметра нужно рассчитать самостоятельно, выполнив следующее:
		1. Возьмите значение верхнего предела из параметра `HBASE_REGIONSERVER_OPTS` и умножьте его на значение параметра `hbase.regionserver.global.memstore.size`. В результате получится размер MemStore.
		2. Вычислите размер файла WAL, как описано выше для параметра `hbase.regionserver.logroll.multiplier`.
		3. Разделите размер MemStore на размер файла WAL. Округлите результат в большую сторону при необходимости.
		Если кластер имеет высокую нагрузку на запись, когда количество файлов WAL достигает предела, серверу регионов может потребоваться выгрузить все MemStore в HFile, пока все файлы WAL не перейдут в архив и не появится достаточно пространства для операций записи. Это явление называется flush storm и может быть определено по следующим строкам в логе:
		`INFO[regionserver//10.22.100.5:16020.`
		`logRoller] wal.FSHLog:Too many wals: logs=35,`
		`maxlogs=32; forcing flush of 20 regions(s):`
		`d4kjnfnkf34335666d03cb1f`
		В этом случае следует увеличить значение этого параметра.
		Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

Настройки MSLAB

Параметр	Значение по умолчанию и описание	Рекомендация
hbase.hregion.memstore.mslab.chunksize	Значение по умолчанию: `2097152` (2 МБ). Этот параметр определяет размер чанка в MemStore Allocation Buffer	При использовании записей большого размера следует увеличить значение этого параметра в два или в четыре раза. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

Настройки клиента

Параметр	Значение по умолчанию и описание	Рекомендация
hbase.client.write.buffer	Значение по умолчанию: `2097152` (2 МБ). Этот параметр определяет размер буфера на запись. Чем больше это значение, тем больше памяти используется как на стороне сервера, так и на стороне клиента, но тем меньше количество RPC	Рекомендуется задать для этого параметра значение `8388608` (8 МБ). Для оценки размера используемой памяти сервера следует умножить значение этого параметра на значение параметра `hbase.regionserver.handler.count`. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.client.pause	Значение по умолчанию: `100`. Этот параметр определяет длительность паузы клиента. Она используется в качестве тайм-аута перед повторным выполнением операции после отказа её выполнения	Рекомендуется значительно уменьшить это значение (например, до `3` или `5`) во избежание потери данных. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.client.retries.number	Значение по умолчанию: `15`. Этот параметр определяет максимальное количество повторных выполнений операции после отказа её выполнения	Рекомендуется уменьшить это значение до `3` или `5` во избежание потери данных. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.rpc.timeout	Значение по умолчанию: `300000` (в миллисекундах; 5 минут). Этот параметр определяет тайм-аут для всех RPC, отправляемых клиентом в HBase	Если клиентские запросы часто вызывают высокую нагрузку на память, увеличьте это значение. Однако не стоит увеличивать его слишком сильно, поскольку ожидание ответа клиентом может занять много времени и в конечном счете клиент получит отказ. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.client.scanner.timeout.period	Значение по умолчанию: `60000` (в миллисекундах; 1 минута). Этот параметр определяет тайм-аут для RPC, связанных с операциями сканирования	Этот параметр становится бесполезен, если его значение выше значения параметра `hbase.rpc.timeout`, поскольку операция будет отменена после первого истекшего тайм-аута. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.client.scanner.caching	Значение по умолчанию: `2147483647`. Этот параметр определяет максимальное количество строк, которое клиентский сканер может извлечь из HBase за один раз	Если это значение слишком велико, сканер может оказаться перегруженным операциями извлечения и передачи слишком большого количества данных клиенту или даже превысить тайм-аут, в результате чего операция будет безуспешна. Также это приведет к слишком большому потреблению heap memory даже в случае успеха. Слишком низкое значение приведет к неэффективному использованию ресурсов кластера из-за простоев. Значение этого параметра следует тщательно подбирать в зависимости от характеристик дисков, памяти, процессоров и средней нагрузки на чтение. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.client.scanner.max.result.size	Значение по умолчанию: `2097152` (2 МБ). Этот параметр определяет максимальное количество байт, возвращаемых при вызове следующего метода сканера. Если размер одного ряда превышает это значение, он все равно будет возвращен полностью	Это значение подходит для сетей 1G Ethernet. Если пропускная способность сети в вашем кластере выше, можно увеличить это значение, но учитывайте следующее: - Проверьте значение параметра `HBASE_MASTER_OPTS`. Рекомендуется задавать значение параметра `hbase.client.scanner.max.result.size` в восемь раз меньше нижнего предела. - Установленное значение, умноженное на значение параметра `hbase.regionserver.handler.count`, не должно превышать верхний предел. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

Прочее

Схема таблиц

Рекомендуется использовать не более двух семейств столбцов в одной таблице.

Дизайн ключей строк

Рекомендуется использовать технологии salting/hashing для более равномерного распределения строк в регионах.

Распределение полного сжатия

Для более равномерной нагрузки на кластер имеет смысл распределить процедуры полного сжатия для различных таблиц и семейств столбцов по расписанию с использованием планировщика `cron` или других подобных утилит.

Компрессия

В некоторых случаях использование компрессии может улучшить производительность за счет уменьшения размера HFile, что в свою очередь уменьшит количество операций ввода-вывода.

Фильтры Блума

Рекомендуется использовать фильтры Блума для ускорения операций поиска в HFile. Фильтры Блума позволяют определить наличие в HFile определенного ключа после прочтения небольшого объема данных из этого HFile. В противном случае выполняется поиск в следующем HFile.

Массовая загрузка

Рекомендуется использовать массовую загрузку при наличии такой возможности, поскольку это самый быстрый способ загрузки данных в HBase. Если массовая загрузка невозможна, рекомендуется объединять одиночные запросы (такие как `GET`, `PUT` или `DELETE`) в списки, поскольку эти операции весьма ресурсоемки при выполнении их по отдельности.

Логирование в HBase

Альберт Багдасарян

Содержание

Назад к содержанию

Использование файловой системы

Использование Web-интерфейса

Оба типа серверов (Master Server и Region Server) ведут свои журналы, которые могут быть полезны при анализе причин появления различных ошибок. Эти журналы расположены в каталоге */var/log/hbase* на каждом из серверов. Доступ к ним можно получить двумя способами, приведенными ниже.

Использование файловой системы

Файловая система сервера предоставляет прямой доступ к файлам журналов:

1. Подключитесь к серверу с помощью SSH и запустите следующую команду:

```
$ ls -la /var/log/hbase
```

Вывод команды на консоль приведен ниже.

```
total 10004
drwxr-xr-x.  2 hbase hbase     4096 Nov 29 06:17 .
drwxr-xr-x. 16 root  root      4096 Nov 29 05:35 ..
-rw-r--r--.  1 hbase hbase 10209158 Nov 29 11:34 hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.log
-rw-r--r--.  1 hbase hbase      692 Nov 29 06:17 hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out
-rw-r--r--.  1 hbase hbase      692 Nov 29 05:35 hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.1
-rw-r--r--.  1 hbase hbase      692 Nov 26 07:43 hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.2
-rw-r--r--.  1 hbase hbase      692 Nov 26 05:58 hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.3
-rw-r--r--.  1 hbase hbase      692 Nov 26 05:26 hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.4
-rw-r--r--.  1 hbase hbase      692 Nov 25 15:00 hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.5
```

ПРИМЕЧАНИЕ

Журналы расположены в локальной файловой системе сервера, а не в HDFS.

2. Выведите на экран содержимое какого-либо файла, например:

```
$ tail /var/log/hbase/hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.log
```

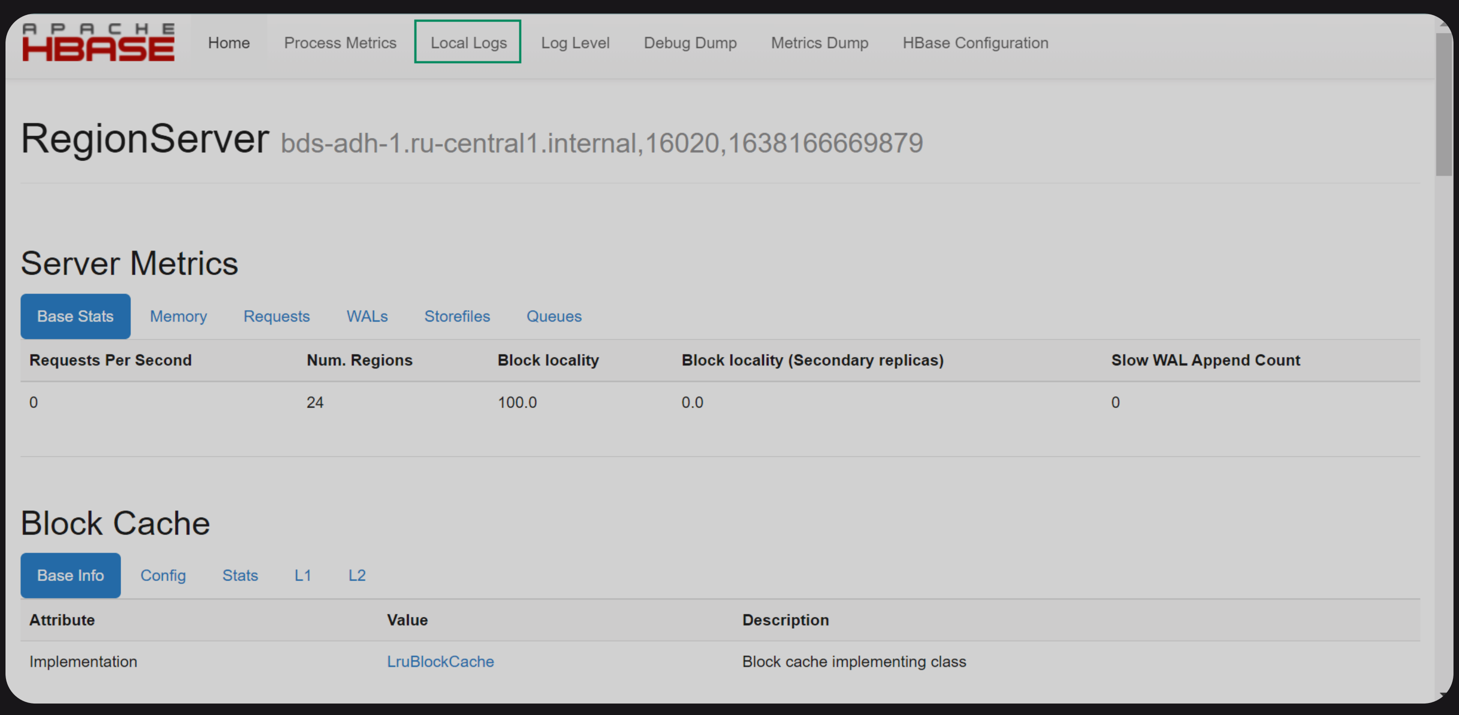
Пример вывода на экран:

```
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-pig.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-thin-client.jar!/org/slf4j/impl/StaticLoggerBinder.
class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
0    [RS-EventLoopGroup-1-9] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:48266, version=2.0.2,
sasl=false, ugi=hbase (auth:SIMPLE), service=ClientService
10629 [RS-EventLoopGroup-1-10] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:48328, version=2.0.
2, sasl=false, ugi=hbase (auth:SIMPLE), service=AdminService
10263041 [RS-EventLoopGroup-1-5] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:35086, version=2.
0.2, sasl=false, ugi=hbase (auth:SIMPLE), service=ClientService
10570940 [RS-EventLoopGroup-1-9] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:36368, version=2.
0.2, sasl=false, ugi=hbase (auth:SIMPLE), service=ClientService
18488201 [RS-EventLoopGroup-1-16] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:41578,
version=2.0.2, sasl=false, ugi=hbase (auth:SIMPLE), service=AdminService
18991216 [RS-EventLoopGroup-1-12] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:43712,
version=2.0.2, sasl=false, ugi=hbase (auth:SIMPLE), service=AdminService
```

Использование Web-интерфейса

Следующие шаги демонстрируют просмотр журналов в Web-интерфейсе HBase:

1. Откройте домашнюю страницу Web-интерфейса какого либо сервера: Master Server или Region Server.
2. Выберите пункт меню ***Local Logs***.



RegionServer bds-adh-1.ru-central1.internal,16020,1638166669879

Server Metrics

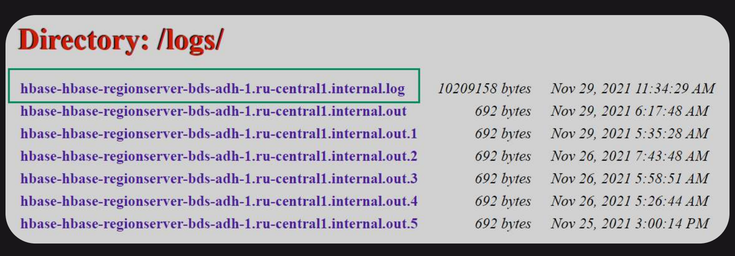
Requests Per Second	Num. Regions	Block locality	Block locality (Secondary replicas)	Slow WAL Append Count
0	24	100.0	0.0	0

Block Cache

Attribute	Value	Description
Implementation	LruBlockCache	Block cache implementing class

Переключение на список журналов

3. Кликните на названии какого-либо файла в полученном списке.

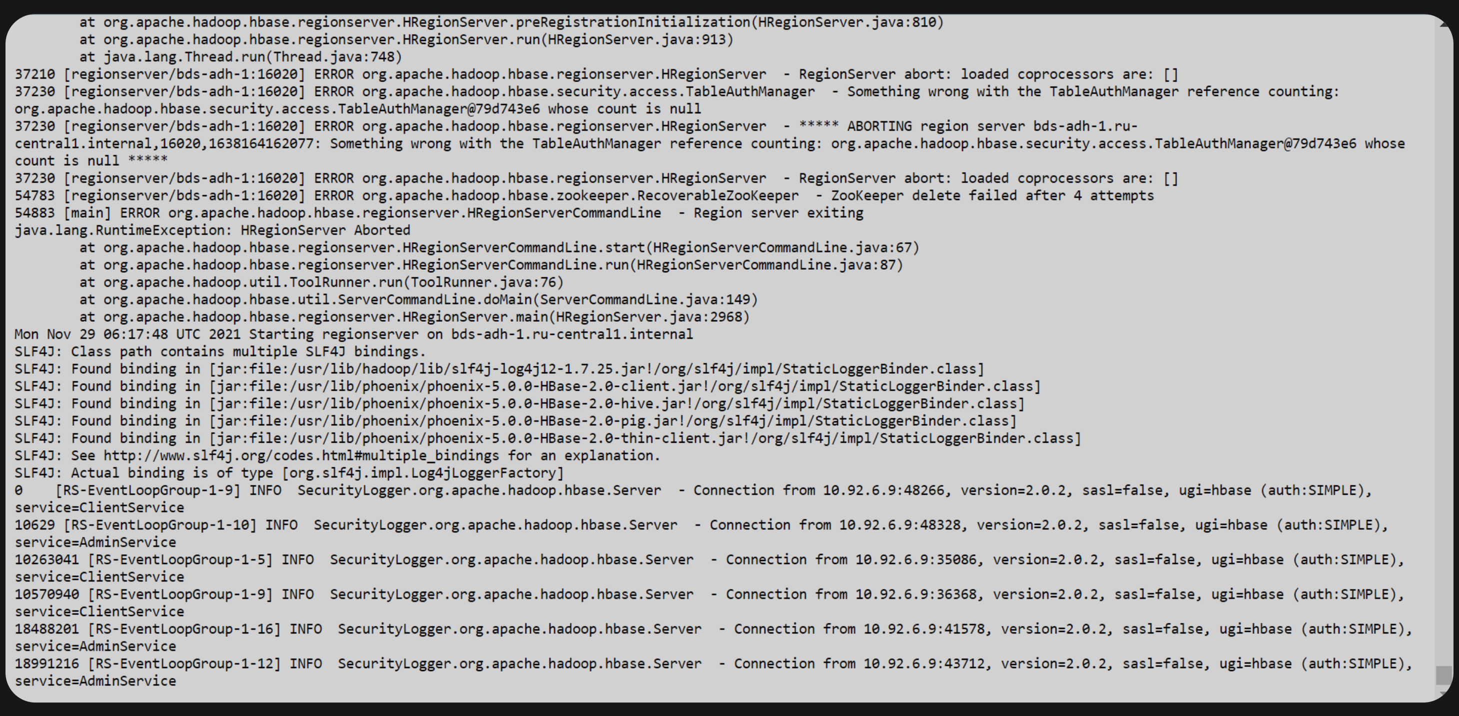


Directory: /logs/

hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.log	10209158 bytes	Nov 29, 2021 11:34:29 AM
hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out	692 bytes	Nov 29, 2021 6:17:48 AM
hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.1	692 bytes	Nov 29, 2021 5:35:28 AM
hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.2	692 bytes	Nov 26, 2021 7:43:48 AM
hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.3	692 bytes	Nov 26, 2021 5:58:51 AM
hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.4	692 bytes	Nov 26, 2021 5:26:44 AM
hbase-hbase-regionserver-bds-adh-1.ru-central1.internal.out.5	692 bytes	Nov 25, 2021 3:00:14 PM

Выбор файла журнала

4. Просмотрите содержимое файла. Сообщения, записанные последними, расположены внизу.



```
        at org.apache.hadoop.hbase.regionserver.HRegionServer.preRegistrationInitialization(HRegionServer.java:810)
        at org.apache.hadoop.hbase.regionserver.HRegionServer.run(HRegionServer.java:913)
        at java.lang.Thread.run(Thread.java:748)
37210 [regionserver/bds-adh-1:16020] ERROR org.apache.hadoop.hbase.regionserver.HRegionServer  - RegionServer abort: loaded coprocessors are: []
37230 [regionserver/bds-adh-1:16020] ERROR org.apache.hadoop.hbase.security.access.TableAuthManager  - Something wrong with the TableAuthManager reference counting:
org.apache.hadoop.hbase.security.access.TableAuthManager@79d743e6 whose count is null
37230 [regionserver/bds-adh-1:16020] ERROR org.apache.hadoop.hbase.regionserver.HRegionServer  - ***** ABORTING region server bds-adh-1.ru-
central1.internal,16020,1638164162077: Something wrong with the TableAuthManager reference counting: org.apache.hadoop.hbase.security.access.TableAuthManager@79d743e6 whose
count is null *****
37230 [regionserver/bds-adh-1:16020] ERROR org.apache.hadoop.hbase.regionserver.HRegionServer  - RegionServer abort: loaded coprocessors are: []
54783 [regionserver/bds-adh-1:16020] ERROR org.apache.hadoop.hbase.zookeeper.RecoverableZooKeeper  - ZooKeeper delete failed after 4 attempts
54883 [main] ERROR org.apache.hadoop.hbase.regionserver.HRegionServerCommandLine  - Region server exiting
java.lang.RuntimeException: HRegionServer Aborted
        at org.apache.hadoop.hbase.regionserver.HRegionServerCommandLine.start(HRegionServerCommandLine.java:67)
        at org.apache.hadoop.hbase.regionserver.HRegionServerCommandLine.run(HRegionServerCommandLine.java:87)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:76)
        at org.apache.hadoop.hbase.util.ServerCommandLine.doMain(ServerCommandLine.java:149)
        at org.apache.hadoop.hbase.regionserver.HRegionServer.main(HRegionServer.java:2968)
Mon Nov 29 06:17:48 UTC 2021 Starting regionserver on bds-adh-1.ru-central1.internal
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-client.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-hive.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-pig.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-thin-client.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
0    [RS-EventLoopGroup-1-9] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:48266, version=2.0.2, sasl=false, ugi=hbase (auth:SIMPLE),
service=ClientService
10629 [RS-EventLoopGroup-1-10] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:48328, version=2.0.2, sasl=false, ugi=hbase (auth:SIMPLE),
service=AdminService
10263041 [RS-EventLoopGroup-1-5] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:35086, version=2.0.2, sasl=false, ugi=hbase (auth:SIMPLE),
service=ClientService
10570940 [RS-EventLoopGroup-1-9] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:36368, version=2.0.2, sasl=false, ugi=hbase (auth:SIMPLE),
service=ClientService
18488201 [RS-EventLoopGroup-1-16] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:41578, version=2.0.2, sasl=false, ugi=hbase (auth:SIMPLE),
service=AdminService
18991216 [RS-EventLoopGroup-1-12] INFO  SecurityLogger.org.apache.hadoop.hbase.Server  - Connection from 10.92.6.9:43712, version=2.0.2, sasl=false, ugi=hbase (auth:SIMPLE),
service=AdminService
```

Чтение журнала

Управление сервисом HBase через ADCM

Евгения Кузина

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом HBase. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса HBase, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты HBase. При выполнении этого действия на этапе ***Configuration*** доступны следующие параметры: ■ ***Remove RegionServer without region unloading***. При значении `true` удаляет компонент RegionServer, не выгружая из него данные. ■ ***Limit the number of threads used by the region mover***. Устанавливает максимальное количество потоков для процесса RegionMover. ■ ***Timeout for moving regions***. Если регионы не могут быть перемещены в течение указанного тайм-аута, действие завершается с ошибкой.
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются. Сервис поддерживает функцию ***Rolling restart***, которая позволяет перезапускать компоненты сервиса по одному (или партиями), а не все сразу. Это помогает минимизировать время простоя сервиса при рестарте. Опция ***Rolling restart*** имеет следующие параметры: ■ ***Limit the number of threads used by the region mover*** — максимальное количество регионов, перемещаемых параллельно. ■ ***Timeout for moving regions*** — максимальное время, отведённое на перемещение регионов. Если не удается переместить все регионы в течение указанного времени, процедура рестарта завершается с ошибкой. ■ ***Graceful restart RegionServers*** — включает/выключает плавный перезапуск компонентов RegionServer. Этот параметр может привести к многократному перемещению регионов, что может ухудшить производительность клиента HBase. ■ ***Roll masters*** — позволяет перезапускать компоненты HBase Master Server по одному. Если установлено значение `false`, все компоненты HBase Master Server перезапускаются параллельно. ■ ***max_wait_time*** — максимальное время в секундах для проверки результатов после перезапуска (проверка Znode). ■ ***pause_after_masters_restart*** — тайм-аут простоя в секундах после перезапуска компонентов HBase Master Server. Может потребоваться для инициализации/стабилизации работы компонентов.
Manage Ranger plugin	Включает или отключает плагин Ranger для HBase. Во время этого действия RegionServers перезапускаются. Когда RegionServer завершает работу, регионы (regions), размещенные на этом RegionServer, перемещаются на другие RegionServer перед остановкой текущего RegionServer. При запуске этого действия необходимо указать следующие параметры: `Limit the number of threads used by the region mover for graceful RegionServer restarting` (ограничение количества потоков, используемых средством перемещения региона (region mover), для корректного перезапуска RegionServer) и `Timeout for moving regions for graceful RegionServer restarting` (тайм-аут для перемещения регионов для корректного перезапуска RegionServer). Если регионы не были перемещены до истечения тайм-аута, действие прерывается и возникает ошибка
Reconfig and graceful restart	Применяет настройки, указанные в пользовательском интерфейсе ADCM, и перезапускает сервис HBase. Во время этого действия RegionServers перезапускаются. Когда RegionServer завершает работу, регионы (regions), размещенные на этом RegionServer, перемещаются на другие RegionServer перед остановкой текущего RegionServer. При запуске этого действия необходимо указать следующие параметры: `Limit the number of threads used by the region mover for graceful RegionServer restarting` (ограничение количества потоков, используемых средством перемещения региона (region mover), для корректного перезапуска RegionServer) и `Timeout for moving regions for graceful RegionServer restarting` (тайм-аут для перемещения регионов для корректного перезапуска RegionServer). Если регионы не были перемещены до истечения тайм-аута, действие прерывается и возникает ошибка

Действия с компонентами HBase

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Подключение к HDFS

Елена Костюченко

Содержание

Назад к содержанию

- Командная строка
- HTTP REST API
- Библиотека libhdfs для C
  - Подготовительные шаги
  - Создание программы на C
  - Компиляция и запуск
- Java API

Статья описывает подключение к сервису HDFS с помощью командной строки (CLI), WebHDFS API и библиотек HDFS. Вы также можете подключиться к HDFS через веб-интерфейс. Для получения информации о графических интерфейсах HDFS обратитесь к статье Обзор HDFS UI.

Командная строка

Используя CLI, можно получить доступ к HDFS с любого хоста в кластере. Следующие шаги описывают, как подключиться к HDFS с помощью командной строки. Подразумевается, что для кластера уже настроен доступ по SSH и пароль не требуется.

1. Подключитесь к хосту кластера ADH через SSH:

```
$ ssh <USER>@<HOST>
```

   Здесь:

   - `<USER>` — имя пользователя хоста;
   - `<HOST>` — IP-адрес хоста.

2. Когда соединение с хостом установлено, запустите нужную команду HDFS. Например:

```
$ hdfs dfs -ls /
```

   Пример вывода команды `ls`:

```
Found 4 items
drwxrwxrwt   - yarn hadoop          0 2023-07-24 13:47 /logs
drwxr-xr-x   - hdfs hadoop          0 2023-08-03 17:49 /system
drwxrwxrwx   - hdfs hadoop          0 2023-08-30 19:51 /tmp
drwxr-xr-x   - hdfs hadoop          0 2023-07-24 13:47 /user
```

Дополнительную информацию о различиях между командами `hadoop fs`, `hadoop dfs` и `hdfs dfs` можно прочитать в статье Командная строка Hadoop.

Для некоторых команд требуется доступ на уровне администратора (пользователь `hdfs`). Если вы попытаетесь войти в систему как пользователь `hdfs`, потребуется пароль. Поскольку пароля по умолчанию для пользователя `hdfs` нет, войдите в систему как пользователь `root` на локальном хосте. Это позволит авторизоваться как пользователь `hdfs` без пароля.

HTTP REST API

Вы можете получить доступ к HDFS, используя WebHDFS REST API.

Чтобы выполнить запрос с помощью curl или браузера, структурируйте запрос следующим образом:

```
http://<HOST>:<HTTP_PORT>/webhdfs/v1/<PATH>?<AUTH>op=<COMMAND>
```

Здесь:

- `<HOST>` — IP-адрес хоста.
- `<HTTP_PORT>` — HTTP-порт активной NameNode.
- `<PATH>` — директория для выполнения команды.
- `<AUTH>` — запрос на аутентификацию в следующем формате: `user.name=<USER>&`, где `<USER>` — имя пользователя HDFS. Это опциональный параметр. Если этот параметр не указан, запрос будет отправлен от пользователя по умолчанию, если он настроен. В противном случае сервер выдаст ошибку.
- `<COMMAND>` — команда файловой системы.

Если у вас включен SSL, замените `webhdfs` на `swebhdfs`.

Аутентификация в HDFS REST API

Если настройки безопасности включены, метод аутентификации будет отличаться для каждой модели безопасности.

Если включен Kerberos SPNEGO, используйте параметры `--negotiate` и `-u`:

```
$ curl -i --negotiate -u : "http://<HOST>:<HTTP_PORT>/webhdfs/v1/<PATH>?op=<COMMAND>"
```

Если включена аутентификация с помощью токена делегирования Hadoop, передайте токен вместо имени пользователя:

```
$ curl -i "http://<HOST>:<HTTP_PORT>/webhdfs/v1/<PATH>?delegation=<TOKEN>&op=..."
```

Подробней о методах аутентификации читайте в статье Authentication for Hadoop HTTP web-consoles.

Пример запроса на подключение к HDFS для запуска команды `ls` может выглядеть так:

```
$ curl -i "http://127.0.0.1:14000/webhdfs/v1/?user.name=admin&op=LISTSTATUS"
```

Возможный вывод команды:

```
HTTP/1.1 200 OK
Date: Mon, 04 Sep 2023 11:50:00 GMT
Cache-Control: no-cache
Expires: Mon, 04 Sep 2023 11:50:00 GMT
Date: Mon, 04 Sep 2023 11:50:00 GMT
Pragma: no-cache
Content-Type: application/json
Set-Cookie: hadoop.auth="u=admin&p=admin&t=simple-dt&e=6593757/Hvhgbi/PxjQ="; Path=/; HttpOnly
Transfer-Encoding: chunked

{"FileStatuses":{"FileStatus":
[{"pathSuffix":"logs","type":"DIRECTORY","length":0,"owner":"yarn","group":"hadoop","permission":"1777","accessTime":0,"modificationTime":1690206465850,"blockSize":0,"replication":0},
{"pathSuffix":"system","type":"DIRECTORY","length":0,"owner":"hdfs","group":"hadoop","permission":"755","accessTime":0,"modificationTime":1691084968551,"blockSize":0,"replication":0},
{"pathSuffix":"tmp","type":"DIRECTORY","length":0,"owner":"hdfs","group":"hadoop","permission":"777","accessTime":0,"modificationTime":1693425076362,"blockSize":0,"replication":0},
{"pathSuffix":"user","type":"DIRECTORY","length":0,"owner":"hdfs","group":"hadoop","permission":"755","accessTime":0,"modificationTime":1690206432153,"blockSize":0,"replication":0}]}}
```

Библиотека libhdfs для C

Чтобы подключиться к удаленному хосту HDFS с помощью библиотеки libhdfs, необходимо настроить среду разработки и установить зависимости. В этой инструкции показано как написать, собрать и запустить программу для подключения к удаленному хосту HDFS.

В примере используются следующие инструменты разработки: текстовый редактор Vi, виртуальная машина с CentOS 7 и компилятор GCC. При использовании других инструментов будет необходимо изменить некоторые шаги.

Подготовительные шаги

Чтобы собрать и запустить следующий пример, установите:

- ADH (версия ADH на локальном хосте должна совпадать с версией на удаленном хосте). Вы также можете собрать его самостоятельно.
- Текстовый редактор.
- Компилятор C.
- JDK.
- Библиотеку `libhdfs.so` с заголовочными файлами. Вы также можете собрать библиотеку самостоятельно.

Создание программы на C

В приведенном ниже примере кода создается подключение к удаленному HDFS, выводится подтверждение при успешном подключении и возвращается сообщение об ошибке, если подключиться не удалось. Если вы успешно подключились к HDFS, но соединение вам больше не нужно, завершите его.

```
#include <hdfs.h> ①
#include <stdio.h>

int main(int argc, char **argv) {
    struct hdfsBuilder *builder = hdfsNewBuilder(); ②
    hdfsBuilderSetNameNode(builder, "hdfs://127.0.0.1:8020"); ③
    hdfsFS fs = hdfsBuilderConnect(builder); ④
    if (fs != NULL) {
      printf("Successfully connected to HDFS ");
    } else {
      printf("Failed to connect to HDFS ");
    }
}
```

① Импортируйте библиотеку *libhdfs*.

② Инициализируйте объект компоновщика для подключения к удаленному HDFS с помощью метода `hdfsBuilder`. Предпочтительно использовать этот метод вместо устаревшего `hdfsConnect`.

③ Установите параметры для подключения в формате URI `hdfs://<NN-IP>:<PORT>`, где `<NN-IP>` — это IP-адрес активной NameNode, а `<PORT>` — порт службы метаданных NameNode.

④ Вызовите `builder`, чтобы создать соединение.

Компиляция и запуск

1. Создайте скрипт компиляции для GCC и заполните переменные путей:

```
#!/bin/bash

export HADOOP_HOME=/usr/lib/hadoop ①
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.382.b05-1.el7_9.x86_64/jre ②
export CLASSPATH=`$HADOOP_HOME/bin/hadoop classpath --glob` ③

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$JAVA_HOME/jre/lib/amd64/server

gcc -std=c99 -lpthread main.c -I$HADOOP_HOME/include -L$HADOOP_HOME/lib/native -lhdfs -I$JAVA_HOME/include -I$JAVA_HOME/include/linux -L$JAVA_HOME/jre/lib/amd64/server -ljvm -o my_hdfs_program ④

./my_hdfs_program ⑤
```

① Укажите путь к библиотекам Hadoop.

② Определите путь к библиотекам JDK.

③ Введите правильный путь `CLASSPATH` с помощью команды classpath.

④ Скомпилируйте программу, используя правильные пути и дополнительные параметры.

⑤ Запустите программу.

2. Запустите скрипт компиляции:

```
$ ./<script-name>.sh
```

   Здесь `<script-name>` — название скрипта компиляции.

Java API

Чтобы подключиться к удаленному хосту HDFS с помощью Java API, вам необходимо настроить среду разработки и установить зависимости. В примере ниже создается подключение к удаленному HDFS, выводится подтверждение при успешном подключении и возвращается сообщение об ошибке, если подключиться не удалось.

Для выполнения примера используются следующие инструменты разработки: текстовый редактор Vi, виртуальная машина с CentOS 7 и компилятор Maven. При использовании других инструментов будет необходимо изменить некоторые шаги.

1. Установите Maven.

2. Напишите класс для создания соединения:

```
package my.example.hdfs;

import org.apache.hadoop.conf.Configuration; ①
import org.apache.hadoop.fs.FileSystem;

import java.net.URI;

public class Main {
   public static void main(String[] args) throws Exception {
      String hdfs_uri = "hdfs://127.0.0.1:8020"; ②

      Configuration conf = new Configuration(); ③

      conf.set("fs.defaultFS", hdfs_uri);

      try {
          FileSystem fs = FileSystem.get(URI.create(hdfs_uri), conf); ④
          System.out.println("Successfully connected to HDFS ");

          fs.close(); ⑤
      } catch (Exception e) {
          System.out.print("Failed to connect to HDFS ");
          System.out.println(e.getMessage());
      }
   }
}
```

① Импортируйте необходимые библиотеки.

② Установите параметры для подключения в формате URI `hdfs://<NN-IP>:<PORT>`, где `<NN-IP>` — это IP-адрес активной NameNode, а `<PORT>` — порт службы метаданных NameNode.

③ Инициализируйте объект файловой системы.

④ Вызовите FileSystem.get для создания соединения.

⑤ Завершите соединение, если оно больше не требуется.

3. Создайте файл *pom.xml* для сборки проекта с помощью Maven.

Пример файла pom.xml

```
<project>
        <modelVersion>4.0.0</modelVersion>
        <groupId>my.example.hdfs</groupId>
        <artifactId>my-hdfs-program</artifactId>
        <version>0.1.0-SNAPSHOT</version>
        <packaging>jar</packaging>
  <properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
  <dependencies>
        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-hdfs</artifactId>
            <version>2.7.1</version>
        </dependency>

        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-common</artifactId>
            <version>2.7.1</version>
        </dependency>

        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-core</artifactId>
            <version>1.2.1</version>
        </dependency>

        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-client</artifactId>
            <version>2.7.1</version>
        </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-assembly-plugin</artifactId>
            <version>3.1.0</version>
            <configuration>
              <archive>
                <manifest>
                  <addClasspath>true</addClasspath>
                  <mainClass>my.example.hdfs.Main</mainClass>
                </manifest>
              </archive>
                <descriptorRefs>
                    <descriptorRef>jar-with-dependencies</descriptorRef>
                </descriptorRefs>
            </configuration>
            <executions>
                <execution>
                    <id>assemble-all</id>
                    <phase>package</phase>
                    <goals>
                        <goal>single</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
      <plugin>
        <!-- Build an executable JAR -->
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-jar-plugin</artifactId>
        <version>3.1.0</version>
        <configuration>
          <archive>
            <manifest>
              <addClasspath>true</addClasspath>
              <mainClass>my.example.hdfs.Main</mainClass>
            </manifest>
          </archive>
        </configuration>
      </plugin>
    </plugins>
  </build>

</project>
```

4. Соберите проект:

```
$ mvn package
```

5. Запустите Java-класс:

```
$ java -jar <java-class>.jar
```

   Здесь `<java-class>` — имя Java-программы для подключения к HDFS.

Обзор HDFS UI

Елена Костюченко

Содержание

Назад к содержанию

- NameNode
  - Overview
  - DataNodes
  - Datanode Volume Failures
  - Snapshot
  - Startup Progress
  - Utilities
- DataNode
  - Overview
  - Utilities
- HttpFS Server

Веб-интерфейсы (UI) HDFS обеспечивают доступ к файловой системе и отображают информацию о ее состоянии. Пользовательские интерфейсы доступны по URL-адресу `http://<имя_хоста>:<порт>/`, где `<имя_хоста>` — это IP-адрес хоста или, если вы настроили перенаправление в файле `hosts`, FQDN хоста. А `<порт>` — это порт пользовательского интерфейса компонента HDFS.

Если у вас настроен сертификат SSL, используйте `https://` вместо `http://`.

HDFS предоставляет несколько веб-интерфейсов:

- интерфейс для каждой NameNode;
- интерфейс для каждой DataNode;
- интерфейс HttpFS.

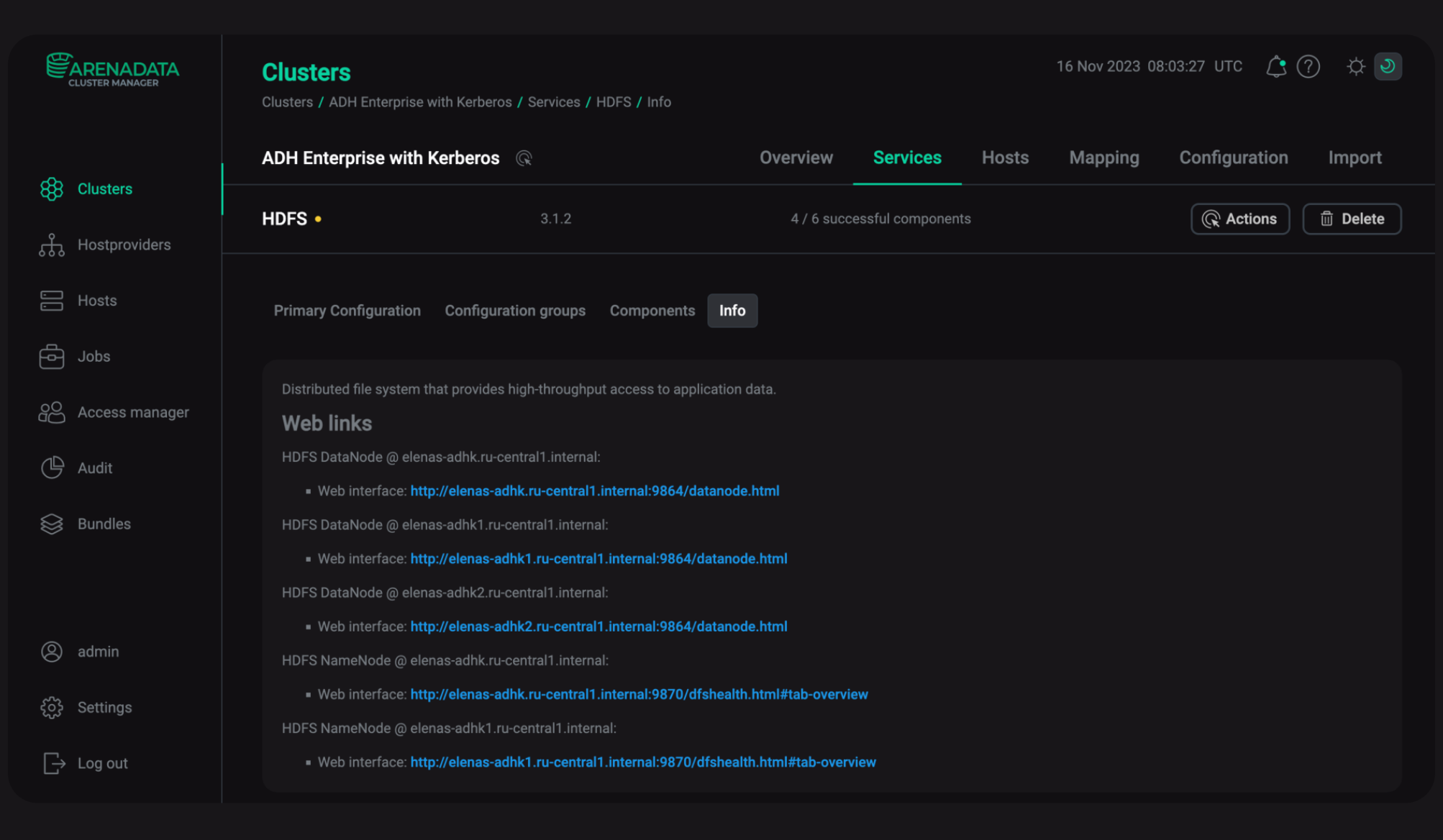
Порты веб-сервера по умолчанию перечислены в таблице ниже.

Компонент	Порт HTTP	Порт HTTPs
NameNode	9870	9871
DataNode	9864	9865
HttpFS	14000	14000

Чтобы изменить порт веб-сервера, измените значение соответствующего параметра `http-address` в конфигурационном файле *hdfs-site.xml*.

Ссылки на пользовательские интерфейсы HDFS также доступны в ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на HDFS.
3. Выберите ***Info***.


ARENADATA CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / ADH Enterprise with Kerberos / Services / HDFS / Info
16 Nov 2023 08:03:27 UTC
ADH Enterprise with Kerberos
Overview
Services
Hosts
Mapping
Configuration
Import
HDFS
3.1.2
4 / 6 successful components
Actions
Delete
Primary Configuration
Configuration groups
Components
Info

Distributed file system that provides high-throughput access to application data.

Web links

HDFS DataNode @ elenas-adhk.ru-central1.internal:
- Web interface: http://elenas-adhk.ru-central1.internal:9864/datanode.html

HDFS DataNode @ elenas-adhk1.ru-central1.internal:
- Web interface: http://elenas-adhk1.ru-central1.internal:9864/datanode.html

HDFS DataNode @ elenas-adhk2.ru-central1.internal:
- Web interface: http://elenas-adhk2.ru-central1.internal:9864/datanode.html

HDFS NameNode @ elenas-adhk.ru-central1.internal:
- Web interface: http://elenas-adhk.ru-central1.internal:9870/dfshealth.html#tab-overview

HDFS NameNode @ elenas-adhk1.ru-central1.internal:
- Web interface: http://elenas-adhk1.ru-central1.internal:9870/dfshealth.html#tab-overview

Ссылки на веб-интерфейс сервиса HDFS

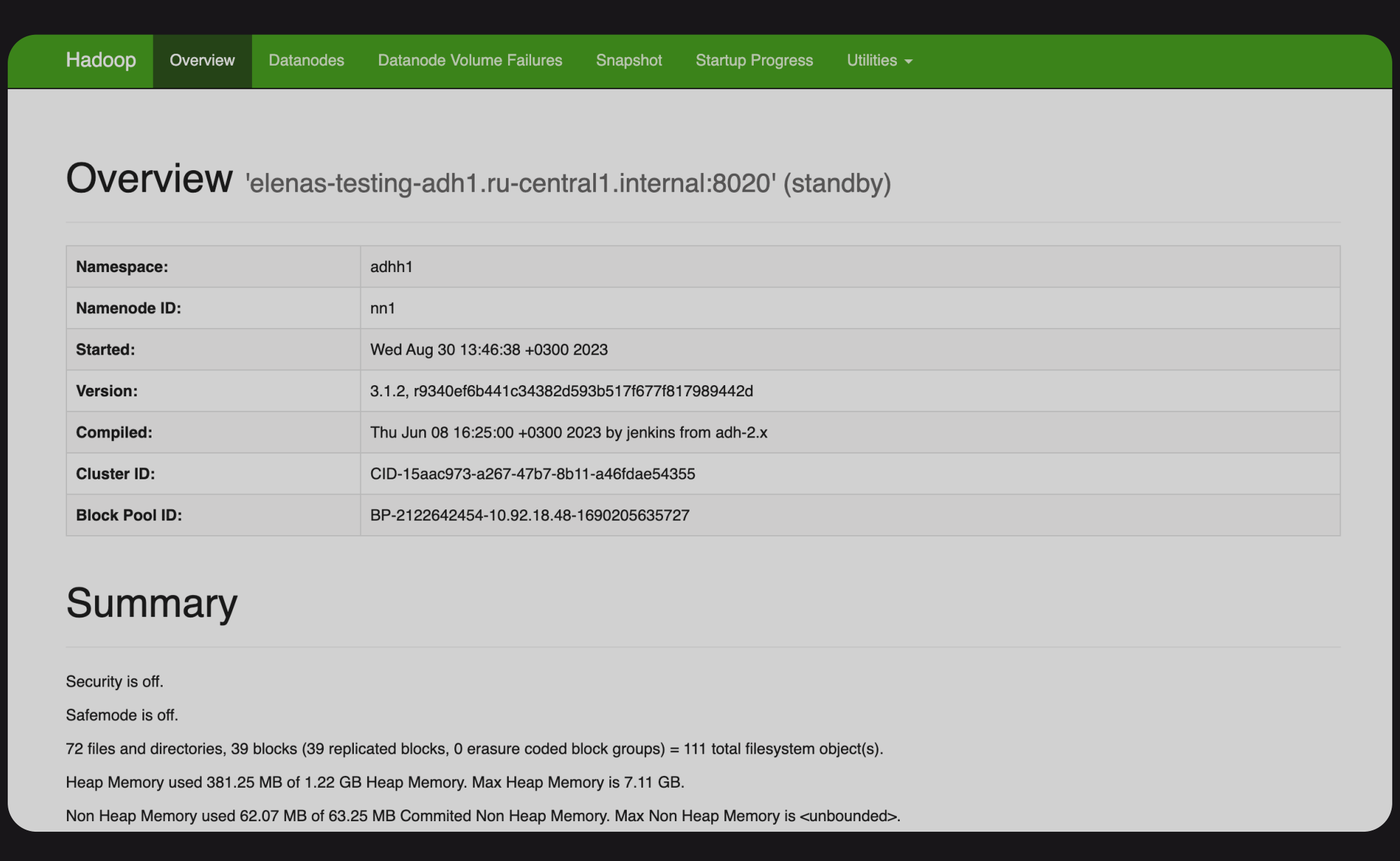
NameNode

Интерфейс NameNode включает информацию о конфигурации кластера, состоянии DataNode, веб-интерфейс файловой системы и логи.

Overview

На главной странице содержится следующая информация:

- ***Overview*** — основная информация о NameNode: тип (активная или резервная), имя хоста, идентификатор кластера и другая информация.
- ***Summary*** — информация о состоянии и конфигурации.
- ***NameNode Journal Status*** — информация о файле *edits*.
- ***NameNode Storage*** — информация о хранилище NameNode: каталог, тип и состояние.
- ***Типы хранилищ DFS*** — используемые типы хранилищ и их размер.


Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities

Overview 'elenas-testing-adh1.ru-central1.internal:8020' (standby)

Namespace:	adhh1
Namenode ID:	nn1
Started:	Wed Aug 30 13:46:38 +0300 2023
Version:	3.1.2, r9340ef6b441c34382d593b517f677f817989442d
Compiled:	Thu Jun 08 16:25:00 +0300 2023 by jenkins from adh-2.x
Cluster ID:	CID-15aac973-a267-47b7-8b11-a46fdae54355
Block Pool ID:	BP-2122642454-10.92.18.48-1690205635727

Summary

Security is off.

Safemode is off.

72 files and directories, 39 blocks (39 replicated blocks, 0 erasure coded block groups) = 111 total filesystem object(s).

Heap Memory used 381.25 MB of 1.22 GB Heap Memory. Max Heap Memory is 7.11 GB.

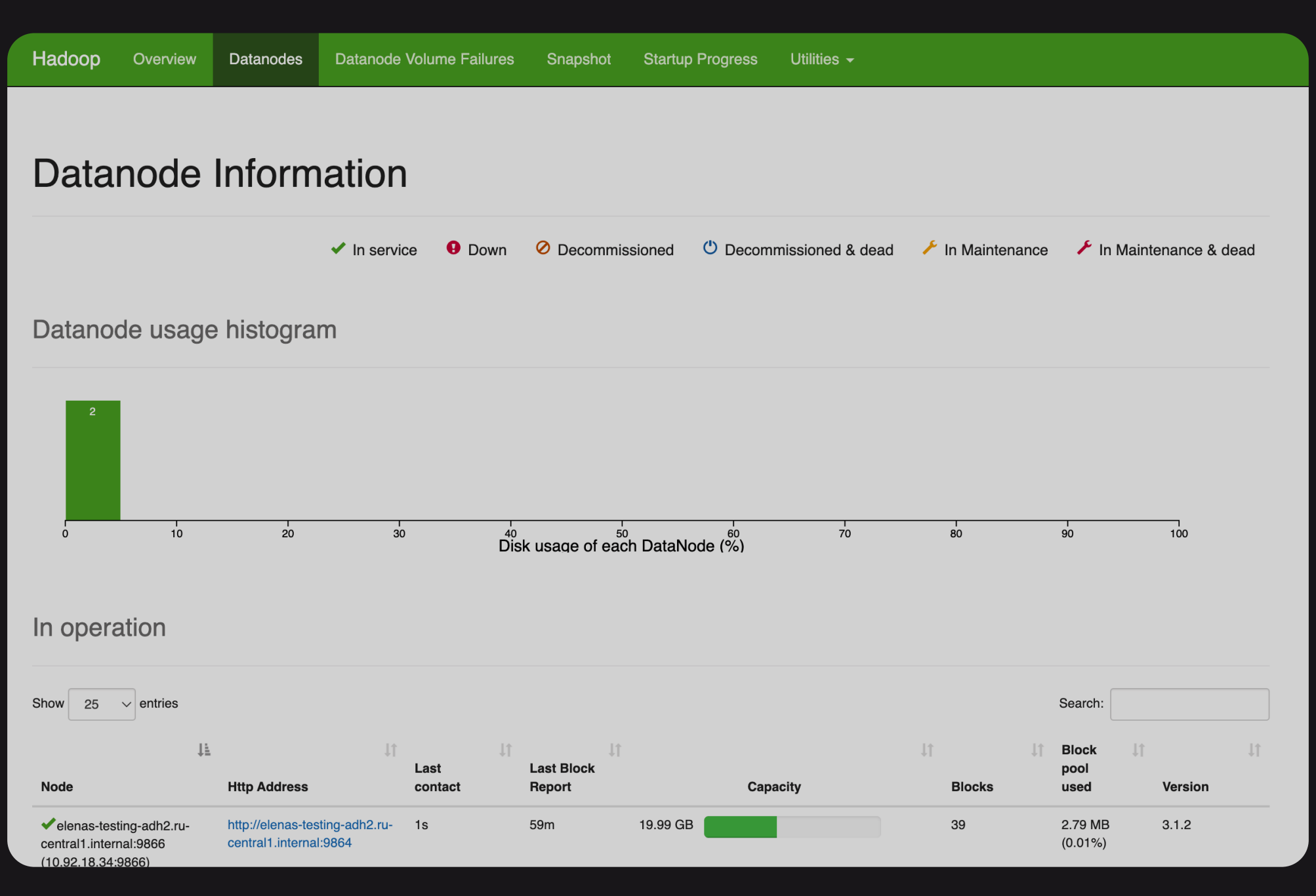
Non Heap Memory used 62.07 MB of 63.25 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Веб-интерфейс NameNode: главная страница

DataNodes

На странице ***Datanodes*** отображается гистограмма использования диска для каждой DataNode, информация о их состоянии и ссылки на веб-интерфейсы DataNodes.

Если в кластере есть ноды, которые готовятся к переводу в режим `Maintenance` или выводятся из эксплуатации, на странице будут отображаться имена их хостов и статистика по репликации блоков.


Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Datanode Information
✓ In service
Down
Decommissioned
Decommissioned & dead
In Maintenance
In Maintenance & dead
Datanode usage histogram
2
0
10
20
30
40
50
60
70
80
90
100
Disk usage of each DataNode (%)
In operation
Show 25 entries
Search:

Node	Http Address	Last contact	Last Block Report	Capacity		Blocks	Block pool used	Version
elenas-testing-adh2.ru-central1.internal:9866 (10.92.18.34:9866)	http://elenas-testing-adh2.ru-central1.internal:9864	1s	59m	19.99 GB		39	2.79 MB (0.01%)	3.1.2

Веб-интерфейс NameNode: страница Datanodes

У нод может быть одно из следующих состояний:

- ***In service*** — нода активна и регулярно отправляет heartbeat в NameNode.
- ***Down*** — нода не отправляла heartbeat более 10 минут.
- ***Decommissioned*** — нода находится в состоянии `Decommissioned`, но может быть восстановлена.
- ***Decommissioned & Dead*** — нода находилась в состоянии `Decommissioned` и теперь находится в состоянии `Down`.
- ***In Maintenance*** — нода находится в состоянии `Maintenance`.
- ***In Maintenance & Dead*** — нода была переведена в состояние `Maintenance` и теперь находится в состоянии `Down`.

Datanode Volume Failures

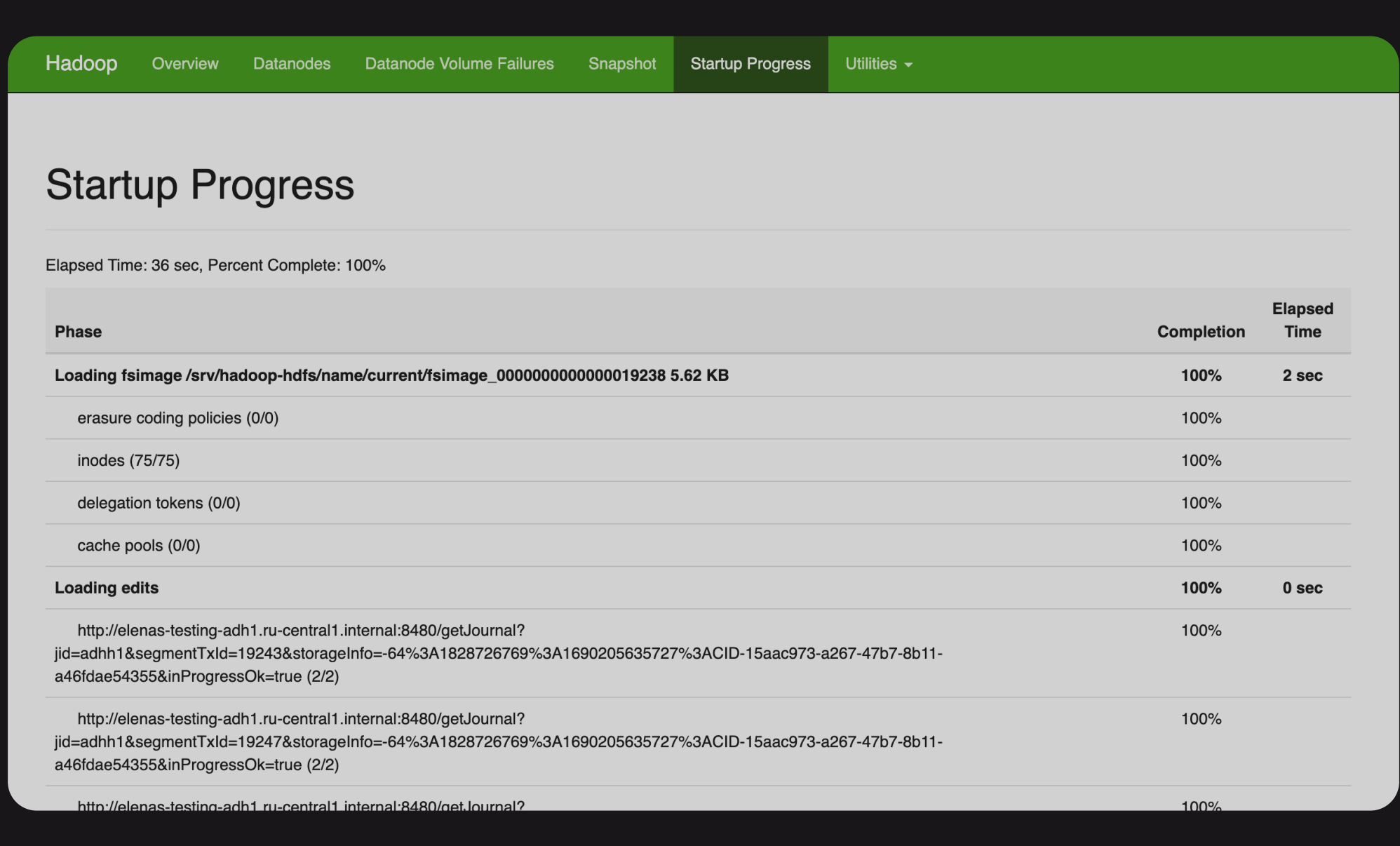
На странице представлена информация об ошибках типа `volume failure`, зарегистрированных на узлах данных.

Snapshot

Страница ***Snapshot*** предоставляет информацию о созданных снимках и о каталогах, для которых можно сделать снимок.

Startup Progress

На странице представлена информация о последней загруженной версии fsimage и внесенных изменениях (*edits*).


Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities

Startup Progress

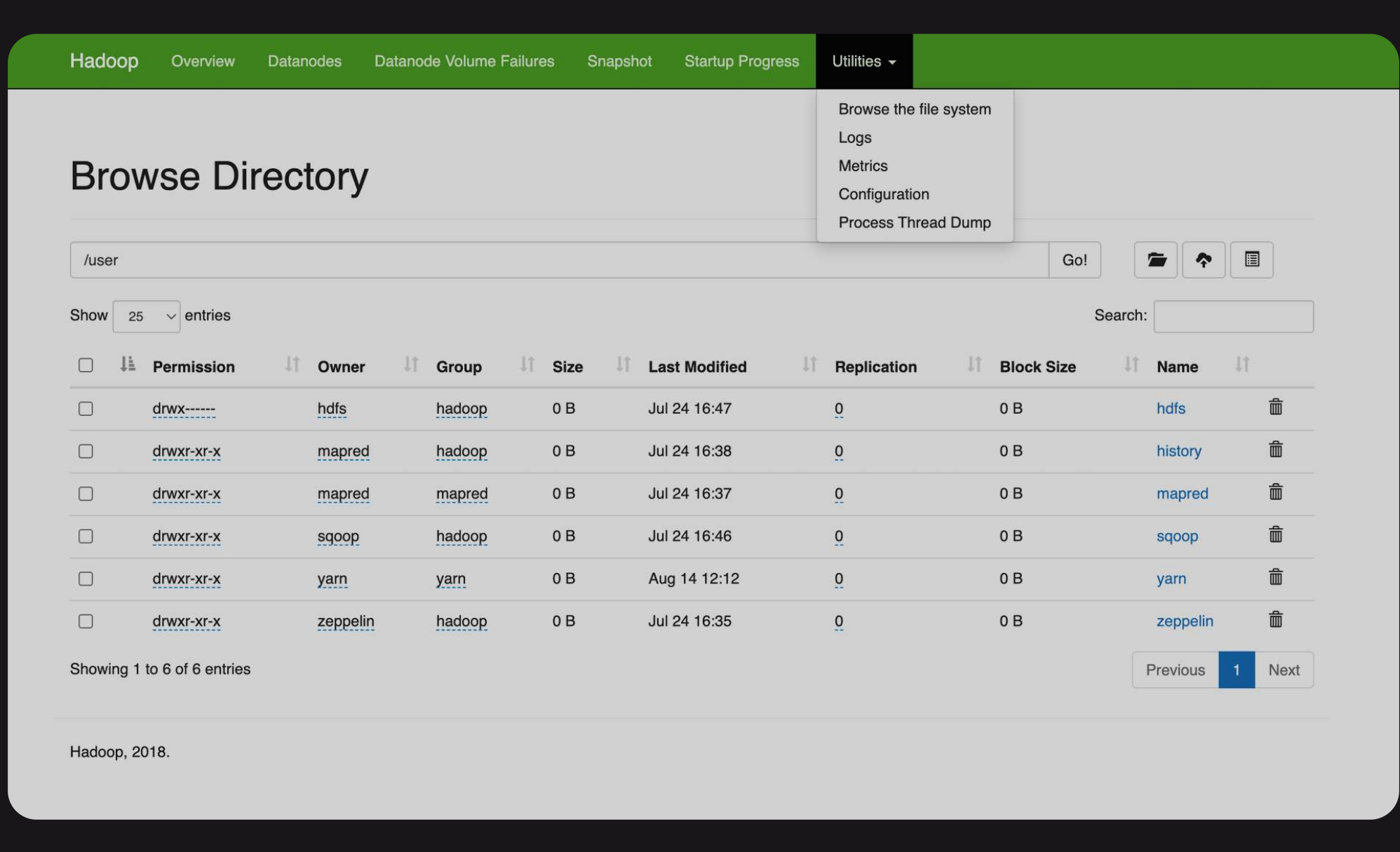
Elapsed Time: 36 sec, Percent Complete: 100%

Phase	Completion	Elapsed Time
Loading fsimage /srv/hadoop-hdfs/name/current/fsimage_0000000000000019238 5.62 KB	100%	2 sec
erasure coding policies (0/0)	100%	
inodes (75/75)	100%	
delegation tokens (0/0)	100%	
cache pools (0/0)	100%	
Loading edits	100%	0 sec
http://elenas-testing-adh1.ru-central1.internal:8480/getJournal?jid=adhh1&segmentTxId=19243&storageInfo=-64%3A1828726769%3A1690205635727%3ACID-15aac973-a267-47b7-8b11-a46fdae54355&inProgressOk=true (2/2)	100%	
http://elenas-testing-adh1.ru-central1.internal:8480/getJournal?jid=adhh1&segmentTxId=19247&storageInfo=-64%3A1828726769%3A1690205635727%3ACID-15aac973-a267-47b7-8b11-a46fdae54355&inProgressOk=true (2/2)	100%	

Веб-интерфейс NameNode: страница Startup Progress

Utilities

Страницы из меню ***Utilities*** предоставляют доступ к файловой системе, конфигурации, показывают информацию о доступных метриках, логи и список активных процессов.


Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Browse the file system
Logs
Metrics
Configuration
Process Thread Dump
Go!
Show 25 entries
Search:
Previous
1
Next

Browse Directory

/user

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwx------	hdfs	hadoop	0 B	Jul 24 16:47	0	0 B	hdfs
drwxr-xr-x	mapred	hadoop	0 B	Jul 24 16:38	0	0 B	history
drwxr-xr-x	mapred	mapred	0 B	Jul 24 16:37	0	0 B	mapred
drwxr-xr-x	sqoop	hadoop	0 B	Jul 24 16:46	0	0 B	sqoop
drwxr-xr-x	yarn	yarn	0 B	Aug 14 12:12	0	0 B	yarn
drwxr-xr-x	zeppelin	hadoop	0 B	Jul 24 16:35	0	0 B	zeppelin

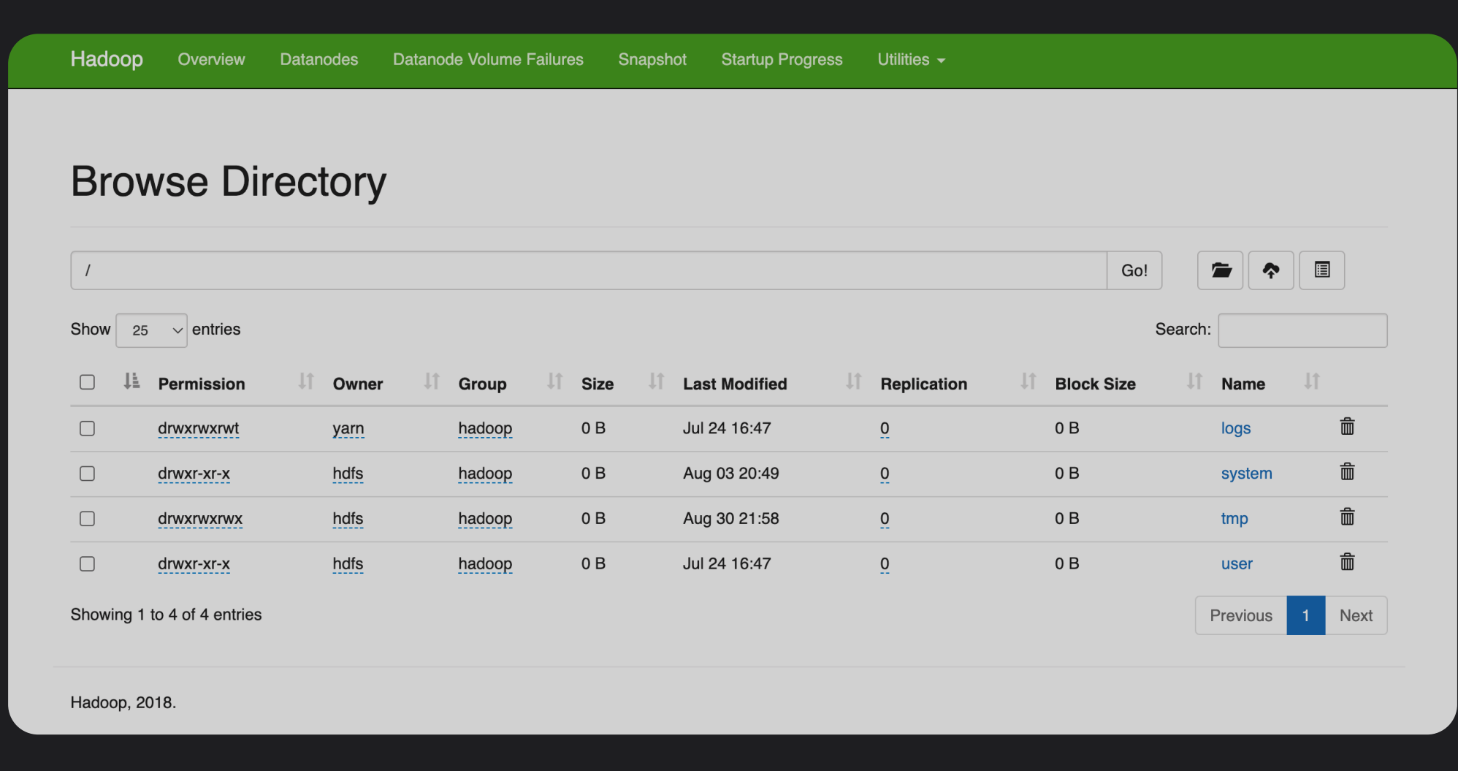
Showing 1 to 6 of 6 entries

Hadoop, 2018.

NameNode UI: Utilities menu

Browse the file system

Эта страница является веб-интерфейсом файловой системы. Доступ к ней возможен только через веб-интерфейс активной NameNode.


Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Go!
Show 25 entries
Search:
Previous
1
Next

Browse Directory

/

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxrwxrwt	yarn	hadoop	0 B	Jul 24 16:47	0	0 B	logs
drwxr-xr-x	hdfs	hadoop	0 B	Aug 03 20:49	0	0 B	system
drwxrwxrwx	hdfs	hadoop	0 B	Aug 30 21:58	0	0 B	tmp
drwxr-xr-x	hdfs	hadoop	0 B	Jul 24 16:47	0	0 B	user

Showing 1 to 4 of 4 entries

Hadoop, 2018.

Веб-интерфейс NameNode: страница Browse Directory

Для каждого файла или каталога доступна следующая информация:

- ***Permission*** — разрешения, установленные для файла или каталога. Нажмите на список разрешений, чтобы внести изменения.
- ***Owner*** — владелец файла или каталога.
- ***Group*** — группа безопасности для файла или каталога.
- ***Size*** — размер файла или каталога.
- ***Last Modified*** — дата внесения последних изменений в файле или каталоге.
- ***Replication*** — количество реплик для каждого блока. Нажмите на этот параметр, чтобы установить другой фактор репликации для файла. Фактор репликации по умолчанию — 3.
- ***Block Size*** — размер одного блока в файле. Размер блока по умолчанию — 128 МБ.
- ***Name*** — имя файла или каталога. При нажатии на имя каталога открывается дочерний каталог. Нажатие на имя файла открывает окно информации о файле, где вы можете скачать файл и посмотреть информацию о его блоках.

На этой же странице доступны следующие действия:

- ***Create directory*** — создание нового каталога.
- ***Upload files*** — загрузка файла через веб-интерфейс.
- ***Cut & Paste*** — вырезание или вставка.
- ***Delete*** — удаление файла или директории.

Для выполнения некоторых действий необходимо выдать дополнительные права для приложения веб-интерфейса (пользователь `dr.who`).

Logs

На странице ***Logs*** представлен список логов для компонентов DataNode, NameNode, JournalNode и службы HDFS ZKFC.

Metrics

На страницы ***Metrics*** представлен список метрик доступных для HDFS.

Configuration

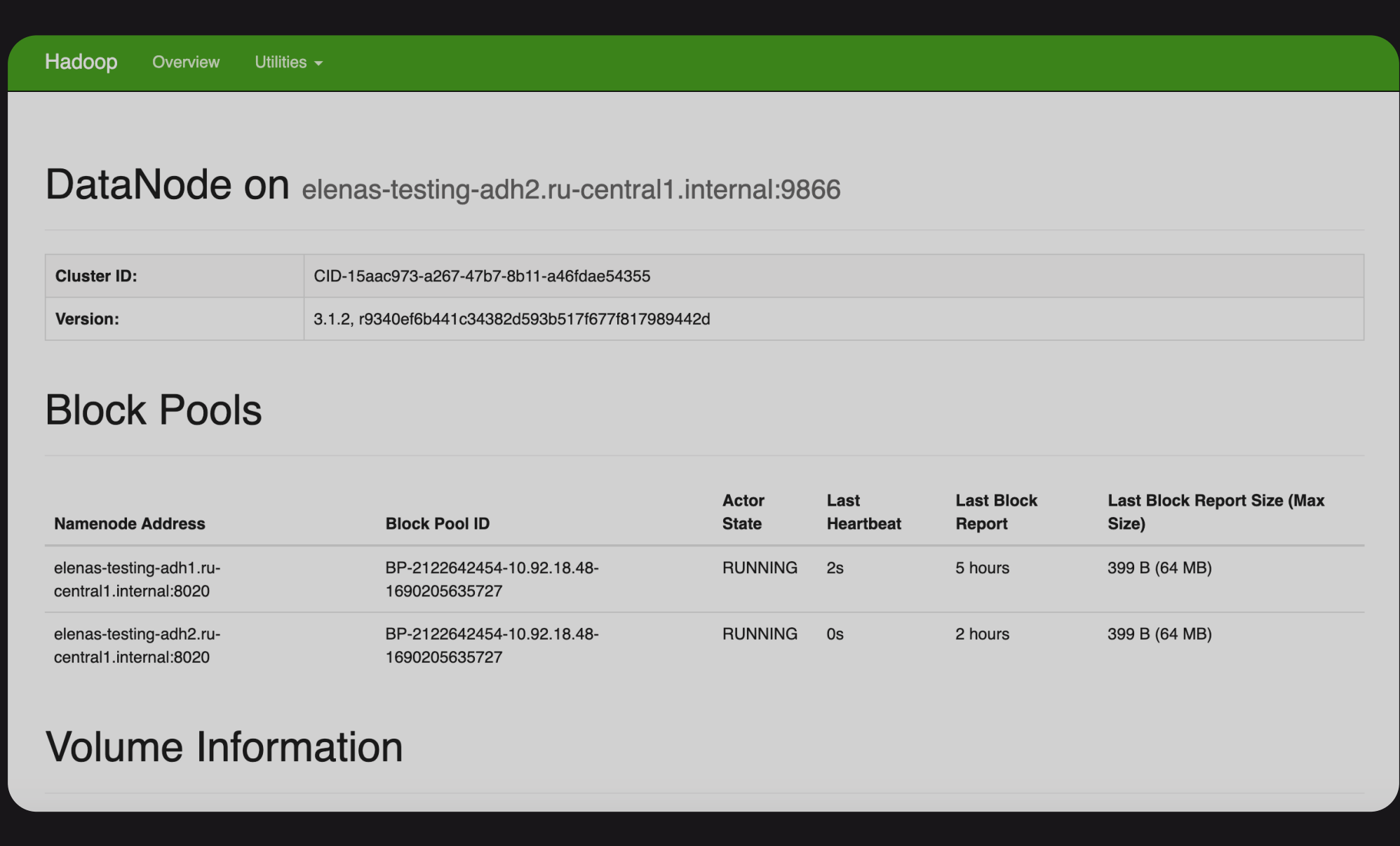
На странице ***Configuration*** представлен XML-файл со списком всех параметров конфигурации HDFS и их значения.

Process Thread Dump

На странице ***Process Thread Dump*** представлен список активных потоков и их статус.

DataNode

Интерфейс DataNode отображает базовую информацию о DataNode, ее статусе, конфигурации HDFS и логи.


Hadoop
Overview
Utilities

DataNode on elenas-testing-adh2.ru-central1.internal:9866

Cluster ID:	CID-15aac973-a267-47b7-8b11-a46fdae54355
Version:	3.1.2, r9340ef6b441c34382d593b517f677f817989442d

Block Pools

Namenode Address	Block Pool ID	Actor State	Last Heartbeat	Last Block Report	Last Block Report Size (Max Size)
elenas-testing-adh1.ru-central1.internal:8020	BP-2122642454-10.92.18.48-1690205635727	RUNNING	2s	5 hours	399 B (64 MB)
elenas-testing-adh2.ru-central1.internal:8020	BP-2122642454-10.92.18.48-1690205635727	RUNNING	0s	2 hours	399 B (64 MB)

Volume Information

Веб-интерфейс DataNode: главная страница

Overview

На главной странице содержится следующая информация:

- ***DataNode on*** — таблица с информацией о ноде: имя хоста DataNode, порт, идентификатор кластера и версия.
- ***Block Pools*** — информация о наборах блоков (Block Pools), размещенных на узле.
- ***Volume Information*** — информация о каталогах DataNode и использовании дискового пространства.

Utilities

Страница ***Utilities*** веб-интерфейса DataNode предоставляет ту же информацию, что и страница ***Utilities*** веб-интерфейса NameNode: конфигурация HDFS, доступные метрики, логи и активные процессы.

Logs

На странице ***Logs*** представлен список логов для компонентов DataNode, NameNode, JournalNode и службы HDFS ZKFC.

Metrics

На страницы ***Metrics*** представлен список метрик доступных для HDFS.

Configuration

На странице ***Configuration*** представлен XML-файл со списком всех параметров конфигурации HDFS и их значения.

Process Thread Dump

На странице ***Process Thread Dump*** представлен список активных потоков и их статус.

HttpFS Server

Пользовательский интерфейс сервера HttpFS — это веб-приложение, которое обеспечивает доступ к файловой системе через webhdfs REST HTTP API.

Интерфейс содержит список ссылок:

- ***/webhdfs/v1/?op=LISTSTATUS*** — список корневых каталогов.
- ***HttpFS configuration properties*** — XML-файл со списком всех параметров конфигурации HDFS и их значения.
- ***HttpFS JMX*** — список метрик доступных для HDFS.
- ***HttpFS log level*** — методы `GET` и `SET` для уровней логирования.
- ***HttpFS log files*** — список логов HttpFS-сервера.
- ***HttpFS stacks*** — список активных потоков и их статус.

Защита файлов в HDFS

Альберт Багдасарян

Содержание

Назад к содержанию

Права доступа к файлам и каталогам

Список управления доступом

Существуют два метода защиты файлов в HDFS:

- Настройка прав доступа в свойствах файлов.
- Создание списков доступа (Access Control Lists, ACLs).

Права доступа к файлам и каталогам

В HDFS можно ограничивать доступ к файлам и каталогам с использованием стандартной модели защиты, базирующейся на соответствущем стандарте POSIX с некоторыми модификациями. Вы можете предоставлять права на файл или каталог отдельно владельцу этого объекта, определенной группе пользователей и всем остальным пользователям. Для файлов используют следующие флаги:

- `r` — право на чтение.
- `w` — право на запись.

В отличие от других файловых систем, HDFS содержит только файлы с данными и не позволяет запустить какой-либо файл на выполнение. По этой причине в файлах HDFS не используют флаг типа `x`, который означал бы право на исполнение. По этой же причине HDFS не поддерживает флаги доступа типа `setUID` и `setGID`.

Следующие флаги гарантируют различные права доступа к каталогу HDFS:

- `r` — получить список содержимого каталога.
- `w` — создавать, переименовывать и удалять дочерние файлы и каталоги (одновременно требуется право `x`).
- `x` — входить внутрь каталога (команда `cd`) и получать доступ ко всем дочерним файлам и каталогам (то есть возможность использовать каталог как рабочий).

Как и для файлов, HDFS не предоставляет флаги `setUID` и `setGID` для каталогов.

Прежде чем начать управление правами необходимо получить доступ к командному интерфейсу (Command-line Interface, CLI) узла NameNode. После этого используйте команды, приведенные в следующих шагах:

1. Прочитайте содержимое корневого каталога:

```
$ hdfs dfs -ls /
```

Вывод на консоль приведен ниже:

```
-rw-r--r--   3 hdfs hadoop         20 2021-10-07 13:34 /hadoop
drwxrwxrwt   - yarn hadoop          0 2021-09-15 16:58 /logs
drwxr-xr-x   - hdfs hadoop          0 2021-10-12 13:17 /staging
drwxr-xr-x   - hdfs hadoop          0 2021-09-15 16:57 /system
drwxrwxrwx   - hdfs hadoop          0 2021-10-20 09:00 /tmp
drwxr-xr-x   - hdfs hadoop          0 2021-09-27 12:24 /user
```

Обратите внимание на каталог */logs* с флагами `drwxrwxrwt`, которые означают следующее:

- `d` обозначает каталог (directory).
- Первые три флага `rwx` определяют права владельца файла или каталога (в данном примере — `hdfs` и `yarn`).
- Вторая группа флагов `rwx` определяют права для группы пользователей (в примере — `hadoop`).
- Третья группа флагов `rwx` определяет права для остальных пользователей.

2. Определите права на каталог */tmp* как в следующем примере:

```
$ hdfs dfs -ls /tmp
```

Вывод на консоль приведен ниже:

```
drwxr-xr-x   - hdfs hadoop          0 2021-10-20 09:03 /tmp/hadoop01
-rw-r--r--   3 hdfs hadoop         20 2021-10-07 15:36 /tmp/test01
-rw-r--r--   3 hdfs hadoop         21 2021-10-08 08:40 /tmp/test02
-rw-r--r--   3 hdfs hadoop         19 2021-10-08 08:27 /tmp/test03
-rw-r--r--   3 hdfs hadoop          0 2021-10-20 09:00 /tmp/test04
```

В данном примере все файлы имеют одинаковую настройку, то есть владелец имеет право на чтение и запись, группа `hadoop` — на чтение и все остальные пользователи — на чтение.

3. Прежде чем изменить права на файл, используйте команды `sudo` и `su` (switch user) для переключения на авторизацию с правами пользователя по имени `hdfs` (подмена пользователя или имперсонация).

```
$ sudo -s
$ su - hdfs
```

Выполненные команды предоставили вам привилегии владельца используемых файлов. С этого моменты вы будете выполнять последующие команды с привилегиями пользователя `hdfs`.

4. Измените права доступа для группы (`g`) и других пользователей (`o`) путем добавления им права на запись (`+w`):

```
$ hdfs dfs -chmod  go+w /tmp/test01
```

Проверьте права на файл */tmp/test01*:

```
$ hdfs dfs -ls /tmp
```

Убедитесь в том, что права доступа для группы `hadoop` и остальных пользователей изменились:

```
drwxr-xr-x   - hdfs hadoop          0 2021-10-20 09:03 /tmp/hadoop01
-rw-rw-rw-   3 hdfs hadoop         20 2021-10-07 15:36 /tmp/test01
-rw-r--r--   3 hdfs hadoop         21 2021-10-08 08:40 /tmp/test02
-rw-r--r--   3 hdfs hadoop         19 2021-10-08 08:27 /tmp/test03
-rw-r--r--   3 hdfs hadoop          0 2021-10-20 09:00 /tmp/test04
```

Необходимые права назначены. Таким же образом вы можете изменять права на любые файлы и каталоги в HDFS. Вы также можете назначить другую группу или другого владельца на файл или каталог с помощью команд `chgrp` and `chown` соответственно.

Список управления доступом

Список управления доступом (Access Control Lists, ACLs) содержит права доступа пользователей и групп на файлы и каталоги. К его достоинствам относится более гибкое и наглядное представление процесса управления привилегиями.

Например, вы можете использовать ACL, если необходимо назначить права различным пользователям и группам на один файл или каталог.

Для разрешения использования ACL в HDFS, активируйте соответствующее свойство в вашем кластере Hyperwave.

Следующие шаги составляют процесс настройки ACL:

1. Разрешите ACL путем добавления следующего свойства в файле *hdfs-site.xml*:

```
dfs.namenode.acls.enabled : true
```

ПРИМЕЧАНИЕ

Для настройки ADH рекомендуем воспользоваться менеджером управления кластерами Arenadata Cluster Manager (ADCM).

2. Перезагрузите сервисы HDFS, чтобы применить измененные настройки. С помощью ADCM запустите задачу перенастройки и рестарта кластера.

ВАЖНО

Проверьте все ли ваши сервисы выделены зеленым цветом (нормальное состояние).

3. Используйте терминал на узле NameNode, чтобы выполнить следующую команду для чтения содержимого вашего тестового каталога HDFS, например */tmp*:

```
$ hdfs dfs -ls /tmp
```

Данная команда отображает все файлы и подкаталоги в каталоге */tmp*.

4. Выберите файл, например *test01*, и выполните следующую команду для считывания ACL этого файла:

```
$ hdfs dfs -getfacl /tmp/test01
```

Вывод на консоль приведен ниже:

```
# file: /tmp/test01
# owner: hdfs
# group: hadoop
user::rw-
group::rw-
other::rw-
```

5. Добавьте в список пользователя, например `yarn`, со всеми привилегиями на этот файл с помощью следующей команды:

```
$ hdfs dfs -setfacl -m user:yarn:rwx /tmp/test01
```

6. Повторите команду `hdfs dfs -getfacl /tmp/test01` для проверки обновленного списка, который теперь выглядит следующим образом:

```
# file: /tmp/test01
# owner: hdfs
# group: hadoop
user::rw-
user:yarn:rwx
group::rw-
mask::rwx
other::rw-
```

Добавленный пользователь `yarn` может читать и писать в тестовый файл. Этот пример демонстрирует, что ACL предоставляет более развитые возможности по управлению доступом к файлам и каталогам HDFS.

Копирование данных из локальной файловой системы в HDFS

Альберт Багдасарян

Назад к содержанию

Ниже представлен процесс копирования любых данных из локальной файловой системы в файловую систему Hadoop. Для этого сервис HDFS предоставляет специальную команду `hdfs dfs -put` согласно следующему синтаксису:

```
$ hdfs dfs -put <source_path> <destination_path>
```

ПРИМЕЧАНИЕ

Путь назначения должен указывать на каталог.

Для получения всех аргументов этой команды введите в терминале `hdfs dfs -put`.

Для копирования файлов с данными из локальной файловой системы в HDFS следуйте дальнейшим шагам:

1. Убедитесь, что файл с данными, например *file1.txt*, доступен в локальной файловой системе, например в вашем домашнем каталоге:

```
$ ls -a ~
```

Вывод на консоль приведен ниже:

```
.  ..  .ansible  .bash_history  .bash_logout  .bash_profile  .bashrc  file1.txt  .ssh  test
```

2. Проверьте содержимое корневого каталога HDFS:

```
$ hdfs dfs -ls /
```

Вывод на консоль приведен ниже:

```
Found 6 items
-rw-r--r--   3 hdfs hadoop         20 2021-10-07 13:34 /hadoop
drwxrwxrwt   - yarn hadoop          0 2021-09-15 16:58 /logs
drwxr-xr-x   - hdfs hadoop          0 2021-10-12 13:17 /staging
drwxr-xr-x   - hdfs hadoop          0 2021-09-15 16:57 /system
drwxrwxrwx   - hdfs hadoop          0 2021-10-20 09:00 /tmp
drwxr-xr-x   - hdfs hadoop          0 2021-09-27 12:24 /user
```

3. Создайте отдельный каталог в HDFS для копирования в него файлов, например:

```
$ hdfs dfs -mkdir /how_to_example
```

Обычно, если вы работаете как ординарный пользователь, HDFS возвращает сообщение об ошибке следующего содержания:

`mkdir: Permission denied: user=user_name, access=WRITE, inode="/user user_name":centos:centos:drwxr-xr-x`

Обычный пользователь не имеет права на запись в корневой каталог HDFS.

4. Для преодоления этого ограничения выполните требуемые команды, используя последовательно привилегии пользователей `root` и `hdfs`. Для этого прежде всего получите эти привилегии:

```
$ sudo -s
$ su - hdfs
```

ВНИМАНИЕ

Пользоваться привилегиями `root` небезопасно. Используйте их только при необходимости.

5. Используя привилегии пользователя `hdfs`, повторите попытку создания нового каталога с названием *hdfsDirectory* и убедитесь, что эта операция завершилась успешно:

```
$ hdfs dfs -mkdir /hdfsDirectory
$ hdfs dfs -ls /
```

Вывод на консоль приведен ниже:

```
Found 7 items
-rw-r--r--   3 hdfs hadoop         20 2021-10-07 13:34 /hadoop
drwxr-xr-x   - hdfs hadoop          0 2021-11-18 13:35 /hdfsDirectory
drwxrwxrwt   - yarn hadoop          0 2021-09-15 16:58 /logs
drwxr-xr-x   - hdfs hadoop          0 2021-10-12 13:17 /staging
drwxr-xr-x   - hdfs hadoop          0 2021-09-15 16:57 /system
drwxrwxrwx   - hdfs hadoop          0 2021-10-20 09:00 /tmp
drwxr-xr-x   - hdfs hadoop          0 2021-09-27 12:24 /user
```

6. Назначьте вашего обычного пользователя, например `admin`, владельцем нового каталога:

```
$ hdfs dfs -chown admin /hdfsDirectory
$ hdfs dfs -ls /
```

Вывод на консоль приведен ниже:

```
Found 7 items
-rw-r--r--   3 hdfs hadoop         20 2021-10-07 13:34 /hadoop
drwxr-xr-x   - admin hadoop          0 2021-11-18 13:35 /hdfsDirectory
drwxrwxrwt   - yarn hadoop          0 2021-09-15 16:58 /logs
drwxr-xr-x   - hdfs hadoop          0 2021-10-12 13:17 /staging
drwxr-xr-x   - hdfs hadoop          0 2021-09-15 16:57 /system
drwxrwxrwx   - hdfs hadoop          0 2021-10-20 09:00 /tmp
drwxr-xr-x   - hdfs hadoop          0 2021-09-27 12:24 /user
```

Теперь ваш обычный пользователь является владельцем каталога для приема файлов. Вернитесь в контекст этого пользователя:

```
$ exit
$ exit
```

7. Скопируйте файл *file1.txt* из локальной файловой системы в каталог HDFS и проверьте, что этот файл появился в каталоге назначения. Используйте команду `put` или `copyFromLocal`:

```
$ hdfs dfs -copyFromLocal ~/file1.txt /hdfsDirectory
$ hdfs dfs -ls /hdfsDirectory/file1.txt
```

Вывод на консоль приведен ниже:

```
-rw-r--r--   3 admin hadoop         29 2021-11-18 14:01 /hdfsDirectory/file1.txt
```

8. Назначьте право записи в каталог */hdfsDirectory* для всех пользователей, если вам необходимо, чтобы все они могли копировать данные в этот каталог:

```
$ hdfs dfs -chmod 777 /hdfsDirectory
```

Любой пользователь, имеющий доступ к HDFS, теперь может копировать свои файлы и каталоги с файлами в этот каталог.

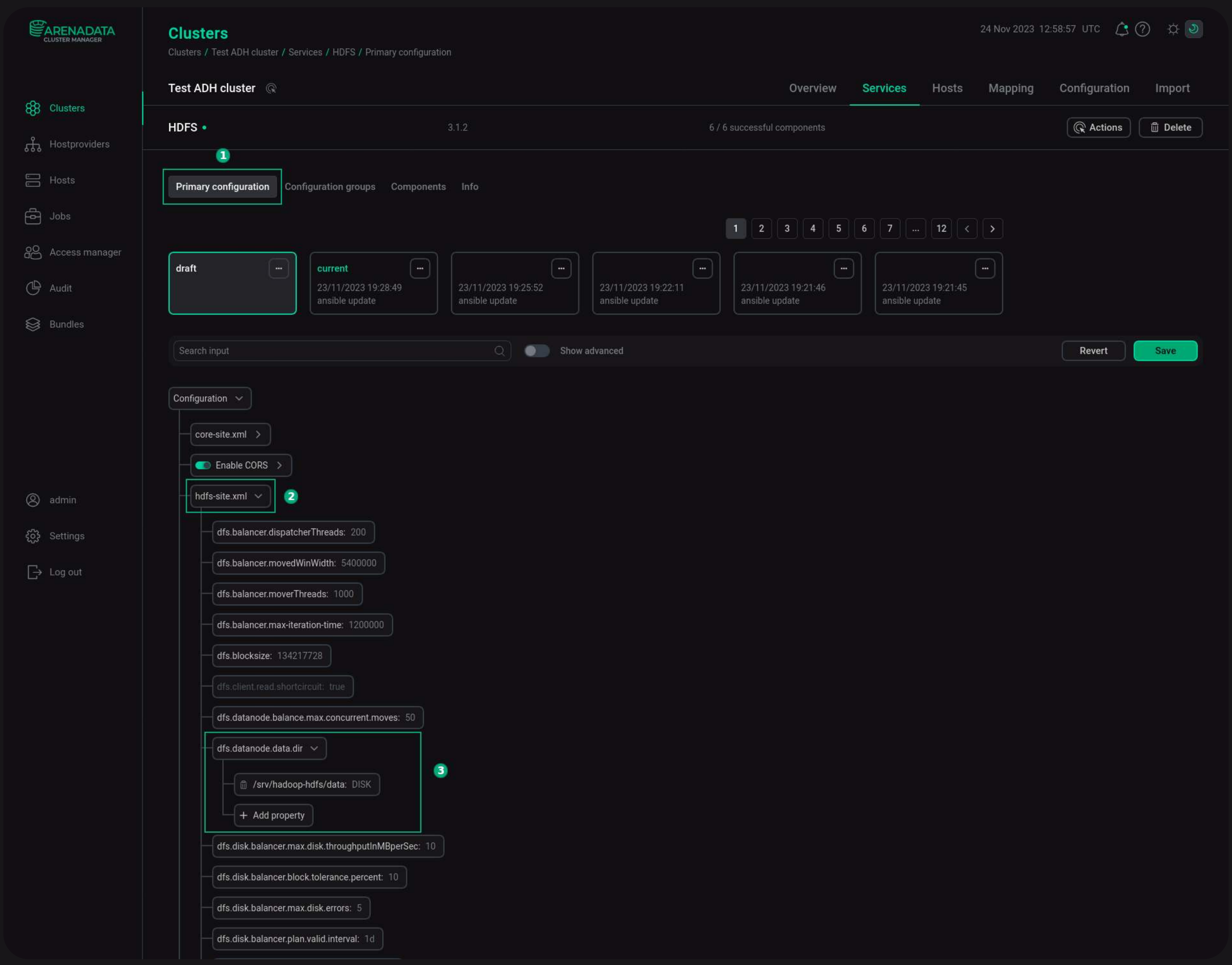
Выполнив представленные выше шаги, вы успешно скопировали файл *file1.txt* в HDFS а также обеспечили обычным пользователям право копировать их файлы в определенный каталог HDFS.

Добавление каталогов данных HDFS

Евгения Кузина

Назад к содержанию

Каталоги данных HDFS — это каталоги, в которых ноды DataNodes хранят свои блоки данных. Эти каталоги могут располагаться на разных устройствах, например, на примонтированных физических дисках. Локальные каталоги данных можно добавить в пользовательском интерфейсе ADCM. Откройте сервис HDFS кластера ADH. На вкладке ***Primary configuration***, в дереве конфигурационных параметров, найдите ноду ***hdfs-site.xml***, раскройте её и найдите параметр ***dfs.datanode.data.dir*** среди вложенных элементов.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / Test ADH cluster / Services / HDFS / Primary configuration
24 Nov 2023 12:58:57 UTC
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
HDFS
3.1.2
6 / 6 successful components
Actions
Delete
1
Primary configuration
Configuration groups
Components
Info
draft
current
Search input
Show advanced
Revert
Save
Configuration
core-site.xml
Enable CORS
hdfs-site.xml
2
3
Add property

23/11/2023 19:28:49 ansible update

23/11/2023 19:25:52 ansible update

23/11/2023 19:22:11 ansible update

23/11/2023 19:21:46 ansible update

23/11/2023 19:21:45 ansible update

dfs.balancer.dispatcherThreads: 200

dfs.balancer.movedWinWidth: 5400000

dfs.balancer.moverThreads: 1000

dfs.balancer.max-iteration-time: 1200000

dfs.blocksize: 134217728

dfs.client.read.shortcircuit: true

dfs.datanode.balance.max.concurrent.moves: 50

dfs.datanode.data.dir

/srv/hadoop-hdfs/data: DISK

dfs.disk.balancer.max.disk.throughputInMBperSec: 10

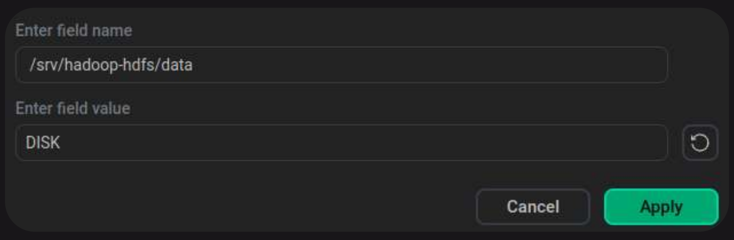
dfs.disk.balancer.block.tolerance.percent: 10

dfs.disk.balancer.max.disk.errors: 5

dfs.disk.balancer.plan.valid.interval: 1d

Добавление каталога данных HDFS

Нажмите вложенный элемент ***Add property***, чтобы добавить новый каталог данных. В появившемся окне укажите путь, например: `/srv/hadoop-hdfs/data`, и тип хранилища: `SSD`, `DISK`, `ARCHIVE` или `RAM_DISK`.


Cancel
Apply

Enter field name: /srv/hadoop-hdfs/data

Enter field value: DISK

Параметры каталога данных HDFS

Нажмите ***Apply***, чтобы применить настройки. Элемент с новым каталогом данных появится в дереве конфигурации.

Также необходимо выполнить следующие шаги:

1. Прежде чем добавить новое значение в параметр `dfs.datanode.data.dir`, необходимо указать владельца и разрешения для всех DataNodes следующим образом:

```
$ chown -R hdfs:hadoop /path/to/data/dir
$ chmod -R 755 /path/to/data/dir
```

   Где `/path/to/data/dir` это путь к каталогу данных или путь в локальной файловой системе, где этот каталог должен быть создан.

2. Каталоги данных должны иметь один и тот же путь и содержимое на всех DataNodes. Разные настройки разрешены только для групп с разной конфигурацией.

3. Для повышения производительности рекомендуется монтировать внешние каталоги данных в файле */etc/fstab* с параметром `noatime`, например:

```
/path/to/data/dir          ext3    defaults,noatime 1 1
```

Если разрешения для каталогов не установлены, при перезапуске HDFS произойдёт ошибка NameNode, например: `the reported blocks 0 needs additional 478 blocks to reach the threshold 0.9990 of total blocks 479`. Чтобы исправить эту ошибку, установите правильные разрешения для добавленных каталогов данных и перезапустите HDFS. Если это не поможет, убедитесь, чтобы все добавленные каталоги содержат данные с идентичной структурой на разных DataNodes. Набор файлов и каталогов должен быть одинаковым.

Замена диска без остановки DataNode

Елена Костюченко

Горячая замена диска (hot swapping) в HDFS — это процесс замены диска без приостановки работы DataNode.

Эта статья описывает, как подключить новый диск и добавить его в DataNode с помощью CLI и ADCM.

Чтобы добавить новый диск в DataNode:

1. Подключите диск к нужному хосту. Чтобы проверить, доступен ли диск в системе, выполните команду `lsblk`. Возможный вывод:

```
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda    253:0    0  100G  0 disk
├─vda1 253:1    0    1M  0 part
└─vda2 253:2    0  100G  0 part /
vdb    253:16   0   20G  0 disk
```

2. Создайте директорию для HDFS:

```
$ mkdir -p /srv/hadoop-hdfs/data1
```

3. Создайте файловую систему на диске:

```
$ mkfs.xfs /dev/vdb
```

4. Смонтируйте диск:

```
$ mount /dev/vdb /srv/hadoop-hdfs/data1
```

5. Добавьте новую файловую систему в *fstab*:

```
$ echo "/dev/vdb /srv/hadoop-hdfs/data1 xfs defaults,noatime 0 0" | sudo tee --append /etc/fstab
```

6. Смонтируйте файловую систему:

```
$ mount -a
```

   Чтобы проверить, успешно ли смонтирована система, используйте команду `lsblk`. Возможный вывод:

```
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda    253:0    0  100G  0 disk
├─vda1 253:1    0    1M  0 part
└─vda2 253:2    0  100G  0 part /
vdb    253:16   0   20G  0 disk /srv/hadoop-hdfs/data1
```

7. Сделайте `hdfs` владельцем новой директории и предоставьте следующие разрешения:

```
$ chown -R hdfs:hadoop /srv/hadoop-hdfs/data1
$ chmod -R 755 /srv/hadoop-hdfs/data1
```

8. Укажите созданную директорию в значении параметра `dfs.datanode.data.dir` конфигурации нужной DataNode. Вы можете сделать это вручную, отредактировав файл *hdfs-site.xml* на хосте DataNode, или создав группу конфигурации в ADCM. Дополнительную информацию о том, как изменить значение параметра `dfs.datanode.data.dir`, можно получить в статье Добавление каталогов данных HDFS. Чтобы узнать, как создать группу конфигурации, обратитесь к статье Настройка конфиг-групп.

> **ВНИМАНИЕ**
>
> Параметр `dfs.datanode.data.dir` необходимо изменить только для узла DataNode, на хосте которого создана нужная директория. Изменение этого параметра для всей системы может привести к ошибке.

9. Запустите реконфигурацию DataNode с помощью команды:

```
$ hdfs dfsadmin -reconfig datanode <HOST>:9867 start
```

   Здесь `<HOST>` — это FQDN хоста DataNode. Чтобы проверить статус задачи реконфигурации, используйте команду:

```
$ hdfs dfsadmin -reconfig datanode <HOST>:9867 status
```

Чтобы удалить диск из DataNode, отредактируйте свойство `dfs.datanode.data.dir` на хосте DataNode и запустите команду реконфигурации.

Удаление файлов и директорий

Елена Костюченко

Содержание

Назад к содержанию

Корзина

Удаление файлов через CLI

В HDFS можно удалять файлы и директории с помощью CLI, пользовательского интерфейса HDFS или разных API. Для получения информации об использовании этих утилит обратитесь к статье Подключение к HDFS.

При удалении файлов из HDFS удаляются сначала метаданные этих файлов и их реплик в NameNode. Если для блоков файла нет метаданых, они со временем удаляются.

Для внесения любых изменений в каталогах HDFS необходимо иметь права на запись (`w`). Если таких прав нет, сервис выдаст ошибку при попытке удалить файл или создать корзину. Более подробную информацию о том, как получить необходимые права доступа или изменить разрешения для каталога, можно получить в статье Защита файлов в HDFS.

Корзина

Корзина — это настройка для безопасного удаления данных. Если в конфигурационном файле *core-site.xml* включена опция корзины, то при удалении файла будет создана директория *.Trash* и файл переместится в нее. При этом в метаданных сохранится путь к блокам файла, поэтому файл удален не будет.

В ADH корзина включена по умолчанию. За работу корзины отвечают следующие параметры:

- `fs.trash.interval` — время в минутах, после которого файл будет удален навсегда. Чтобы отключить корзину, установите значение этого параметра равным `0`. Значение по умолчанию — `1440`.
- `fs.trash.checkpoint.interval` — периодичность удаления файлов из корзины в минутах. Значение по умолчанию — `60`.

Это значит, что каждые 60 минут (значение `fs.trash.checkpoint.interval`) проверяется как долго файлы находятся в корзине. Если какие-то из файлов находятся в корзине дольше, чем 1440 минут (значение `fs.trash.interval`), то они удаляются безвозвратно.

Удаление файлов через CLI

Чтобы удалить файл или директорию, выполните одну из следующих команд:

- rm — удаляет файл или директорию.
- rmdir — удаляет директорию.
- rmr — удаляет файл и директорию со всеми ее каталогами. Это устаревшая команда. Вместо нее рекомендуется использовать команду `rm` с опцией `R`.

Пример команды для удаления директории без перемещения его в корзину:

```
$ hadoop fs -rm -R -skipTrash <URI>
```

Здесь `<URI>` — путь к файлу или директории, которые нужно удалить.

Настройка WebHDFS

Елена Костюченко

Назад к содержанию

WebHDFS — это сервис, позволяющий взаимодействовать с HDFS через HTTP REST API.

В ADH-кластерах WebHDFS включен по умолчанию. Дополнительную информацию о том, как подключиться к HDFS с помощью HTTP REST API, читайте в статье Подключение к HDFS.

Для кластеров с включенной безопасностью (Kerberos SPNEGO) необходимо настроить дополнительные параметры в файле hdfs-site.xml:

1. Создайте нового принципала для сервиса HTTP:

```
$ addprinc -randkey HTTP/$<HOST>@$<REALM>.COM
```

   Здесь `<HOST>` — это FDQN или IP-адрес хоста с сервером NameNode, а `<REALM>` — имя Kerberos realm.

2. Создайте keytab для сервиса HTTP:

```
$ xst -norandkey -k /etc/security/spnego.service.keytab HTTP/$<HOST>
```

   Здесь `<HOST>` — это FDQN или IP-адрес хоста с сервером NameNode.

3. Убедитесь, что keytab для сервиса настроен:

```
$ klist -k -t /etc/security/spnego.service.keytab
```

4. Установите созданного ранее принципала в качестве значения параметра `dfs.web.authentication.kerberos.principal`, а keytab — как значение параметра в `dfs.web.authentication.kerberos.keytab`. Результат должен выглядеть так:

```
<property>
        <name>dfs.web.authentication.kerberos.principal</name>
        <value>HTTP/$<HOST>@$<REALM>.COM</value>
</property>
<property>
        <name>dfs.web.authentication.kerberos.keytab</name>
        <value>/etc/security/spnego.service.keytab</value>
</property>
```

5. Перезапустите серверы NameNode и DataNode.

Проверка файлов через fsck

Елена Костюченко

Назад к содержанию

В HDFS команда fsck выводит подробную информацию о файловой системе, ее состоянии, а также может выполнять некоторые действия с поврежденными файлами.

С помощью команды `fsck` вы можете:

- Получить информацию о репликации файлов: блоки, требующие репликации; расположение блоков по стойкам; размер блока и т.д.
- Посмотреть отчет по файловой системе: работоспособность системы; топология сети; количество узлов данных, стоек, каталогов и т.д.
- Вывести информацию по erasure coding.
- Управлять поврежденными файлами: удалить их или переместить в каталог */lost+found*.

Пример использования `fsck` для проверки каталога */tmp/* и вывода данных о соответствии политикам хранения:

```
$ hdfs fsck /tmp/ -storagepolicies
```

Примерный вывод команды:

```
Connecting to namenode via http://127.0.0.1:9870/fsck?ugi=admin&storagepolicies=1&path=%2Ftmp
FSCK started by admin (auth:SIMPLE) from /127.0.0.1 for path /tmp at Wed Sep 13 12:56:43 UTC 2023

.
/tmp/test/test_file.tgz:  Under replicated BP-2122642454-127.0.0.1-1690205635727:blk_1073741974_1150. Target Replicas is 3 but
found 2 live replica(s), 0 decommissioned replica(s), 0 decommissioning replica(s).
.
/tmp/test/test_file.zip:  Under replicated BP-2122642454-127.0.0.1-1690205635727:blk_1073741965_1141. Target Replicas is 3 but
found 2 live replica(s), 0 decommissioned replica(s), 0 decommissioning replica(s).

/tmp/test/test_file.zip:  Under replicated BP-2122642454-127.0.0.1-1690205635727:blk_1073741966_1142. Target Replicas is 3 but
found 2 live replica(s), 0 decommissioned replica(s), 0 decommissioning replica(s).

/tmp/test/test_file.zip:  Under replicated BP-2122642454-127.0.0.1-1690205635727:blk_1073741967_1143. Target Replicas is 3 but
found 2 live replica(s), 0 decommissioned replica(s), 0 decommissioning replica(s).

Status: HEALTHY
 Number of data-nodes:  2
 Number of racks:               1
 Total dirs:                    2
 Total symlinks:                0

Replicated Blocks:
 Total size:    303063571 B
 Total files:   2
 Total blocks (validated):      4 (avg. block size 75765892 B)
 Minimally replicated blocks:   4 (100.0 %)
 Over-replicated blocks:        0 (0.0 %)
 Under-replicated blocks:       4 (100.0 %)
 Mis-replicated blocks:         0 (0.0 %)
 Default replication factor:    3
 Average block replication:     2.0
 Missing blocks:                0
 Corrupt blocks:                0
 Missing replicas:              4 (33.333332 %)

Erasure Coded Block Groups:
 Total size:    0 B
 Total files:   0
 Total block groups (validated):        0
 Minimally erasure-coded block groups:  0
 Over-erasure-coded block groups:       0
 Under-erasure-coded block groups:      0
 Unsatisfactory placement block groups: 0
 Average block group size:      0.0
 Missing block groups:          0
 Corrupt block groups:          0
 Missing internal blocks:       0

Blocks satisfying the specified storage policy:
Storage Policy                  # of blocks       % of blocks
DISK:2(HOT)                        4             100.0000%

All blocks satisfy specified storage policy.
FSCK ended at Wed Sep 13 12:56:43 UTC 2023 in 7 milliseconds


The filesystem under path '/tmp' is HEALTHY
```

Локальное чтение данных

Елена Костюченко

Назад к содержанию

Локальное чтение данных (Short-circuit read) — это режим чтения данных в HDFS, который позволяет читать данные напрямую, в обход обращения к серверу DataNode. Такой подход позволяет ускорить процесс чтения, что может значительно улучшить работу приложений.

В ADH локальное чтение включено по умолчанию. Чтобы использовать его, клиент должен находиться на одном хосте с данными, к которым будет обращаться.

В конфигурационном файле hdfs-site.xml по умолчанию установлены следующие параметры локального чтения:

- `dfs.client.read.shortcircuit` — `True`,
- `dfs.domain.socket.path` — `/var/lib/hadoop-hdfs/dn_socket`.

При необходимости вы можете изменить значения параметров через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на HDFS.
3. Найдите параметр, измените его значение и нажмите ***Apply***.
4. Подтвердите изменения в конфигурации HDFS, нажав ***Save***.
5. В меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Чтобы посмотреть статистику по локальному чтению данных, откройте страницу ***Metrics*** в графическом интерфейсе DataNode и найдите параметр ***ReadsFromLocalClient***. В нем указывается количество локальных чтений.

Использование distcp

Елена Костюченко

Содержание

Назад к содержанию

Обзор команды
Копирование данных между кластерами с Kerberos
- Подготовительные шаги
- Установка доверия
- Использование distcp

Обзор команды

Команда distcp (Distributed Copy) используется для копирования данных. Ее основное преимущество заключается в использовании MapReduce для распределенного и параллельного копирования данных, что позволяет эффективней обрабатывать большие объемы данных. Для получения дополнительной информации о MapReduce можно обратиться к документации MapReduce.

Вы также можете использовать `distcp` для загрузки, скачивания и обновления данных в объектных хранилищах, таких как S3.

Для корректного использования `distcp` необходимо выполнить несколько условий:

- В путях копируемых данных не должно быть коллизий, иначе сервис выдаст ошибку. Будьте осторожны при одновременном копировании из нескольких источников.
- Источник копирования и место назначения должны использовать одну и ту же версию ADH. Чтобы скопировать данные между кластерами с разными версиями ADH, используйте протокол WebHDFS.
- Любые операции с копируемыми файлами должны быть приостановлены. Если во время работы `distcp` клиент записывает в копируемый файл или пытается перезаписать файл, в который копируются данные, копирование, скорее всего, завершится с ошибкой.

Пример команды для копирования данных между кластерами с использованием опции `-update`:

```
$ hadoop distcp -update hdfs://<HOST1>:8020/tmp hdfs://<HOST2>:8020/tmp
```

Пример вывода

```
2023-09-15 07:43:46,260 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFolder=true,
deleteMissing=false, ignoreFailures=false, overwrite=false, append=false, useDiff=false, useRdiff=false, fromSnapshot=null,
toSnapshot=null, skipCRC=false, blocking=true, numListstatusThreads=0, maxMaps=20, mapBandwidth=0.0,
copyStrategy='uniformsize', preserveStatus=[BLOCKSIZE], atomicWorkPath=null, logPath=null, sourceFileListing=null,
sourcePaths=[hdfs://localhost:8020/tmp], targetPath=hdfs://localhost:8020/tmp, filtersFile='null', blocksPerChunk=0,
copyBufferSize=8192, verboseLog=false}, sourcePaths=[hdfs://localhost:8020/tmp], targetPathExists=true, preserveRawXattrsfalse
2023-09-15 07:43:46,429 WARN impl.MetricsConfig: Cannot locate configuration: tried hadoop-metrics2-jobtracker.
properties,hadoop-metrics2.properties
2023-09-15 07:43:46,520 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2023-09-15 07:43:46,520 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2023-09-15 07:43:46,765 INFO tools.SimpleCopyListing: Paths (files+dirs) cnt = 6; dirCnt = 2
2023-09-15 07:43:46,765 INFO tools.SimpleCopyListing: Build file listing completed.
2023-09-15 07:43:46,766 INFO Configuration.deprecation: io.sort.mb is deprecated. Instead, use mapreduce.task.io.sort.mb
2023-09-15 07:43:46,766 INFO Configuration.deprecation: io.sort.factor is deprecated. Instead, use mapreduce.task.io.sort.factor
2023-09-15 07:43:46,793 INFO tools.DistCp: Number of paths in the copy list: 6
2023-09-15 07:43:46,804 INFO tools.DistCp: Number of paths in the copy list: 6
2023-09-15 07:43:46,812 WARN impl.MetricsSystemImpl: JobTracker metrics system already initialized!
2023-09-15 07:43:46,884 INFO mapreduce.JobSubmitter: number of splits:1
2023-09-15 07:43:47,324 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1206860640_0001
2023-09-15 07:43:47,326 INFO mapreduce.JobSubmitter: Executing with tokens: []
2023-09-15 07:43:47,459 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2023-09-15 07:43:47,459 INFO tools.DistCp: DistCp job-id: job_local1206860640_0001
2023-09-15 07:43:47,460 INFO mapreduce.Job: Running job: job_local1206860640_0001
2023-09-15 07:43:47,462 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2023-09-15 07:43:47,468 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-09-15 07:43:47,468 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output
directory:false, ignore cleanup failures: false
2023-09-15 07:43:47,469 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.tools.mapred.CopyCommitter
2023-09-15 07:43:47,501 INFO mapred.LocalJobRunner: Waiting for map tasks
2023-09-15 07:43:47,502 INFO mapred.LocalJobRunner: Starting task: attempt_local1206860640_0001_m_000000_0
2023-09-15 07:43:47,525 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-09-15 07:43:47,525 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output
directory:false, ignore cleanup failures: false
2023-09-15 07:43:47,588 INFO mapred.Task:  Using ResourceCalculatorProcessTree : [ ]
2023-09-15 07:43:47,591 INFO mapred.MapTask: Processing split: file:/tmp/hadoop/mapred/staging/admin1599424582/.
staging/_distcp15296078/fileList.seq:0+1192
2023-09-15 07:43:47,597 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2023-09-15 07:43:47,597 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output
directory:false, ignore cleanup failures: false
2023-09-15 07:43:47,605 INFO mapred.CopyMapper: Copying hdfs://localhost:8020/tmp to hdfs://localhost:8020/tmp/tmp
2023-09-15 07:43:47,640 INFO mapred.CopyMapper: Copying hdfs://localhost:8020/tmp/test/hadoop-jvm.tgz to
hdfs://localhost:8020/tmp/tmp/test/hadoop-jvm.tgz
2023-09-15 07:43:47,652 INFO mapred.RetriableFileCopyCommand: Creating temp file: hdfs://localhost:8020/tmp/.distcp.tmp.
attempt_local1206860640_0001_m_000000_0
2023-09-15 07:43:48,462 INFO mapreduce.Job: Job job_local1206860640_0001 running in uber mode : false
2023-09-15 07:43:48,463 INFO mapreduce.Job:  map 0% reduce 0%
2023-09-15 07:43:48,698 INFO mapred.CopyMapper: Copying hdfs://localhost:8020/tmp/test to hdfs://localhost:8020/tmp/tmp/test
2023-09-15 07:43:48,708 INFO mapred.CopyMapper: Copying hdfs://localhost:8020/tmp/test/pom.xml to
hdfs://localhost:8020/tmp/tmp/test/pom.xml
2023-09-15 07:43:48,717 INFO mapred.RetriableFileCopyCommand: Creating temp file: hdfs://localhost:8020/tmp/.distcp.tmp.
attempt_local1206860640_0001_m_000000_0
2023-09-15 07:43:48,900 INFO mapred.CopyMapper: Copying hdfs://localhost:8020/tmp/test/03_02.mov to
hdfs://localhost:8020/tmp/tmp/test/03_02.mov
2023-09-15 07:43:48,909 INFO mapred.RetriableFileCopyCommand: Creating temp file: hdfs://localhost:8020/tmp/.distcp.tmp.
attempt_local1206860640_0001_m_000000_0
2023-09-15 07:43:59,535 INFO mapred.LocalJobRunner: 66.2% Copying hdfs://localhost:8020/tmp/test/03_02.mov to
hdfs://localhost:8020/tmp/tmp/test/03_02.mov [674.9M/1018.9M] > map
2023-09-15 07:44:00,470 INFO mapreduce.Job:  map 82% reduce 0%
2023-09-15 07:44:05,536 INFO mapred.LocalJobRunner: 100.0% Copying hdfs://localhost:8020/tmp/test/03_02.mov to
hdfs://localhost:8020/tmp/tmp/test/03_02.mov [1018.9M/1018.9M] > map
2023-09-15 07:44:07,014 INFO mapred.CopyMapper: Copying hdfs://localhost:8020/tmp/test/test_file.zip to
hdfs://localhost:8020/tmp/tmp/test/test_file.zip
2023-09-15 07:44:07,023 INFO mapred.RetriableFileCopyCommand: Creating temp file: hdfs://localhost:8020/tmp/.distcp.tmp.
attempt_local1206860640_0001_m_000000_0
2023-09-15 07:44:11,537 INFO mapred.LocalJobRunner: 88.6% Copying hdfs://localhost:8020/tmp/test/test_file.zip to
hdfs://localhost:8020/tmp/tmp/test/test_file.zip [256.1M/289.0M] > map
2023-09-15 07:44:12,412 INFO mapred.LocalJobRunner: 88.6% Copying hdfs://localhost:8020/tmp/test/test_file.zip to
hdfs://localhost:8020/tmp/tmp/test/test_file.zip [256.1M/289.0M] > map
2023-09-15 07:44:12,412 INFO mapred.Task: Task:attempt_local1206860640_0001_m_000000_0 is done. And is in the process of
committing
2023-09-15 07:44:12,413 INFO mapred.LocalJobRunner: 88.6% Copying hdfs://localhost:8020/tmp/test/test_file.zip to
hdfs://localhost:8020/tmp/tmp/test/test_file.zip [256.1M/289.0M] > map
2023-09-15 07:44:12,414 INFO mapred.Task: Task attempt_local1206860640_0001_m_000000_0 is allowed to commit now
2023-09-15 07:44:12,415 INFO output.FileOutputCommitter: Saved output of task 'attempt_local1206860640_0001_m_000000_0' to
file:/tmp/hadoop/mapred/staging/admin1599424582/.staging/_distcp15296078/_logs
2023-09-15 07:44:12,416 INFO mapred.LocalJobRunner: 100.0% Copying hdfs://localhost:8020/tmp/test/test_file.zip to
hdfs://localhost:8020/tmp/tmp/test/test_file.zip [289.0M/289.0M]
2023-09-15 07:44:12,417 INFO mapred.Task: Task 'attempt_local1206860640_0001_m_000000_0' done.
2023-09-15 07:44:12,424 INFO mapred.Task: Final Counters for attempt_local1206860640_0001_m_000000_0: Counters: 25
        File System Counters
                FILE: Number of bytes read=191251
                FILE: Number of bytes written=697086
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=1371500500
                HDFS: Number of bytes written=1371500500
                HDFS: Number of read operations=48
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=15
        Map-Reduce Framework
                Map input records=6
                Map output records=0
                Input split bytes=150
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=41
                Total committed heap usage (bytes)=326631424
        File Input Format Counters
                Bytes Read=1232
        File Output Format Counters
                Bytes Written=8
        DistCp Counters
                Bandwidth in Btyes=57145854
                Bytes Copied=1371500500
                Bytes Expected=1371500500
                Files Copied=4
                DIR_COPY=2
2023-09-15 07:44:12,424 INFO mapred.LocalJobRunner: Finishing task: attempt_local1206860640_0001_m_000000_0
2023-09-15 07:44:12,425 INFO mapred.LocalJobRunner: map task executor complete.
2023-09-15 07:44:12,436 INFO mapred.CopyCommitter: About to preserve attributes: B
2023-09-15 07:44:12,441 INFO mapred.CopyCommitter: Preserved status on 0 dir entries on target
2023-09-15 07:44:12,441 INFO mapred.CopyCommitter: Cleaning up temporary work folder:
file:/tmp/hadoop/mapred/staging/admin1599424582/.staging/_distcp15296078
2023-09-15 07:44:12,475 INFO mapreduce.Job:  map 100% reduce 0%
2023-09-15 07:44:12,476 INFO mapreduce.Job: Job job_local1206860640_0001 completed successfully
2023-09-15 07:44:12,482 INFO mapreduce.Job: Counters: 25
        File System Counters
                FILE: Number of bytes read=191251
                FILE: Number of bytes written=697086
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=1371500500
                HDFS: Number of bytes written=1371500500
                HDFS: Number of read operations=48
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=15
        Map-Reduce Framework
                Map input records=6
                Map output records=0
                Input split bytes=150
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=41
                Total committed heap usage (bytes)=326631424
        File Input Format Counters
                Bytes Read=1232
        File Output Format Counters
                Bytes Written=8
        DistCp Counters
                Bandwidth in Btyes=57145854
                Bytes Copied=1371500500
                Bytes Expected=1371500500
                Files Copied=4
                DIR_COPY=2
```

Копирование данных между кластерами с Kerberos

Чтобы запустить `distcp` в кластере из другого Kerberos realm, необходимо установить доверие между кластерами. Для этого необходимо внести изменения в файлы конфигурации Kerberos. Для получения дополнительной информации о том, как настроить доверие между кластерами вручную, читайте в статье Setting up a Realm Trust.

В этом примере доверие настраивается через ADCM, а `distcp` вызывается с использованием параметра ***Additional nameservices***. Более подробную информацию о концепции Internal nameservice можно получить в статье Управление сервисом HDFS через ADCM.

Подготовительные шаги

Предполагается, что Kerberos уже установлен и настроен. Чтобы узнать больше о том, как установить Kerberos, обратитесь к статье Настройки сервера Kerberos.

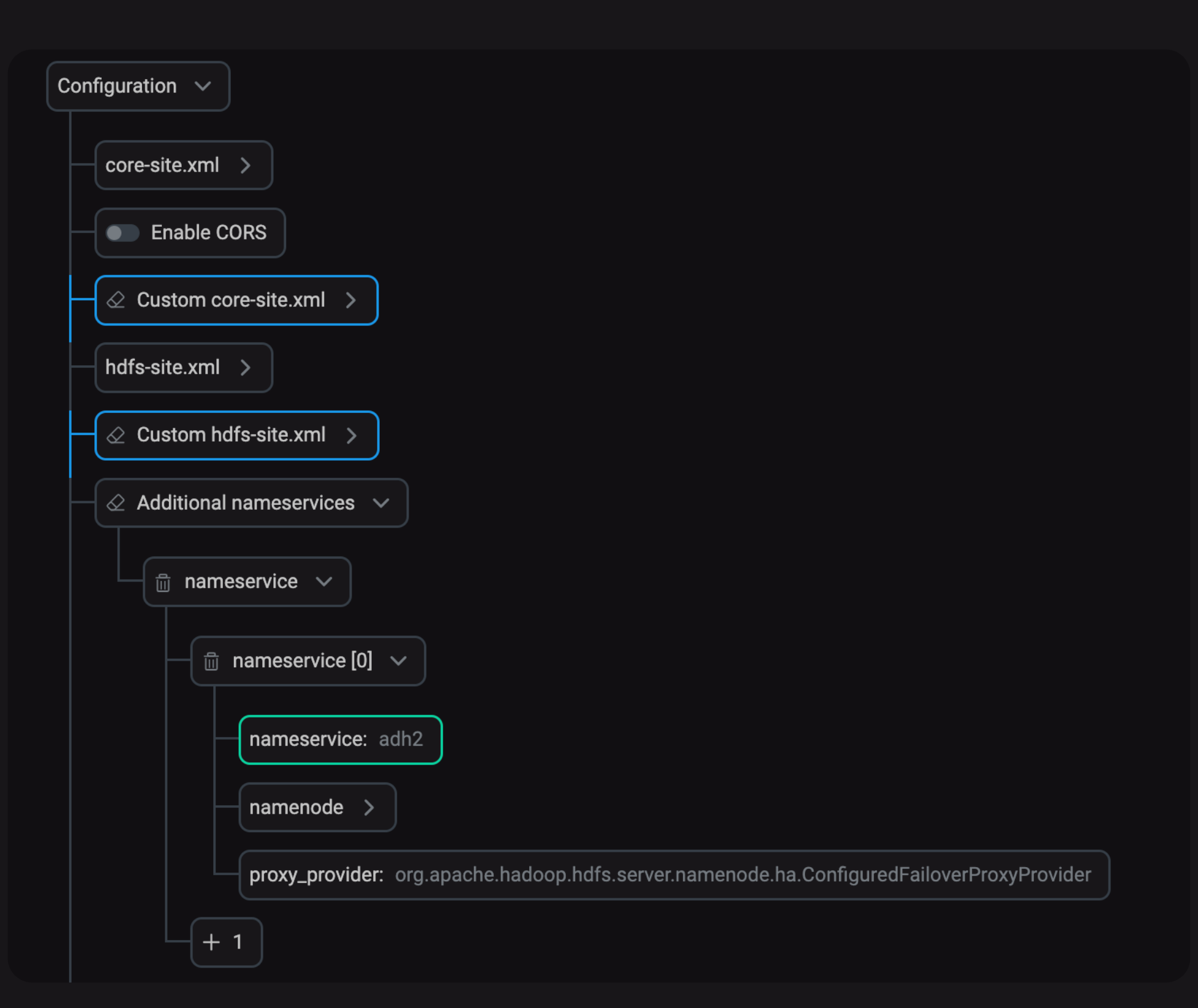
Для выполнения следующих шагов необходима информация:

- имена Kerberos realm кластеров;
- Internal nameservices кластеров (значение параметра `dfs.internal.nameservices` на странице конфигурации HDFS);
- FQDN или IP серверов Kerberos;
- FQDN и IP серверов NameNode;
- NameNodes ID (NameNode ID можно найти в графическом интерфейсе NameNode по адресу: `http://<NameNode IP or FQDN>:9870/`);
- реквизиты принципала аминистратора Kerberos.

Установка доверия

Чтобы настроить доверие между двумя серверами Kerberos через ADCM, выполните следующие действия для обоих кластеров:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и выберите HDFS.
3. Найдите параметр ***Additional nameservices*** в секции ***hdfs-site.xml*** и заполните его поля. Для каждого кластера укажите параметры другого кластера.



nameservice: adh2
proxy_provider: org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider

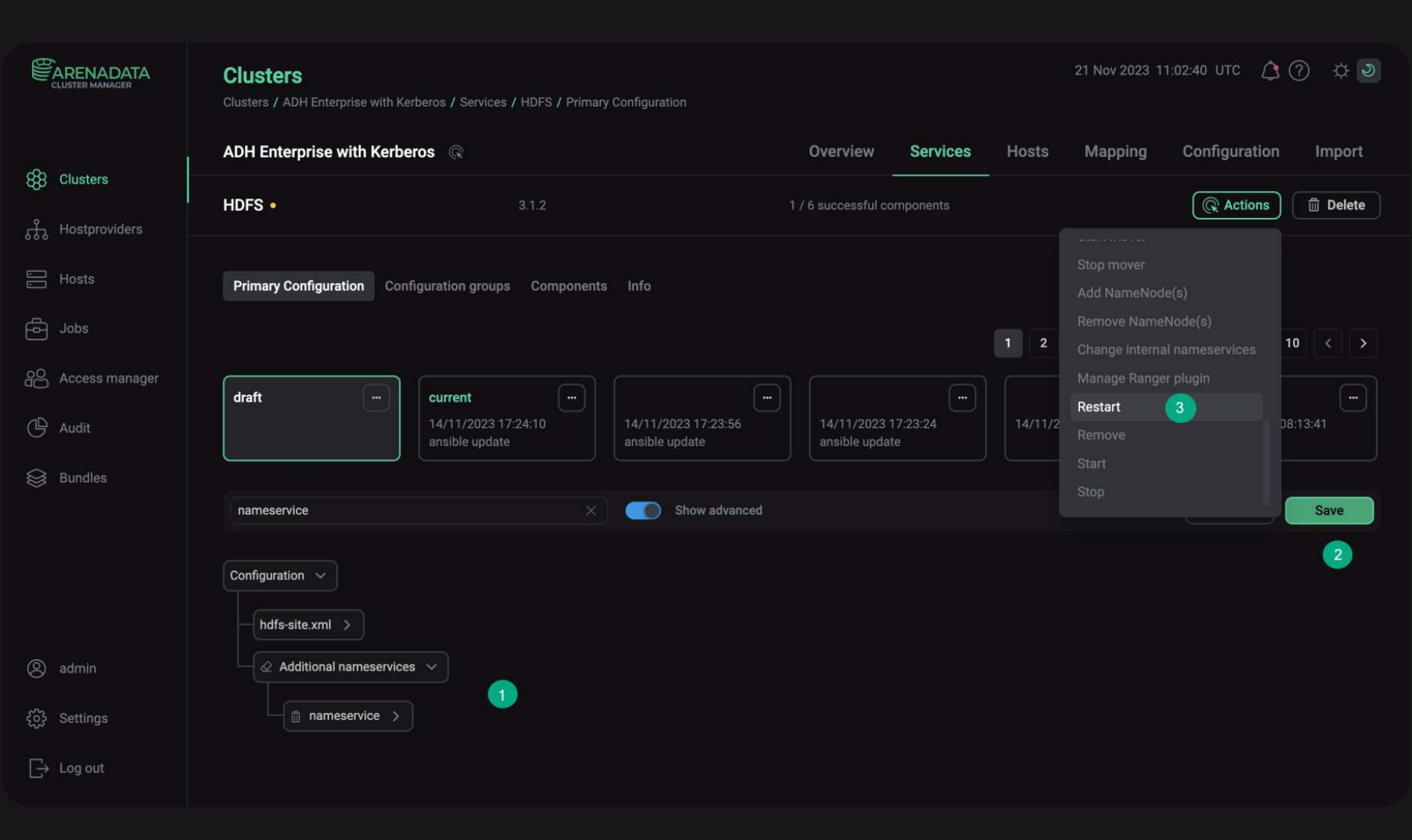
Параметр Additional nameservices в ADCM

4. В HDFS-параметр ***hadoop.security.auth_to_local*** добавьте следующие правила для настройки связи между пользователями разных кластеров:

```
RULE:[2:$1/$2@$0](hdfs-namenode/.*@<REALM2>)s/.*/hdfs/RULE:[2:$1/$2@$0](hdfs-datanode/.*@<REALM2>)s/.*/hdfs/RULE:[2:$1/$2@$0](hdfs-
journalnode/.*@<REALM2>)s/.*/hdfs/RULE:[2:$1/$2@$0](hdfs/.*@<REALM2>)s/.*/hdfs/RULE:[2:$1/$2@$0](yarn-resourcemanager/.
*@<REALM2>)s/.*/yarn/RULE:[2:$1/$2@$0](yarn-nodemanager/.*@<REALM2>)s/.*/yarn/RULE:[2:$1/$2@$0](yarn/.*@<REALM2>)s/.*/yarn/RULE:
[2:$1/$2@$0](hbase-master/.*@<REALM2>)s/.*/hbase/RULE:[2:$1/$2@$0](hbase-regionserver/.*@<REALM2>)s/.*/hbase/RULE:[2:$1/$2@$0]
(mapreduce-historyserver/.*@<REALM2>)s/.*/mapred/RULE:[2:$1/$2@$0](hdfs-namenode/.*@<REALM1>)s/.*/hdfs/RULE:[2:$1/$2@$0](hdfs-
datanode/.*@<REALM1>)s/.*/hdfs/RULE:[2:$1/$2@$0](hdfs-journalnode/.*@<REALM1>)s/.*/hdfs/RULE:[2:$1/$2@$0](hdfs/.*@<REALM1>)s/.
*/hdfs/RULE:[2:$1/$2@$0](yarn-resourcemanager/.*@<REALM1>)s/.*/yarn/RULE:[2:$1/$2@$0](yarn-nodemanager/.*@<REALM1>)s/.*/yarn/RULE:
[2:$1/$2@$0](yarn/.*@<REALM1>)s/.*/yarn/RULE:[2:$1/$2@$0](hbase-master/.*@<REALM1>)s/.*/hbase/RULE:[2:$1/$2@$0](hbase-
regionserver/.*@<REALM1>)s/.*/hbase/RULE:[2:$1/$2@$0](mapreduce-historyserver/.*@<REALM1>)s/.*/mapred/DEFAULT
```

   Здесь `<REALM1>` и `<REALM2>` — названия Kerberos realm кластеров. Более подробную информацию о настройке правил можно получить в статье Hadoop in Secure Mode.

5. На той же странице найдите параметр ***User Managed Hadoop.security.auth_to_local*** и установите для него значение `true`.
6. После настройки параметров HDFS для обоих кластеров перезагрузите сервисы HDFS с включенной опцией ***Apply configs from ADCM***.



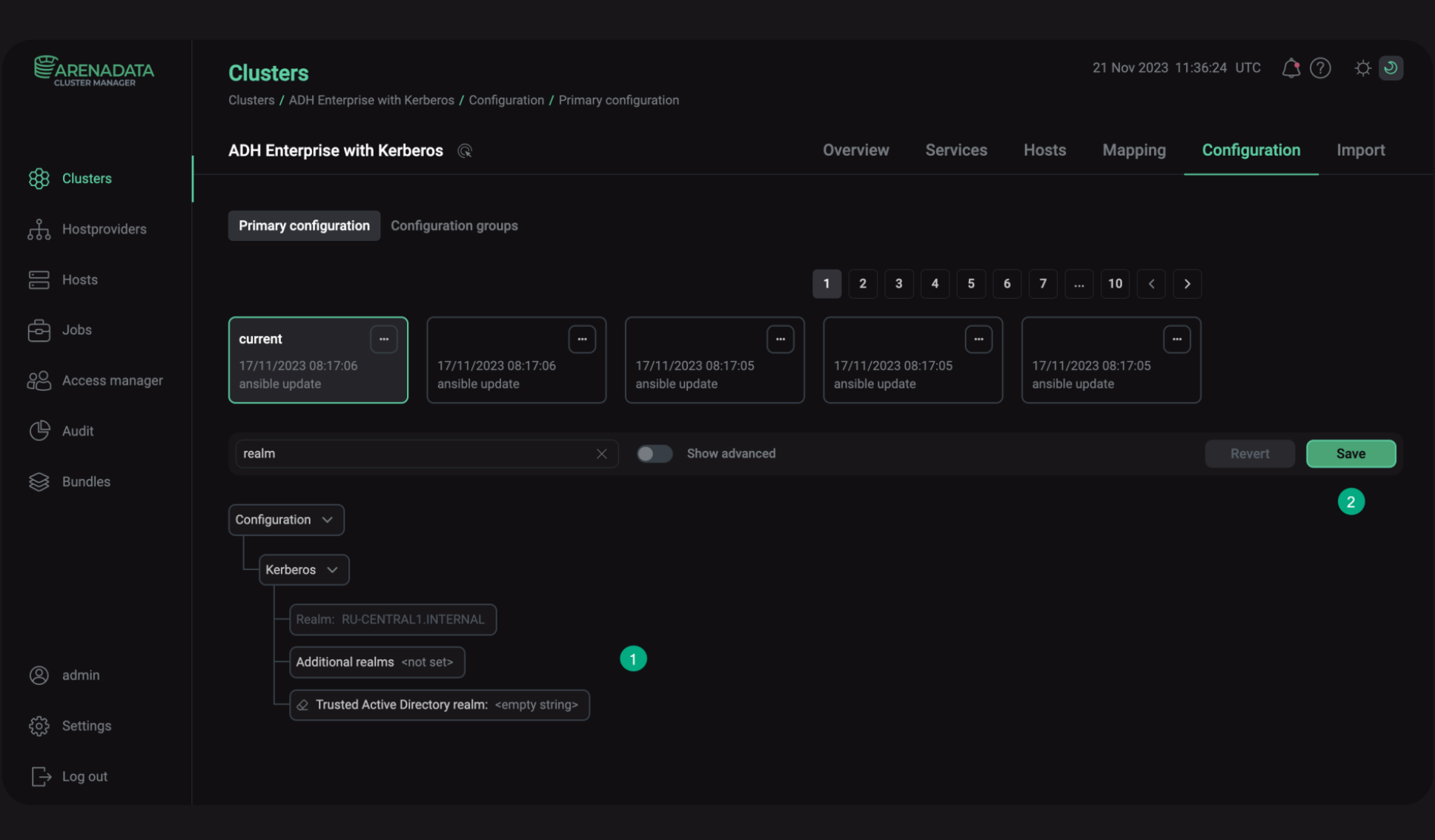
Настройка HDFS для доверия в Kerberos

7. На обоих серверах Kerberos создайте общего принципала, выполнив следующие команды:

```
$ kadmin.local addprinc -e "aes128-cts-hmac-sha1-96:normal des3-cbc-sha1:normal arcfour-hmac-md5:normal" krbtgt/<REALM1>@<REALM2>
$ kadmin.local addprinc -e "aes128-cts-hmac-sha1-96:normal des3-cbc-sha1:normal arcfour-hmac-md5:normal" krbtgt/<REALM2>@<REALM1>
```

   Здесь `<REALM1>` и `<REALM2>` — названия Kerberos realm кластеров.

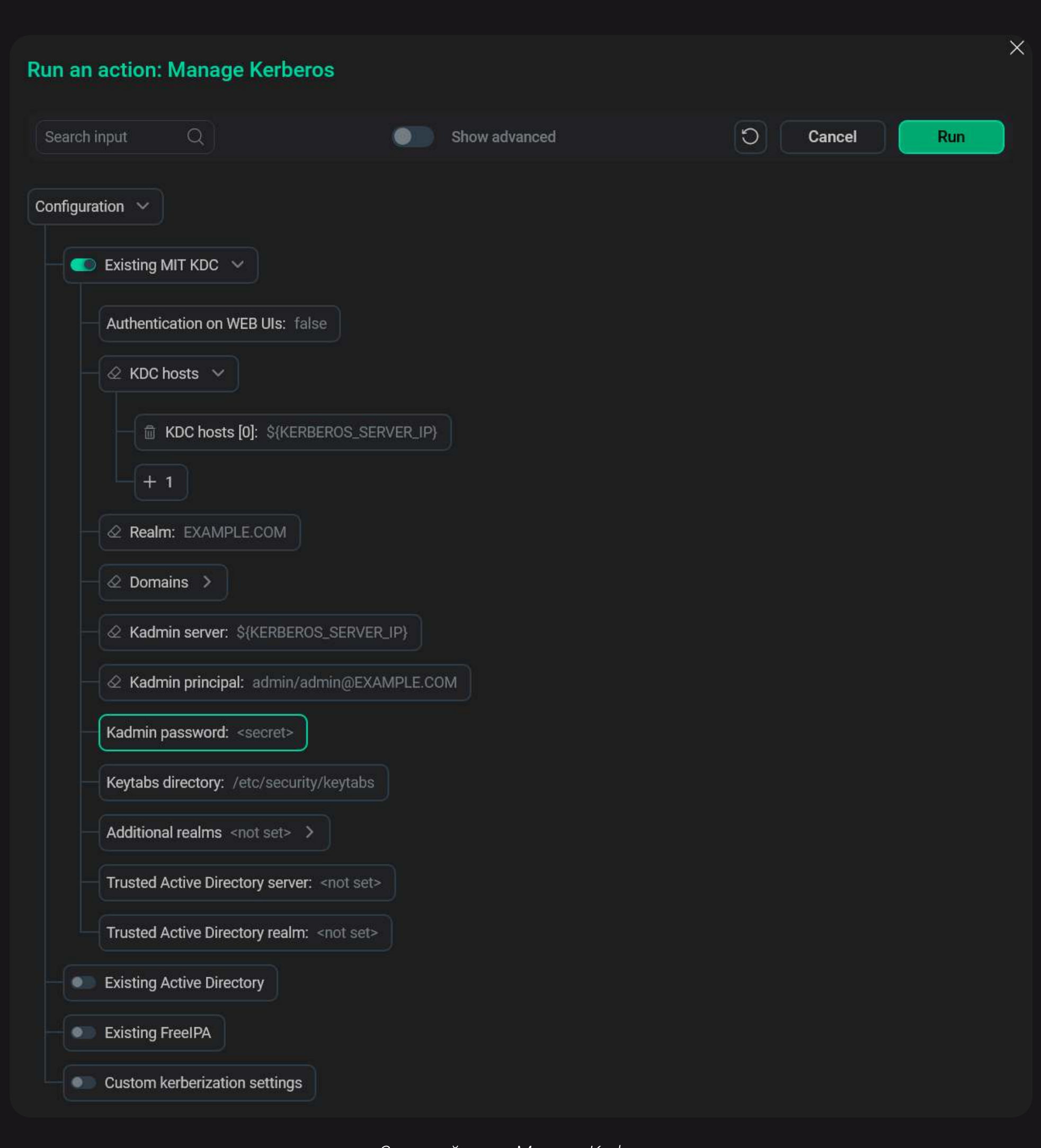
8. Настройте параметр ***Additional realms***. Для каждого кластера укажите параметры другого кластера.



Realm: RU-CENTRAL1.INTERNAL
Additional realms <not set>
Trusted Active Directory realm: <empty string>

Параметр Additional realms в ADCM

9. Включите Kerberos в обоих кластерах. Установите флажок ***Advanced***, затем нажмите ***Existing MIT KDC*** и заполните параметры:
   - KDC hosts;
   - Realm;
   - Kadmin server;
   - Kadmin principal;
   - Kadmin password.



Authentication on WEB UIs: false
KDC hosts [0]: ${KERBEROS_SERVER_IP}
Realm: EXAMPLE.COM
Kadmin server: ${KERBEROS_SERVER_IP}
Kadmin principal: admin/admin@EXAMPLE.COM
Kadmin password: <secret>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
Trusted Active Directory server: <not set>
Trusted Active Directory realm: <not set>

Окно действия Manage Kerberos

Использование distcp

1. Подключитесь к хосту кластера через SSH и войдите в систему как принципал HDFS:

```
$ kinit hdfs
```

2. Запустите команду `distcp`:

```
$ hadoop distcp hdfs://<nameservice1>:8020/<source directory> hdfs://<nameservice2>:8020/<target directory>
```

   Здесь:

   - `<nameservice1>` — internal nameservice кластера, из которого вы хотите скопировать данные;
   - `<nameservice2>` — internal nameservice кластера, в который необходимо скопировать данные;
   - `<исходный каталог>` — директория, из которой необходимо скопировать данные;
   - `<целевой каталог>` — директория, в которую необходимо скопировать данные.

Использование HttpFS

Елена Костюченко

Назад к содержанию

HDFS предоставляет встроенный сервер HttpFS, который можно использовать для передачи данных между кластерами с разными версиями ADH, для доступа к данным через firewall, использования HTTP-сервисов и многого другого.

В Arenadata Hyperwave сервер HttpFS предустановлен и доступен в виде пользовательского интерфейса HDFS. Для получения информации о пользовательских интерфейсах HDFS обратитесь к статье Обзор HDFS UI.

Вы можете ознакомиться с настройками HttpFS-сервера по умолчанию изучив следующие разделы на странице конфигурации HDFS: ***httpfs-site.xml***, ***httpfs-env.sh*** и ***HDFS Heap Memory settings***.

Чтобы найти эти параметры в ADCM, перейдите на страницу ***Clusters***, выберите нужный кластер, перейдите на вкладку ***Services*** и нажмите на HDFS.

Пример запроса для вызова команды `ls` в директории *user/tmp* под пользователем `admin`:

```
$ curl 'http://<httpfs-host>:14000/webhdfs/v1/user/tmp?op=LISTSTATUS&user.name=admin'
```

Здесь `<httpfs-host>` — IP-адрес хоста с установленным сервером HttpFS.

Политики хранения

Елена Костюченко

Содержание

Назад к содержанию

Обзор
Типы хранилищ
Политики хранения
Команды CLI

Обзор

Политики хранения — это набор правил, которые помогают повысить производительность приложений за счет оптимизации распределения данных в зависимости от потребности в вычислительной мощности и доступности. Например, если какие-то данные используются чаще других, вы можете переместить их на SSD-диск для повышения производительности, а неиспользуемые данные переместить в архив для экономии ресурсов.

Если для каталога указана политика хранения, HDFS будет хранить по крайней мере одну копию каждого файла в этом каталоге в указанном типе хранилища. Если места недостаточно, HDFS сохранит новые данные в **Fallback storages for creation**, а реплики будут храниться в **Fallback storages for replication**.

Типы хранилищ

В зависимости от выбранной политики доступны разные типы хранилищ:

- **ARCHIVE** (архив) — хранилище с небольшими вычислительными ресурсами, но большой емкостью;
- **DISK** — тип хранилища по умолчанию со средним ресурсом вычислений и емкостью;
- **SSD** — хранилище, размещенное на SSD-дисках;
- **RAM_DISK** — оперативная память узла данных;
- **PROVIDED** — хранилище вне системы HDFS. Дополнительную информацию о хранилище этого типа можно получить в статье HDFS Provided Storage.

Прежде чем настраивать политику хранения, убедитесь, что существует каталог с необходимым типом хранилища. Более подробную информацию о том, как добавить определенный тип хранилища, можно получить в статье Добавление каталогов данных HDFS.

Политики хранения

В HDFS доступны следующие политики хранения:

- **Hot**. Эта политика подходит для данных, регулярно используемых для обработки. Все блоки данных сохраняются на DISK, когда на нем достаточно места, а если пространство на DISK ограничено, блоки будут храниться в ARCHIVE.
- **Cold**. Политика хранения холодных данных применяется к редко используемым данным. Все блоки хранятся в ARCHIVE.
- **Warm**. Если данные используются только время от времени, вы можете установить политику Warm. В этом случае в DISK хранится только одна реплика, а остальные реплики хранятся в ARCHIVE.
- **All_SSD**. Эта политика хранения применяется для хранения всех данных на SSD. Если на SSD нет места, данные сохраняются на DISK.
- **One_SSD**. Согласно этой политике хранения, только одна из реплик хранится на SSD. Остальные реплики хранятся в DISK.
- **Lazy_Persist**. При использовании политики хранения Lazy_Persist в оперативной памяти всегда хранится одна реплика. Эта политика полезна только для блоков с одной репликой. Если реплик несколько, они все будут записаны на DISK.
- **Provided**. Эта политика хранения применяется для хранения одной реплики в хранилище типа PROVIDED, а остальные реплики хранятся на DISK.

Название политики	ID	Размещение блоков	Fallback storages for creation	Fallback storages for replication
Hot	7	Все реплики хранятся в DISK	—	ARCHIVE
Cold	2	Все реплики хранятся в ARCHIVE	—	—
Warm	5	Одна реплика хранится в DISK, остальные в ARCHIVE	ARCHIVE, DISK	ARCHIVE, DISK
All_SSD	12	Все реплики хранятся на SSD	DISK	DISK
One_SSD	10	Одна реплика хранится на SSD, а остальные в DISK	SSD, DISK	SSD, DISK
Lazy_Persist	15	Одна реплика хранится в RAM_DISK. Если реплик несколько, все реплики хранятся в DISK	DISK	DISK
Provided	1	Одна реплика хранится в PROVIDED, а остальные — в DISK	PROVIDED, DISK	PROVIDED, DISK

ВАЖНО

Для файлов с настройкой striped erasure coding единственными подходящими политиками хранения являются All_SSD, Hot или Cold. Если вы установите другую политику, она не будет работать.

Команды CLI

Для управления политиками хранения используйте команду storagepolicies.

Пример использования команды для проверки назначенной политики хранения в директории */tmp*:

```
$ hdfs storagepolicies -getStoragePolicy -path /tmp
```

Когда установлена новая политика, она не применяется автоматически. Чтобы разместить блоки в соответствии с политиками хранения, используйте действие Mover в ADCM.

Вывод команды:

```
The storage policy of /tmp:
BlockStoragePolicy{HOT:7, storageTypes=[DISK], creationFallbacks=[], replicationFallbacks=[ARCHIVE]}
```

Квоты

Елена Костюченко

Содержание

Назад к содержанию

Квоты имен

Квоты пространств

Квоты типа хранилища

В HDFS можно установить ограничения на размер директории и количество файлов (имен) с помощью квот.

Существуют три типа квот:

- Квоты имен (name quotas) — ограничение на количество имен в каталоге.
- Квоты пространств (space quotas) — ограничение на дисковое пространство, используемое каталогом.
- Квоты типа хранилища (storage type quotas) — ограничение на дисковое пространство определенного типа.

Все типы квот работают независимо. Это значит, что вы можете настроить все три типа квот одновременно.

Все квоты записываются в файл *fsimage*. Когда сервис NameNode запускается, он выводит информацию обо всех нарушениях квот.

Информация об установке или удалении квоты заносится в EditLogs.

Квоты имен

Квота имен — это ограничение на количество имен файлов и директорий в каталоге, включая сам каталог. В квоте учитывается создание новых файлов или директорий и их переименование.

Когда квота будет заполнена, любые операции создания или переименования файлов будут завершаться с ошибкой.

Попытка установить квоту будет успешной, даже если каталог уже нарушает эту квоту.

Чтобы установить квоту имен, используйте команду:

```
$ hdfs dfsadmin -setQuota <N> <directory>...<directory>
```

Здесь:

- `<N>` — положительное целое число, представляющее количество имен, разрешенных в каталоге;
- `<directory>` — путь к директории, для которой нужно установить квоту.

Чтобы удалить квоту имен для директории, используйте команду:

```
$ hdfs dfsadmin -clrQuota <directory>...<directory>
```

Здесь `<directory>` — путь к директории, для которой нужно удалить квоту.

Квоты пространств

Квота пространства — это ограничение размера директории. Квота измеряется блоками, включая их реплики. Это означает, что изменение коэффициента репликации файлов повлияет на заполнение квоты. Например, 1 ГБ данных с коэффициентом репликации 3 занимает 3 ГБ квоты.

В квоте не учитываются операции, при которых не происходит создание блоков. Например:

- создание каталога;
- создание файла (если квота заполнена, файл будет создан, но он будет пустым);
- размещение метаданных.

Чтобы установить квоту пространств для директории, используйте команду:

```
$ hdfs dfsadmin -setSpaceQuota <N> <directory>...<directory>
```

Здесь:

- `<N>` — положительное целое число, представляющее предельный размер директории в байтах. Для удобства размер также может быть указан с помощью двоичного префикса, например, 50g для 50 ГБ, 2t для 2 ТБ и т. д.
- `<directory>` — путь к директории, для которой необходимо установить квоту пространства.

Чтобы удалить квоту пространства, используйте команду:

```
$ hdfs dfsadmin -clrSpaceQuota <directory>...<directory>
```

Здесь `<directory>` — путь к директории, для которой нужно удалить квоту.

Квоты типа хранилища

Квоты типа хранилища ограничивают объем определенного типа дискового пространства, который может использоваться директорией. При подсчете используемого пространства этот тип квот работает аналогично квотам пространства.

Прежде чем устанавливать квоту типа хранилища, убедитесь, что для директории настроена политика хранения. Для получения информации о политиках хранения обратитесь к статье Политики хранения.

Чтобы установить квоту типа хранилища для директории, используйте команду:

```
$ hdfs dfsadmin -setSpaceQuota <N> -storageType <storagetype> <directory>...<directory>
```

Здесь:

- `<N>` — положительное целое число, представляющее предельный размер определенного типа хранилища, которое занимает директория, в байтах. Для удобства размер также может быть указан с помощью двоичного префикса, например, 50g для 50 ГБ, 2t для 2 ТБ и т. д.
- `<storagetype>` — тип хранилища, для которого необходимо установить ограничение. Возможные значения: `RAM_DISK`, `DISK`, `SSD`, `ARCHIVE`.
- `<directory>` — путь к директории, для которой необходимо установить квоту типа хранилища.

Чтобы удалить квоту типа хранилища для директории, используйте команду:

```
$ hdfs dfsadmin -clrSpaceQuota -storageType <storagetype> <directory>...<directory>
```

- `<storagetype>` — тип хранилища, для которого необходимо снять ограничения. Возможные значения: `RAM_DISK`, `DISK`, `SSD`, `ARCHIVE`.
- `<directory>` — путь к директории, для которой необходимо удалить квоту типа хранилища.

Управление кешем

Елена Костюченко

Содержание

Назад к содержанию

Обзор

Директивы и пулы кеша

Конфигурация

Поток данных

Команды CLI

Обзор

В HDFS можно настроить **централизованное управление кешем**, которое позволит сохранять выбранную директорию или файл в кеше. Это обеспечит быстрый доступ к данным и ускорит выполнение задач, связанных с кешированными данными.

Когда эта опция отключена, у пользователей и приложений нет контроля над тем, какие данные кешируются. Рабочие нагрузки с низким приоритетом могут вытеснить из кеша задания с высоким приоритетом, а приложения, принимающие решения о размещении задач (например, MapReduce или Impala), не смогут учитывать кешированные данные при создании задач.

Управление кешем разрешает использовать кеш кластера планировщикам задач и позволяет клиентам закреплять свои данные в памяти кластера. Кроме того, приложения потенциально могут выполнять эффективное zero-copy чтение и пропускать проверку контрольных сумм, поскольку целостность кешированных данных уже проверена.

Настройка централизованного кеша повышает производительность нескольких сервисов Hadoop, включая HBase, Impala и Hive.

Директивы и пулы кеша

Существует две основные концепции, используемые для управления кешем: **директивы кеша** и **пулы кеша**.

Директива кеша — это путь, который пользователь или приложение запрашивает для сохранения в кеше. Это может быть как файл, так и директория.

Кеширование нерекурсивно. Это означает, что если указан файл, все его блоки будут сохранены в кеше. Но если указана директория, кешироваться будут только файлы первого уровня этой директории.

Если директива содержит символическую ссылку, содержимое, на которое она указывает, не будет кешироваться.

При создании директивы кеша необходимо указать два параметра:

- **Коэффициент репликации кеша** — количество реплик для хранения в кеше. Если несколько директив кеша относятся к одному и тому же файлу, применяется наибольший указанный коэффициент репликации.
- **Время жизни (time-to-live, TTL)** — срок действия директивы. После истечения срока действия она больше не учитывается при принятии решений о кешировании.

После получения директивы NameNode будет периодически проверять, были ли изменены файлы в каталоге директивы. Если файлы изменились, NameNode даст указание кешировать новые блоки.

Пул кеша — это инструмент администрирования и управления ресурсами кеша. Каждая директива принадлежит определенному пулу кеша.

Для каждого пула кеша можно настроить следующее:

- Разрешения (как в UNIX-системах) — определяют какие пользователи и группы имеют доступ к пулу. Разрешения на запись позволяют пользователям добавлять и удалять директивы кеша, разрешения на чтение позволяют видеть список директив в пуле и метаданные.
- Максимальный размер — суммарное ограничение размера в байтах для всех директив в пуле. Нежелательно, чтобы сумма ограничений всех пулов превышала объем совокупной памяти, зарезервированной для кеширования HDFS в кластере.
- Максимальный TTL — максимальное время жизни директив кеша в пуле.

Конфигурация

Чтобы включить управление кешем, необходимо установить значение, отличное от нуля, для параметра `dfs.datanode.max.locked.memory` в *hdfs-site.xml*. Этот параметр описывает, какой объем памяти DataNode может быть зарезервирован для кеширования.

При настройке параметра `dfs.datanode.max.locked.memory` учитывайте:

- В Unix-подобных системах размер locked-in-memory пользователя DataNode (`ulimit -l`) необходимо увеличить, чтобы он соответствовал этому параметру (подробнее о возможных ошибках ниже).
- DataNode, JVM heap приложений, а также страничный кеш операционной системы также требуют места.
- Lazy Persist Writes также использует значение параметра `dfs.datanode.max.locked.memory`.

Возможные ошибки: конфигурация ulimit в Linux

Одна из возможных ошибок при настройке централизованного кеша:

```
Cannot start datanode because the configured max locked memory size… is more than the datanode’s available RLIMIT_MEMLOCK ulimit.
```

Это означает, что ограничение использования памяти, настроенное на уровне ОС, ниже лимита, установленного для DataNode. Чтобы это исправить, измените значение `ulimit -l` для DataNode.

Чтобы убедиться, что ulimit правильно настроен, выполните команду:

```
$ ulimit -l
```

Если команда выводит `unlimited` или значение большее, чем в параметре `dfs.datanode.max.locked.memory`, конфигурация настроена верно.

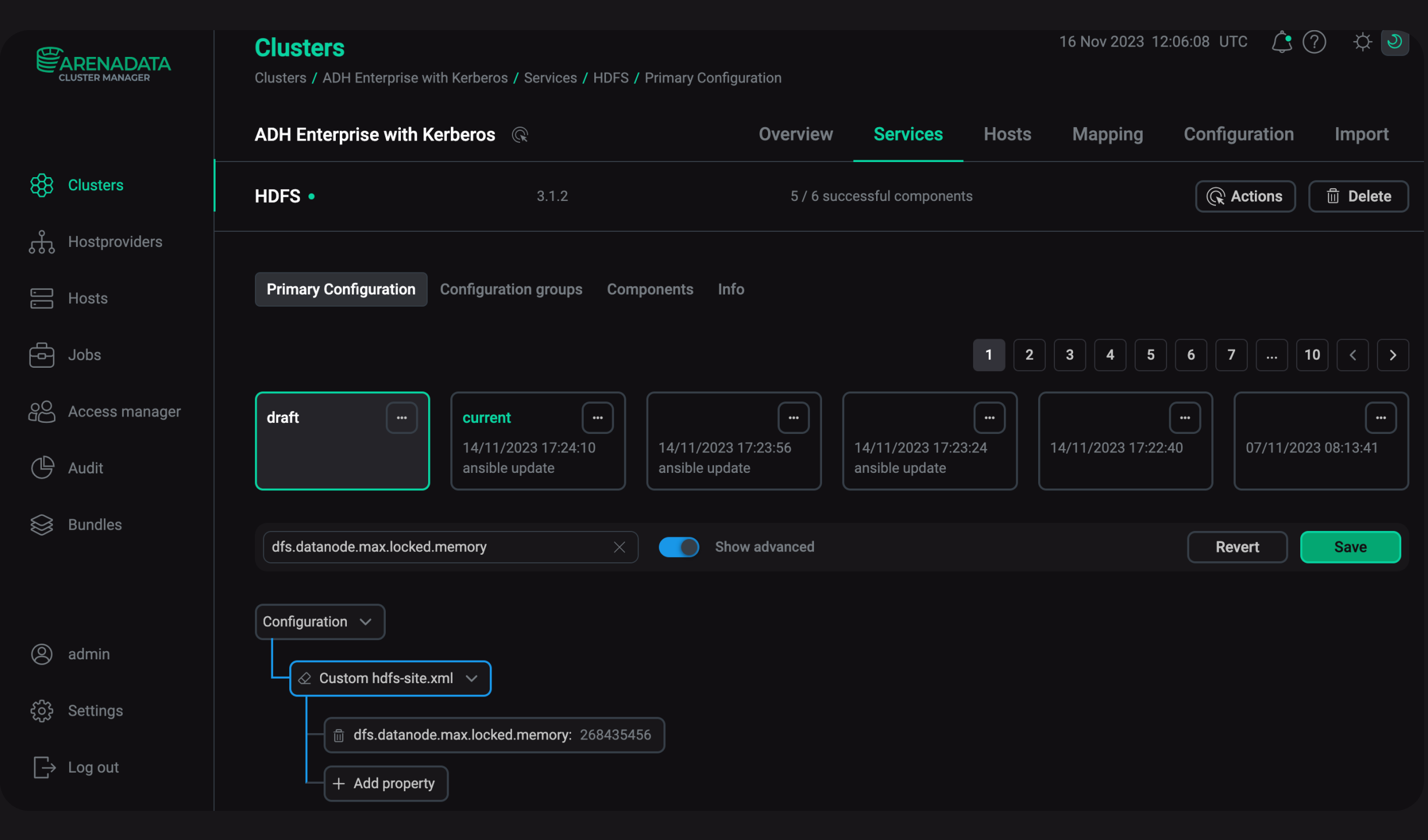
Обратите внимание, что эта команда выводит значение в килобайтах, а значение `dfs.datanode.max.locked.memory` в байтах.

По умолчанию централизованный кеш недоступен в ADH, но его можно настроить с помощью пользовательского параметра конфигурации в ADCM.

Прежде чем включать управление кешем, убедитесь, что нативная библиотека *libhadoop.so* доступна для DataNodes. Более подробную информацию об установке библиотеки *libhadoop.so* можно получить в статье Native Libraries Guide .

Чтобы включить централизованный кеш через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services***.
3. Выберите сервис HDFS.
4. Включите опцию ***Show advanced*** и найдите ***Custom hdfs-site.xml***.
5. Откройте раскрывающийся список и выберите ***Add property***.
6. Введите имя параметра `dfs.datanode.max.locked.memory` и желаемое значение в байтах.
7. Нажмите ***Apply***.



dfs.datanode.max.locked.memory: 268435456

Параметр Custom hdfs-site.xml в ADCM

Таким же образом можно настроить опциональные параметры кеша из таблицы ниже.

Дополнительные настройки кеширования

Параметр	Описание	Значение по умолчанию
dfs.namenode.path.based.cache.refresh.interval.ms	Параметр описывает, как часто NameNode проверяет есть ли на DataNodes новые блоки, которые нужно кешировать. Указывается в миллисекундах	30000
dfs.datanode.fsdatasetcache.max.threads.per.volume	Максимальное количество потоков на DataNode, которое можно использовать для кеширования новых данных. Эти потоки потребляют вычислительные ресурсы, что может повлиять на работу DataNode	4
dfs.cachereport.intervalMsec	Параметр описывает, как часто DataNode отправляет отчеты о кеше в NameNode. Указывается в миллисекундах	10000
dfs.namenode.path.based.cache.block.map.allocation.percent	Процент Java heap, который будет зарезервирован для карты кешированных блоков. Доступ к картам меньшего размера может осуществляться медленнее, если количество кешированных блоков большое. Большие карты потребляют больше памяти. Указывается в виде десятичной дроби	0.25

Поток данных

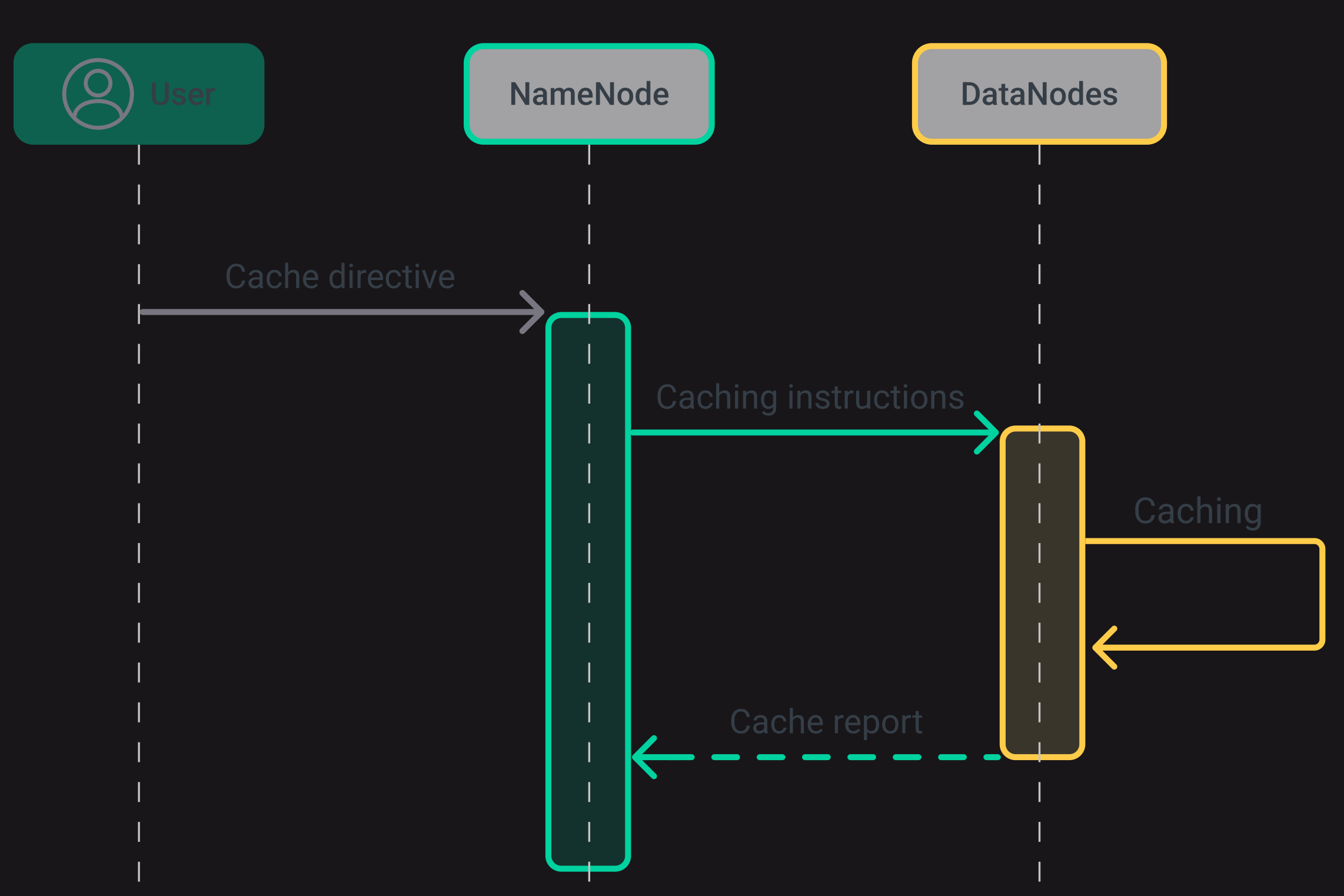


Диаграмма последовательности запроса на кеширование

Централизованное кеширование в HDFS работает аналогично хранению обычных блоков данных: кеш управляется NameNode и хранится на DataNodes.

Когда клиент создает директиву кеширования, активный NameNode проверяет DataNode, на которых находятся нужные блоки, и дает им указание кешировать эти блоки.

В то же время NameNode делает запись о новой директиве в журнале изменений. Это позволяет NameNode повторно кешировать данные при перезапуске.

DataNodes сохраняют запрошенные блоки в памяти и отправляют отчеты о кеше всем существующим NameNode. После этого NameNode обновляет карту кешированных блоков и отправляет DataNodes информацию о том, какие реплики следует кешировать, а какие убрать из кеша.

Команды CLI

Администратор может управлять директивами и пулами кеша с помощью команды cacheadmin.

Чтобы создать новую директиву в новом пуле кеша:

1. Создайте новый пул кеша:

```
$ hdfs cacheadmin -addPool <name> -owner <owner> -group <group> -mode <mode> -limit <limit> -maxTtl <maxTtl>
```

   Здесь:

   - `<name>` — имя нового пула кеша.
   - `<owner>` — имя нового владельца пула. Владельцем по умолчанию является текущий пользователь.
   - `<group>` — группа пользователей пула. Группа по умолчанию — это основная группа текущего пользователя.
   - `<mode>` — UNIX-разрешения для пула. Указываются в восьмеричном формате. По умолчанию установлено значение 0755.
   - `<limit>` — максимальный размер пула в байтах. По умолчанию пул не имеет ограничений.
   - `<maxTtl>` — максимально допустимое время жизни директив в пуле. Может быть указано в секундах, минутах, часах или днях, например: 120s, 30m, 4h, 2d. Если установлено значение `never`, ограничений на устанавливаемое время жизни нет. По умолчанию максимум не установлен.

2. Добавьте новую директиву:

```
$ hdfs cacheadmin -addDirective -path <path> -pool <pool-name> -force -replication <replication> -ttl <time-to-live>
```

   Здесь:

   - `<path>` — путь к каталогу или файлу, который необходимо кешировать.
   - `<pool-name>` — пул, в который нужно добавить директиву. Для добавления новых директив необходимо разрешение на запись в пул кеша.
   - `-force` — если указана эта опция, директива будет добавлена без проверки на превышение ограничений ресурсов пула кеша.
   - `<replication>` — используемый коэффициент репликации кеша. Значение по умолчанию — 1.
   - `<time-to-live>` — срок действия директивы. Значение может быть указано в секундах, минутах, часах и днях, например: 120s, 30m, 4h, 2d. Если установлено значение `never`, директива будет действовать бесконечно. Значение по умолчанию — `never`.

Фактор репликации

Елена Костюченко

Содержание

Назад к содержанию

- Изменение фактора репликации
  - Увеличение фактора репликации
  - Понижение фактора репликации
- Изменение размера блока

Фактор репликации (replication factor) HDFS — это параметр, в котором задается количество копий файла, которые будут храниться в системе.

При создании файла в HDFS он разбивается на блоки — фрагменты данных фиксированного размера. Затем каждый блок этого файла копируется столько раз, сколько задано в значении фактора репликации. Фактор репликации HDFS по умолчанию равен трем.

Более подробно о том, как работает репликация в HDFS, читайте в статье Архитектура HDFS.

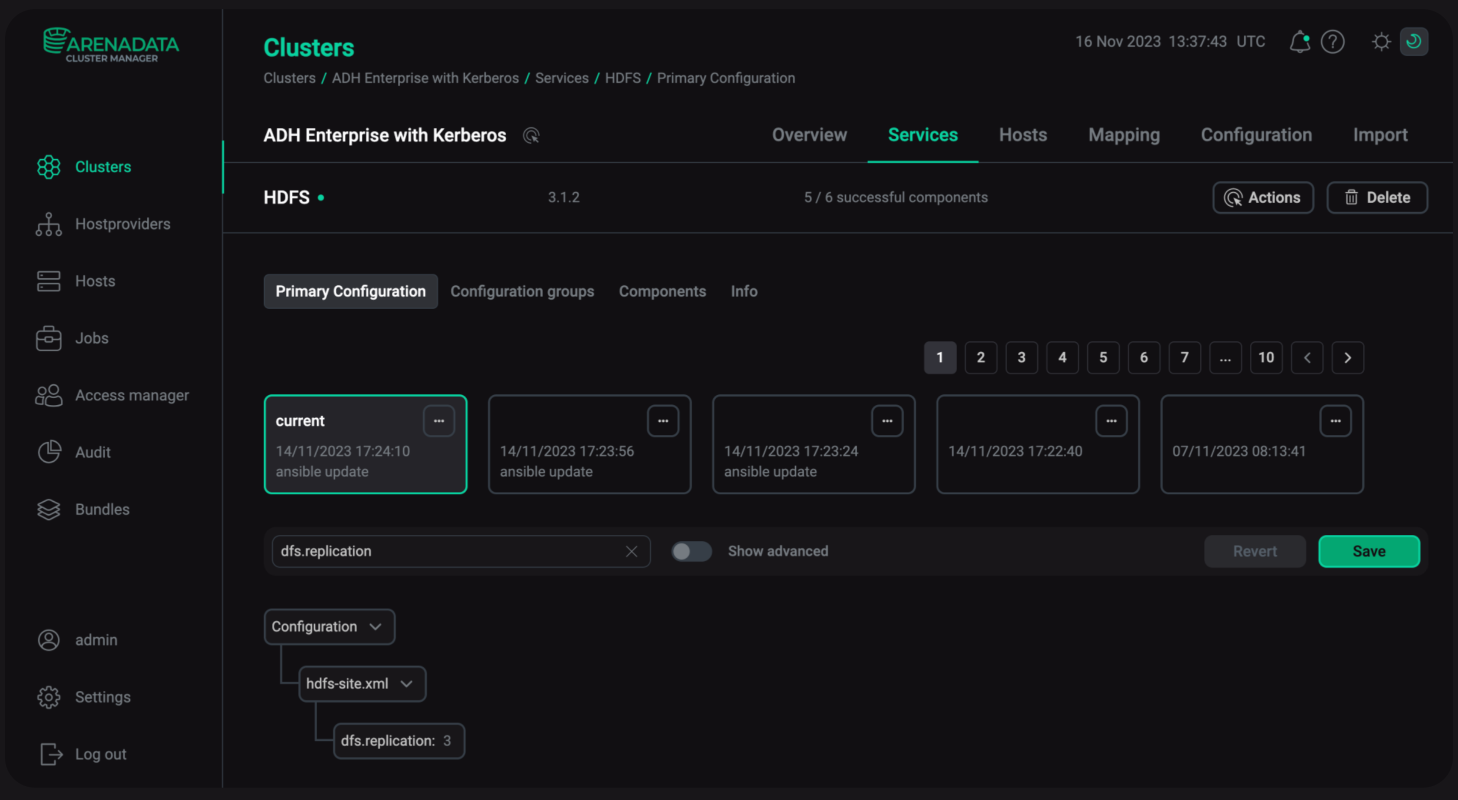
Изменение фактора репликации

Фактор репликации можно изменить как для всей системы, так и для отдельного файла или директории.

Чтобы изменить фактор репликации по умолчанию, измените значение параметра `dfs.replication` в файле конфигурации *hdfs-site.xml*.

Чтобы сделать это с помощью ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на HDFS.
3. Найдите поле параметра ***dfs.replication***, введите новое значение фактора репликации по умолчанию и нажмите ***Apply***. Подтвердите изменения в конфигурации HDFS, нажав ***Save***.
4. В меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters / ADH Enterprise with Kerberos / Services / HDFS / Primary Configuration
16 Nov 2023 13:37:43 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
ADH Enterprise with Kerberos
Overview
Services
Hosts
Mapping
Configuration
Import
HDFS
3.1.2
5 / 6 successful components
Actions
Delete
Primary Configuration
Configuration groups
Components
Info
1 2 3 4 5 6 7 ... 10
current
14/11/2023 17:24:10
ansible update
14/11/2023 17:23:56
ansible update
14/11/2023 17:23:24
ansible update
14/11/2023 17:22:40
07/11/2023 08:13:41
Show advanced
Revert
Save
Configuration
hdfs-site.xml

dfs.replication

dfs.replication: 3

Параметр dfs.replication в ADCM

Чтобы установить фактор репликации для файла или директории, используйте команду setrep:

```
$ hadoop fs -setrep <numReplicas> <path>
```

Здесь:

- `<numReplicas>` — фактор репликации файла или директории.
- `<path>` — путь к файлу или директории. Если указан путь к директории, команда изменяет фактор репликации всех файлов в директории, включая дочерние каталоги.

> **ПРИМЕЧАНИЕ**
>
> Если фактор репликации для директории был изменен, он будет применяться только к новым файлам, добавленным в каталог.

Увеличение фактора репликации

Когда фактор репликации по умолчанию увеличивается, NameNode помечает все блоки как недостаточно реплицированные и инициирует репликацию. Убедитесь, что фактор репликации не превышает количество DataNodes в кластере.

Понижение фактора репликации

Когда фактор репликации снижается, NameNode сообщает DataNodes какие блоки следует удалить. DataNodes может потребоваться некоторое время, чтобы удалить блоки и сообщить об освободившемся дисковом пространстве.

Вы можете установить фактор репликации равным 1. Это снизит отказоустойчивость HDFS, но это может быть необходимо, например, для кластеров с одним хостом. Если вы хотите установить фактор репликации 1 для экономии ресурсов, рассмотрите вместо этого возможность использования политик хранения или erasure coding.

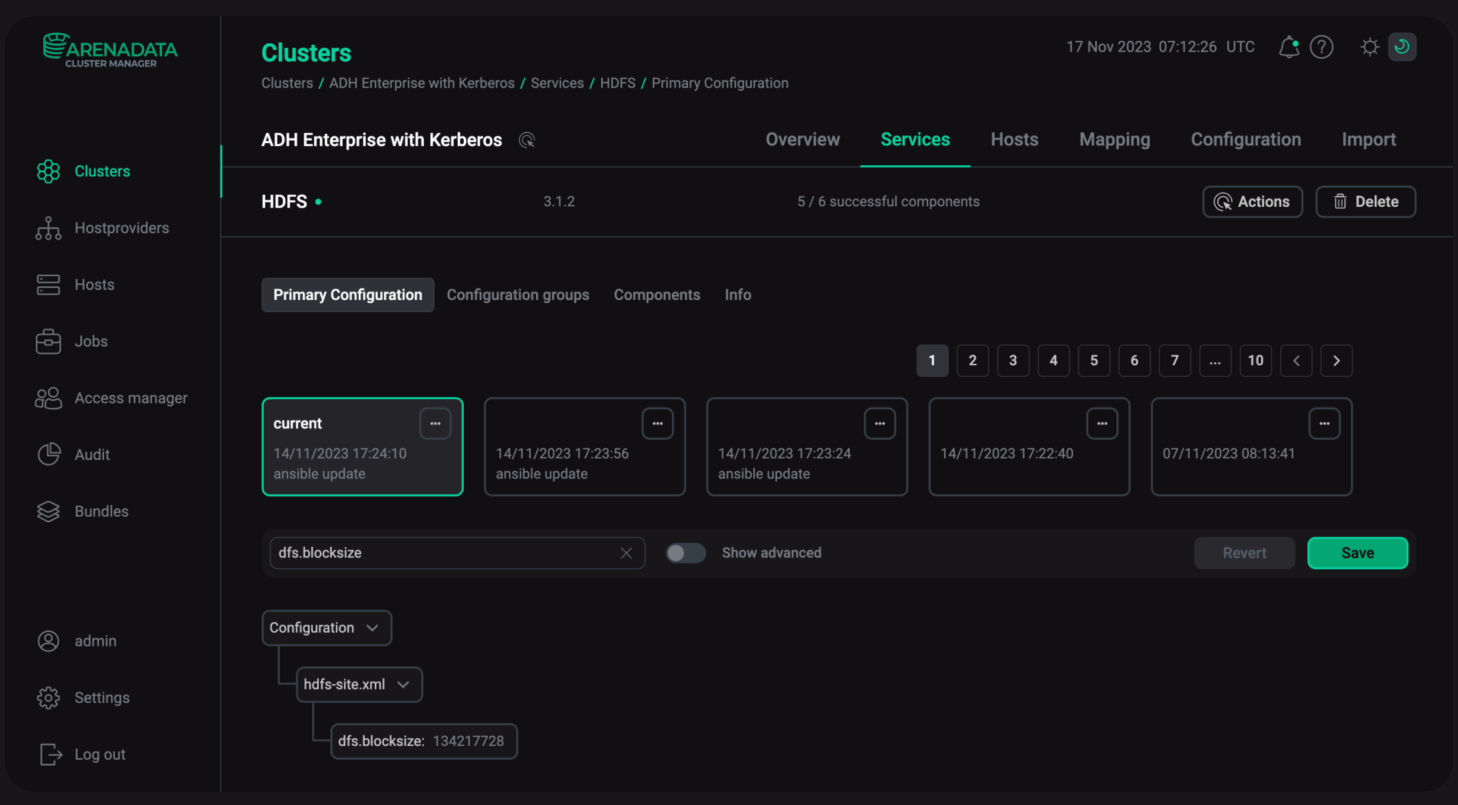
Изменение размера блока

Размер блока по умолчанию — 128 МБ. Вы можете изменить размер блока для HDFS в параметре `dfs.blocksize` файла конфигурации *hdfs-site.xml*.

Чтобы сделать это с помощью ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на HDFS.
3. Найдите поле параметра ***dfs.blocksize***, введите новый размер блока в байтах и нажмите ***Apply***. Вы также можете использовать сокращения, например, `k` для килобайт, `m` для мегабайт и `g` для гигабайт.
4. Подтвердите изменения в конфигурации HDFS, нажав ***Save***.
5. В меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Новый размер блока применится только к новым файлам, добавленным в HDFS.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters / ADH Enterprise with Kerberos / Services / HDFS / Primary Configuration
17 Nov 2023 07:12:26 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
ADH Enterprise with Kerberos
Overview
Services
Hosts
Mapping
Configuration
Import
HDFS
3.1.2
5 / 6 successful components
Actions
Delete
Primary Configuration
Configuration groups
Components
Info
1 2 3 4 5 6 7 ... 10
current
14/11/2023 17:24:10
ansible update
14/11/2023 17:23:56
ansible update
14/11/2023 17:23:24
ansible update
14/11/2023 17:22:40
07/11/2023 08:13:41
Show advanced
Revert
Save
Configuration
hdfs-site.xml

dfs.blocksize

dfs.blocksize: 134217728

Параметр dfs.blocksize в ADCM

Rack awareness

Елена Костюченко

Назад к содержанию

Rack awareness в Hadoop — это функция HDFS, которая обеспечивает дополнительную отказоустойчивость за счет учета топологии сети при размещении блоков данных. Если этот параметр включен, HDFS разместит одну реплику блока в DataNode на другой стойке. Это гарантирует доступность данных, например, в случае сбоя сетевого коммутатора или других проблем.

Чтобы настроить rack awareness через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.

2. Перейдите на вкладку ***Services*** и нажмите HDFS.

3. Включите опцию ***Show advanced*** и найдите поля ***Topology script*** и ***Topology data***.

4. В ***Topology script*** добавьте скрипт, описывающий топологию вашей сети.

Пример Topology script

```
#!/bin/bash

# Adjust/Add the property "net.topology.script.file.name"
# to core-site.xml with the "absolute" path the this
# file.  ENSURE the file is "executable".

# Supply appropriate rack prefix
RACK_PREFIX=default

# To test, supply a hostname as script input:
if [ $# -gt 0 ]; then

CTL_FILE=${CTL_FILE:-"topology.data"}

HADOOP_CONF=${HADOOP_CONF:-"/etc/hadoop/conf"}

if [ ! -f ${HADOOP_CONF}/${CTL_FILE} ]; then
  echo -n "/$RACK_PREFIX/rack "
  exit 0
fi

while [ $# -gt 0 ] ; do
  nodeArg=$1
  exec< ${HADOOP_CONF}/${CTL_FILE}
  result=""
  while read line ; do
    ar=( $line )
    if [ "${ar[0]}" = "$nodeArg" ] ; then
      result="${ar[1]}"
    fi
  done
  shift
  if [ -z "$result" ] ; then
    echo -n "/$RACK_PREFIX/rack "
  else
    echo -n "/$RACK_PREFIX/rack_$result "
  fi
done

else
  echo -n "/$RACK_PREFIX/rack "
fi
```

Больше примеров скриптов топологии доступно в документации Rack Awareness.

5. В поле ***Topology data*** укажите IP-адреса стоек и соответствующие им идентификаторы следующим образом:

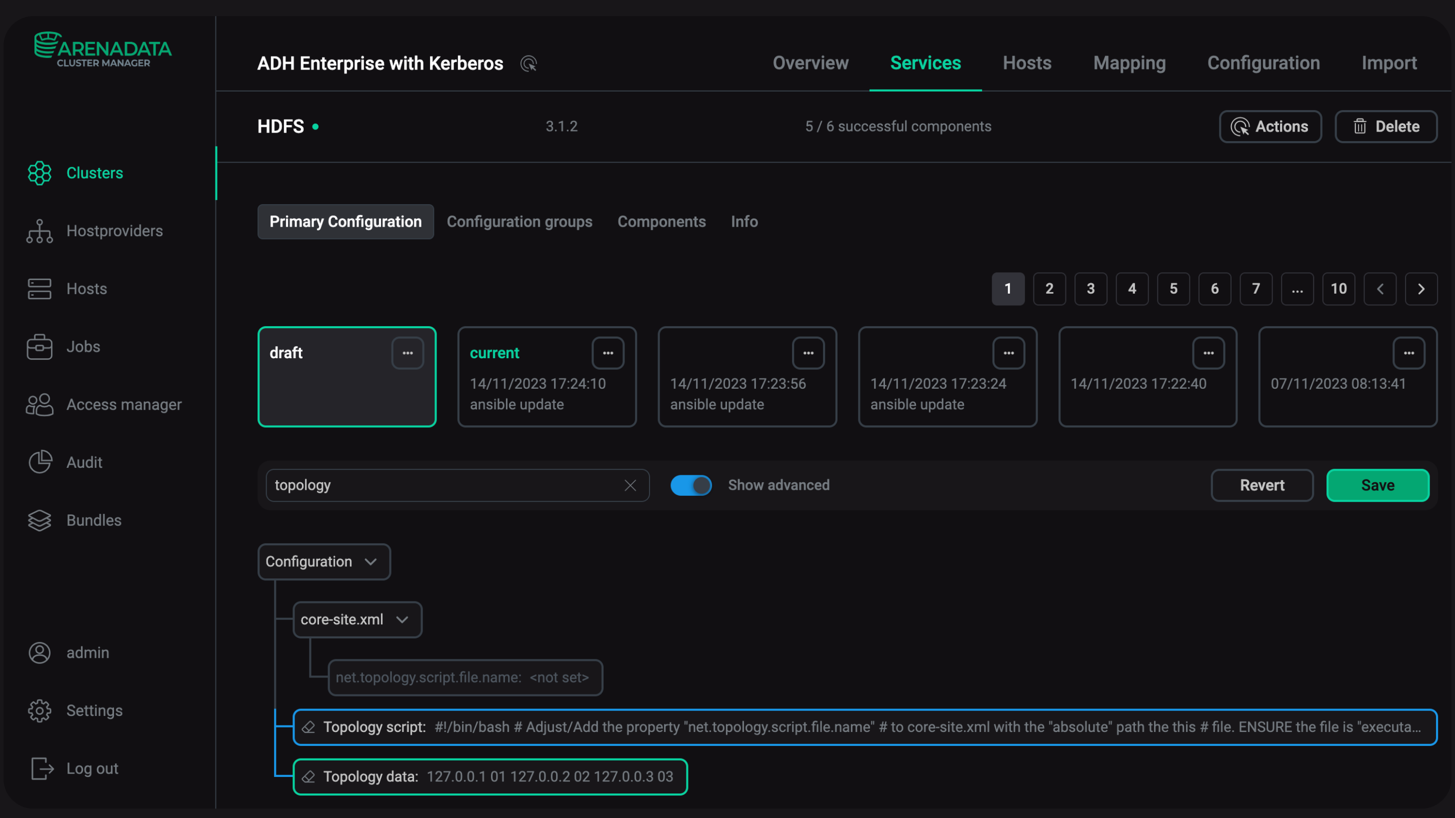
Пример Topology data

```
# This file should be:
#  - Placed in the /etc/hadoop/conf directory
#    - On the Namenode (and backups IE: HA, Failover, etc)
#    - On the Job Tracker OR Resource Manager (and any Failover JT's/RM's)
# This file should be placed in the /etc/hadoop/conf directory.

# Add Hostnames to this file. Format <host ip> <rack_location>
127.0.0.1 01
127.0.0.2 02
127.0.0.3 03
```

6. Подтвердите изменения в конфигурации HDFS, нажав ***Save***.

7. В меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

8. Перезапустите сервис YARN таким же образом.


ARENADATA CLUSTER MANAGER
ADH Enterprise with Kerberos
Overview
Services
Hosts
Mapping
Configuration
Import
HDFS
3.1.2
5 / 6 successful components
Actions
Delete
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Primary Configuration
Configuration groups
Components
Info
1
2
3
4
5
6
7
...
10
draft
current
14/11/2023 17:24:10
ansible update
14/11/2023 17:23:56
ansible update
14/11/2023 17:23:24
ansible update
14/11/2023 17:22:40
07/11/2023 08:13:41
Show advanced
Revert
Save
Configuration
core-site.xml

topology

net.topology.script.file.name: <not set>

Topology script: #!/bin/bash # Adjust/Add the property "net.topology.script.file.name" # to core-site.xml with the "absolute" path the this # file. ENSURE the file is "executa...

Topology data: 127.0.0.1 01 127.0.0.2 02 127.0.0.3 03

Параметры Topology script и Topology data в ADCM

Проверить, успешно ли настроен rack awareness, можно просмотрев логи NameNode или использовав команду fsck.

Логи NameNode будут содержать подобные сообщения:

```
2023-10-23 13:09:52,474 INFO org.apache.hadoop.net.NetworkTopology: Adding a new node: /default/rack_03/127.0.0.1:1019
2023-10-23 13:09:52,569 INFO org.apache.hadoop.net.NetworkTopology: Adding a new node: /default/rack_01/127.0.0.2:1019
2023-10-23 13:09:52,571 INFO org.apache.hadoop.net.NetworkTopology: Adding a new node: /default/rack_02/127.0.0.3:1019
```

Вывод команды `fsck` будет отображать количество стоек:

```
Status: HEALTHY
 Number of data-nodes:  3
 Number of racks:               3
```

Erasure coding

Елена Костюченко

Содержание

Назад к содержанию

Обзор
Политики
Требования
Управление EC

Обзор

Erasure coding (EC) — это механизм обеспечения сохранности редко используемых данных, в котором применяется шифрование вместо репликации. Он позволяет снизить потребление ресурсов для «теплых» и «холодных» данных при сохранении отказоустойчивости.

Фактор репликации, равный трем, позволяет HDFS выдержать два сбоя данных и продолжать работу. При использовании репликации отказоустойчивость гарантируется за счет дискового пространства. С фактором репликации три — это 200% накладных расходов. Erasure coding обеспечивает такой же уровень отказоустойчивости или выше, но стоит значительно меньше: например, файл с политикой EC RS-3-2-1024k выдержит так же два сбоя данных, но потребует лишь 50% накладных расходов.

Когда к файлу применяется политика erasure coding, он разделяется на небольшие блоки, называемые **stiping cell**. Затем HDFS использует алгоритмы шифрования на паре блоков stiping cells чтобы создать **контрольный блок (parity block)**. Если блок stiping cell поврежден или утерян, parity-блок и сохранившиеся блоки stiping cell можно использовать для восстановления отсутствующих. Этот процесс называется расшифровкой (decoding).

Erasure coding требует дополнительных вычислительных ресурсов и использования сети: данные необходимо шифровать во время операций записи и расшифровывать для операций чтения, если целостность данных потеряна.

> **ПРИМЕЧАНИЕ**
>
> Поскольку erasure coding является альтернативой репликации, любые операции, связанные с фактором репликации (например, команда `-setrep`), не затронут файлы, для которых действует политика erasure coding.

Более подробную информацию о репликации можно получить в статье Фактор репликации.

Политики

Политика erasure coding определяет выбор алгоритма шифрования данных и степень отказоустойчивости. Название политики выглядит как `RS-10-4-1024k` и содержит следующую информацию:

- Тип кодека: Reed-Solomon (RS) или XOR.
- Количество блоков данных (stiping cell).
- Количество parity-блоков.
- Размер блока stiping cell.

На данный момент поддерживаются пять встроенных политик: RS-3-2-1024k, RS-6-3-1024k, RS-10-4-1024k, RS-LEGACY-6-3-1024k, XOR-2-1- 1024k.

Вы можете также использовать собственную политику. Для этого необходимо создать XML-файл *user_ec_policies.xml*, описывающий настройки политики, и добавить новую политику, использовав команду `hdfs ec -addPolicies`.

Для получения дополнительной информации о том, как создать этот файл, можно обратиться к примеру user_ec_policies.xml.template.

Требования

Erasure coding предъявляет дополнительные требования к аппаратному обеспечению и архитектуре кластеров:

- **CPU**. Шифрование и расшифровка данных создает нагрузку на CPU как на HDFS-клиентах, так и на DataNodes.
- **DataNodes**. Для erasure coding необходимо минимальное количество DataNodes, равное сумме блоков stiping cell и parity-блоков. Например, для политики RS-3-2-1024k минимальное количество DataNodes будет 5 (3 stiping cell и 2 parity-блока). Это обеспечивает корректную работу политики EC. Если кластер не удовлетворяет этому требованию, HDFS все равно позволит установить политику, но она не будет работать.
- **Стойки**. Erasure coding может использовать информацию о расположении DataNodes по стойкам (rack awareness). Желательно, чтобы количество стоек было больше или равно количеству parity-блоков. Для кластеров с меньшим количеством стоек HDFS не сможет поддерживать отказоустойчивость стойки, но все равно попытается распределить parity-блоки по нескольким DataNodes. По этой причине рекомендуется устанавливать стойки с одинаковым количеством узлов данных. Для получения дополнительной информации о rack awareness можно обратиться к статье Rack awareness.

Управление EC

Erasure coding можно применить только к директории. После применения политики изменения затронут только новые файлы, добавленные в директорию. Это позволяет файлам с разными политиками EC и реплицированным файлам находиться в одном каталоге.

Установка или изменение политики для файла требует перезаписи его данных. Если вы хотите применить EC к существующим файлам, создайте новый каталог, установите для него политику EC и скопируйте туда файлы с помощью команды distcp.

Для управления erasure coding используйте команды hdfs ec.

Например, чтобы установить политику XOR-2-1-1024k для директории:

1. Убедитесь, что политика включена, выполнив команду:

   ```
   $ hdfs ec -listPolicies
   ```

   Пример вывода команды:

   ```
   ErasureCodingPolicy=[Name=RS-10-4-1024k, Schema=[ECSchema=[Codec=rs, numDataUnits=10, numParityUnits=4]], CellSize=1048576,
   Id=5], State=DISABLED
   ErasureCodingPolicy=[Name=RS-3-2-1024k, Schema=[ECSchema=[Codec=rs, numDataUnits=3, numParityUnits=2]], CellSize=1048576, Id=2],
   State=DISABLED
   ErasureCodingPolicy=[Name=RS-6-3-1024k, Schema=[ECSchema=[Codec=rs, numDataUnits=6, numParityUnits=3]], CellSize=1048576, Id=1],
   State=ENABLED
   ErasureCodingPolicy=[Name=RS-LEGACY-6-3-1024k, Schema=[ECSchema=[Codec=rs-legacy, numDataUnits=6, numParityUnits=3]],
   CellSize=1048576, Id=3], State=DISABLED
   ErasureCodingPolicy=[Name=XOR-2-1-1024k, Schema=[ECSchema=[Codec=xor, numDataUnits=2, numParityUnits=1]], CellSize=1048576,
   Id=4], State=DISABLED
   ```

   Прежде чем применить политику, ее необходимо включить. Политика RS-6-3-1024k включена по умолчанию.

2. Чтобы включить политику XOR-2-1-1024k, используйте команду:

   ```
   $ hdfs ec -enablePolicy -policy XOR-2-1-1024k
   ```

3. Чтобы установить включенную политику для директории *tmp*, используйте команду:

   ```
   $ hdfs ec -setPolicy -path /tmp -policy XOR-2-1-1024k
   ```

4. Проверьте политику каталога:

   ```
   $ hdfs ec -getPolicy -path /tmp
   ```

   Вывод команды должен быть таким:

   ```
   XOR-2-1-1024k
   ```

Безопасный режим NameNode

Елена Костюченко

Назад к содержанию

Безопасный режим (safemode) — это состояние NameNode, в котором он не реплицирует блоки и не допускает никаких изменений в файловой системе. Операции, не предполагающие каких-либо изменений в блоках данных или системе, например операции чтения, по-прежнему доступны.

NameNode автоматически переходит в безопасный режим во время запуска, чтобы синхронизировать метаданные с DataNodes. После загрузки состояния файловой системы из файлов *fsimage* и *edits log* NameNode ожидает отчетов о блоках от DataNodes. Если в отчетах количество реплицированных блоков находится на допустимом пороге или выше, NameNode выходит из безопасного режима.

На главной странице NameNode отображается информация о том, включен или выключен безопасный режим. Более подробную информацию о пользовательском интерфейсе NameNode можно получить в статье Обзор HDFS UI.

Вы можете перевести NameNode в состояние безопасного режима с помощью команды dfs admin -safemode.

Чтобы включить безопасный режим, выполните:

```
$ hdfs dfsadmin -safemode enter
```

Пример вывода команды:

```
Safe mode is ON in elenas-adh.ru-central1.internal/127.0.0.1:8020
Safe mode is ON in elenas-adh1.ru-central1.internal/127.0.0.2:8020
```

Если вы включили состояние безопасного режима NameNode вручную, отключить его тоже необходимо вручную.

Чтобы отключить безопасный режим, выполните:

```
$ hdfs dfsadmin -safemode leave
```

Пример вывода команды:

```
Safe mode is OFF in elenas-adh.ru-central1.internal/127.0.0.1:8020
Safe mode is OFF in elenas-adh1.ru-central1.internal/127.0.0.2:8020
```

В некоторых случаях NameNode не выходит из безопасноого режима. Вы можете принудительно отключить его, но это может повлечь потерю метаданных.

Чтобы принудительно отключить безопасный режим, используйте команду:

```
$ hdfs dfsadmin -safemode forceExit
```

Снепшоты в HDFS

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
- Разрешение на создание снепшотов
- Путь к снепшотам
- Команды CLI
  - Разрешение снепшотов
  - Запрет снепшотов
  - Список snapshottable-директорий
  - Список снепшотов
  - Список файлов в снепшоте
  - Восстановление из снепшота
  - Создание снепшота
  - Удаление снепшота
  - Переименование снепшота
  - Сравнение снепшотов

Обзор

Снепшот — это доступный только для чтения образ состояния метаданных файловой системы на определенный момент времени. При создании снепшота копируются только метаданные файла: список блоков и его размер. Несмотря на то, что сами данные не сохраняются, снепшоты используются для резервного копирования и экстренного восстановления данных.

В HDFS можно создать снепшот определенной директории или всей системы. Это требует незначительного количества ресурсов кластера и выполняется мгновенно.

Разрешение на создание снепшотов

Прежде чем сделать снепшот, администратор должен разрешить создание снепшотов для директории, поставив для нее статус **snapshottable**. Это можно сделать с помощью команды allowSnapshot.

Только одна директория в пути может иметь статус snapshottable. Вложенные snapshottable-директории не допускаются. Это значит, что для директории нельзя разрешить снепшоты, если они разрешены для ее родительской или дочерней директории.

При создании снепшота система создает каталог *.snapshot*, в котором будут храниться снепшоты. Каждая директория, для которой разрешены снепшоты, может одновременно содержать до 65536 снепшотов. Директорию, у которой есть снепшоты, нельзя удалить или переименовать пока не будут удалены все ее снепшоты.

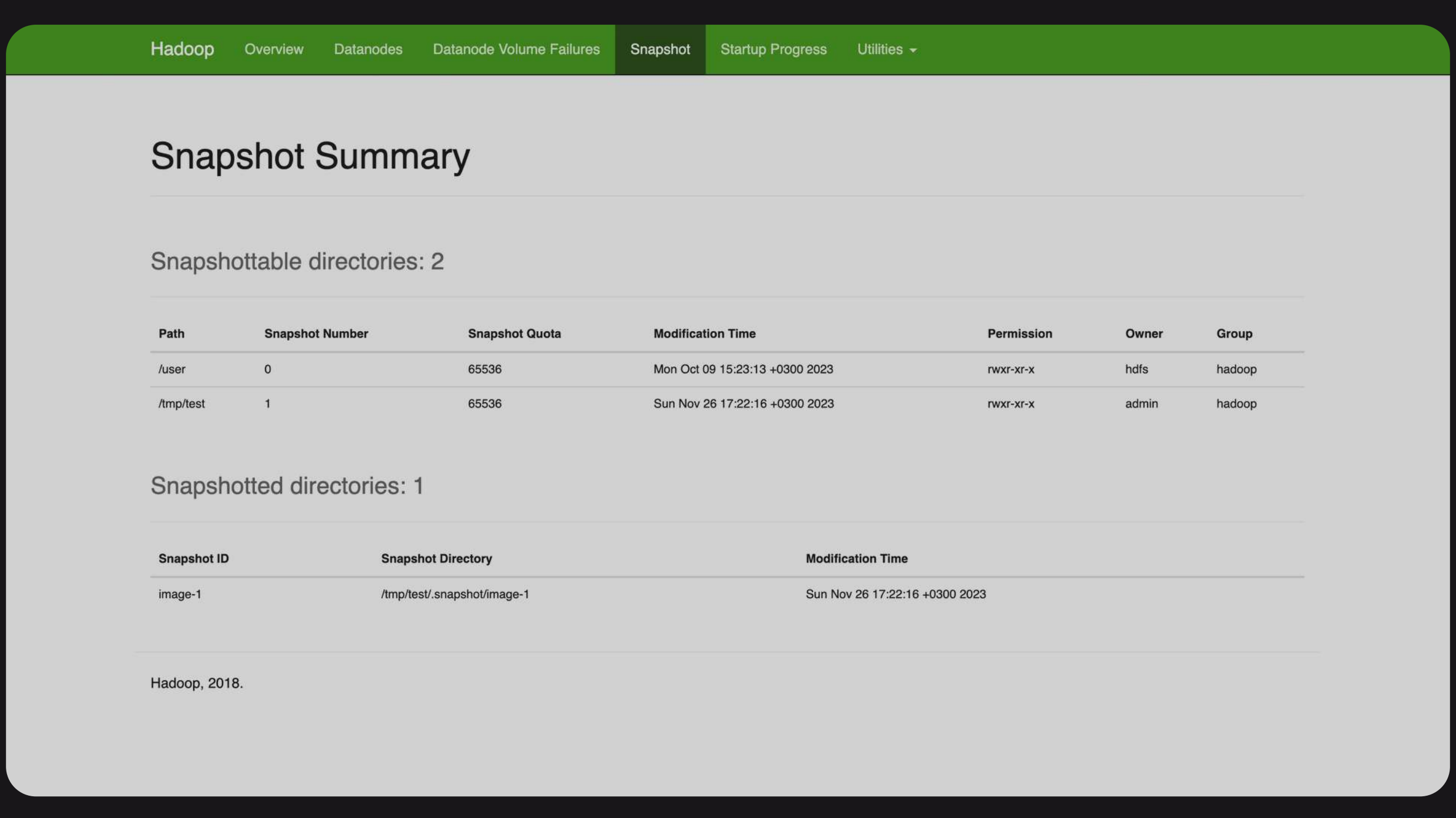
> **ПРИМЕЧАНИЕ**
>
> Создание снепшотов не поддерживается в версиях Hadoop, выпущенных до версии 3.1.2. При обновлении со старой версии убедитесь, что в системе нет директорий с именем *.snapshot*, чтобы избежать конфликтов с зарезервированным путем.

Список директорий, для которых разрешено создание снепшотов, и доступных для них снепшотов, можно найти в пользовательском интерфейсе NameNode на странице ***Snapshots***.

Страница доступна по адресу:

`http://<HOST>:9870/dfshealth.html#tab-snapshot/`

Здесь `<HOST>` это IP-адрес или FQDN хоста NameNode.



Snapshot Summary

Snapshottable directories: 2

Path	Snapshot Number	Snapshot Quota	Modification Time	Permission	Owner	Group
/user	0	65536	Mon Oct 09 15:23:13 +0300 2023	rwxr-xr-x	hdfs	hadoop
/tmp/test	1	65536	Sun Nov 26 17:22:16 +0300 2023	rwxr-xr-x	admin	hadoop

Snapshotted directories: 1

Snapshot ID	Snapshot Directory	Modification Time
image-1	/tmp/test/.snapshot/image-1	Sun Nov 26 17:22:16 +0300 2023

Hadoop, 2018.

NameNode UI: страница Snapshots

Путь к снепшотам

Чтобы обращаться к файлам в снепшоте с помощью вызовов CLI и API, укажите путь следующим образом:

`/<PATH>/.snapshot/<NAME>/<FILE>`

Здесь:

- `<PATH>` — путь к директории, для которой сделан снепшот;
- `<NAME>` — имя нужного снепшота;
- `<FILE>` — относительный путь к нужному файлу в директории.

Команды CLI

Разрешение снепшотов

Разрешает создание снепшотов для указанной директории. Для выполнения команды требуются права суперпользователя HDFS.

```
$ hdfs dfsadmin -allowSnapshot <path>
```

Здесь `<path>` — это путь к директории, для которой нужно разрешить создание снепшотов.

Запрет снепшотов

Снимает разрешение на создание снепшотов для указанной директории. Перед снятием разрешения необходимо удалить все снепшоты каталога. Для выполнения команды требуются права суперпользователя HDFS.

```
$ hdfs dfsadmin -disallowSnapshot <path>
```

Здесь `<path>` — это путь к директории, для которой нужно снять разрешение на создание снепшотов.

Список snapshottable-директорий

Возвращает список директорий, принадлежащих текущему пользователю, для которых разрешены снепшоты. При запуске от имени суперпользователя возвращает список всех директорий системы, для которых разрешены снепшоты.

```
$ hdfs lsSnapshottableDir
```

Список снепшотов

Возвращает список всех снепшотов директории.

```
$ hdfs dfs -ls /<DIR>/.snapshot
```

Здесь `<DIR>` — имя директории, для которой разрешены снепшоты.

Список файлов в снепшоте

Возвращает список каталогов и файлов в снепшоте.

```
$ hdfs dfs -ls /<DIR>/.snapshot/<NAME>/
```

Здесь:

- `<DIR>` — имя директории, для которой разрешены снепшоты;
- `<NAME>` — имя снепшота.

Восстановление из снепшота

Чтобы восстановить файл или каталог из снепшота, используйте команду копирования, например cp или distcp, передав путь к снимку в качестве источника копирования.

Пример команды для восстановления удаленного файла *text.txt* путем копирования его из снимка *image-1*:

```
$ hdfs dfs -cp -ptopax hdfs://127.0.0.1:8020/tmp/test/.snapshot/image-1/text.txt hdfs://127.0.0.1:8020/tmp/test/text.txt
```

Опция `-ptopax` используется для сохранения времени создания файла, информации о владельце, разрешений, списков ACL и XAttr.

Создание снепшота

Создает снепшот указанной директории. Для директории должны быть разрешены снепшоты.

```
$ hdfs dfs -createSnapshot <path> <snapshotName>
```

Здесь:

- `<path>` — путь к директории, для которой разрешены снепшоты.
- `<snapshotName>` — имя снепшота. Это необязательный аргумент. Если имя не указано, система сгенерирует имя, используя метку времени.

Удаление снепшота

Удаляет указанный снепшот.

```
$ hdfs dfs -deleteSnapshot <path> <snapshotName>
```

Здесь:

- `<path>` — путь к директории, для которой разрешены снепшоты;
- `<snapshotName>` — имя снепшота.

Переименование снепшота

Переименовывает указанный снепшот.

```
$ hdfs dfs -renameSnapshot <path> <oldName> <newName>
```

Здесь:

- `<path>` — путь к директории, для которой разрешены снепшоты;
- `<oldName>` — текущее имя снепшота;
- `<newName>` — новое имя снепшота.

Сравнение снепшотов

Выводит отчет о разнице между двумя снимками или между снимком и текущим состоянием директории.

```
$ hdfs snapshotDiff <path> <start> <end>
```

Здесь:

- `<path>` — путь к директории, для которой разрешены снепшоты;
- `<start>` — имя исходного снепшота или `.`, если необходимо сравнить с текущим состоянием директории;
- `<end>` — имя последующего снепшота или `.`, если необходимо сравнить с текущим состоянием директории.

Интерпретация результатов:

- `+` — файл или каталог был создан;
- `-` — файл или каталог был удален;
- `M` — файл или каталог был изменен;
- `R` — файл или каталог был переименован.

Пример отчета:

```
Difference between snapshot image-1 and current directory under directory /tmp/test:
M       .
+       ./text-2.txt
+       ./text.txt
-       ./text.txt
```

Процесс восстановления NameNode в HDFS

Elena Kostyuchenko

Содержание

Назад к содержанию

Прежде чем начать восстановление

Процесс восстановления

Действия при восстановлении

Прежде чем начать восстановление

ВНИМАНИЕ

Ручное восстановление NameNode может привести к потере данных. Не начинайте восстановление, если есть другая действительная копия файлов edits и *fsimage*. Например, на другой NameNode. Использовать рабочую копию безопасней, чем пытаться восстановить поврежденную. Если необходимо ручное восстановление NameNode, предварительно сохраните все метаданные.

Команда `recovery` в Hadoop используется для восстановления потерянных или поврежденных метаданных на NameNode, когда другие способы восстановления недоступны. Например, в кластерах без резервных NameNode и контрольных точек.

Для кластеров высокой доступности восстановление NameNode вручную практически никогда не требуется. Сервисов Journal Nodes и резервных NameNode обычно достаточно для обеспечения целостности метаданных.

Чтобы избежать повреждения метаданных, вы можете:

- Хранить несколько копий метаданных NameNode на разных дисках.
- Настроить режим высокой доступности или использовать checkpointing.

Процесс восстановления

Чтобы вручную восстановить NameNode после сбоя:

1. Подключитесь к хосту NameNode через SSH.
2. От имени пользователя `root` создайте директорию *logs* для запуска команды восстановления и предоставьте пользователю `hdfs` необходимые разрешения:

```
mkdir /usr/lib/hadoop/logs
chown -R hdfs:hadoop /usr/lib/hadoop/logs
chmod -R 755 /usr/lib/hadoop/logs
```

3. Войдите в систему как пользователь `hdfs` и включите безопасный режим NameNode SafeMode:

```
su hdfs
$ hdfs dfsadmin -safemode enter
```

4. Отключите NameNode от имени пользователя `root`:

```
systemctl stop hadoop-hdfs-namenode
```

5. Запустите команду recovery (используйте опцию `-force`, если нужно пропустить выбор) от имени пользователя `hdfs`:

```
$ hdfs namenode -recover
```

Появится следующее сообщение:

```
You have selected Metadata Recovery mode.  This mode is intended to recover lost metadata on a corrupt filesystem.  Metadata
recovery mode often permanently deletes data from your HDFS filesystem.  Please back up your edit log and fsimage before trying
this!

Are you ready to proceed? (Y/N)
 (Y or N)
```

6. Нажмите `Y`, чтобы начать процесс восстановления. Система будет выводить в консоль статус процесса восстановления и при необходимости запрашивать выбор действия. После завершения восстановления в выводе команды появится сообщение:

```
2023-12-06 11:27:00,283 INFO namenode.MetaRecoveryContext: RECOVERY COMPLETE
2023-12-06 11:27:00,318 INFO namenode.FSNamesystem: Stopping services started for active state
2023-12-06 11:27:00,421 INFO namenode.FSNamesystem: Stopping services started for standby state
2023-12-06 11:27:00,423 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at elenas-adh2.ru-central1.internal/127.0.0.1
```

7. Запустите NameNode от имени пользователя `root`:

```
systemctl start hadoop-hdfs-namenode
```

8. Отключите безопасный режим от имени пользователя `hdfs`:

```
$ hdfs dfsadmin -safemode leave
```

Действия при восстановлении

Во время восстановления системе может потребоваться вмешательство пользователя. При возникновении ошибки необходимо выбрать одно из четырех действий:

- `c` (Continue) — проигнорировать ошибку и попытаться сохранить оставшиеся данные.
- `s` (Stop) — прекратить чтение edits-файлов и не пытаться сохранить оставшиеся данные. В этом случае непрочитанные данные будут безвозвратно утеряны.
- `q` (Quit) — завершить команду восстановления без сохранения.
- `a` (Always) — всегда выбирать вариант `c`. При возникновении ошибок система автоматически выберет действие `c`, не предлагая пользователю выбор.

Если на старте команды использована опция `-force`, система всегда выбирает первое действие, то есть `c`.

Настройка HDFS

Альберт Багдасарян

Содержание

Назад к содержанию

core-site.xml

hdfs-site.xml

httpfs-site.xml

hadoop-env.sh

ranger-hdfs-audit.xml

ranger-hdfs-security.xml

Для настройки HDFS существуют два способа:

- Использование Arenadata Cluster Manager.
- Редактирование файлов конфигурации с помощью любого текстового редактора.

Есть несколько основных файлов настройки:

- *core-site.xml*
- *hdfs-site.xml*
- *httpfs-site.xml*
- *ranger-hdfs-audit.xml*
- *ranger-hdfs-security.xml*
- *hadoop-env.sh*

Все эти файлы хранятся в каталоге *<hadoop_home>/etc/hadoop/*. Вы можете редактировать их, используя удобный для вас текстовый редактор.

> **ПРИМЕЧАНИЕ**
>
> Рекомендуемым инструментом для настройки кластера ADH является Arenadata Cluster Manager (ADCM).

core-site.xml

Это самый важный файл конфигурации для HDFS. Его применяют для настройки текущего окружения Hadoop. Например, в нем указывают на каком узле работает NameNode или какие порты TCP используются.

За более подробной информацией обратитесь к разделу Параметры core-site.xml.

hdfs-site.xml

Данный файл содержит параметры настройки серверов NameNode и DataNode. Он также определяет размер блока для процесса репликации.

За более подробной информацией обратитесь к разделу Параметры hdfs-site.xml.

httpfs-site.xml

Вы можете использовать сервис HttpFS для взаимодействия с HDFS. В этом случае потребуется настраивать файл *httpfs-site.xml*. Этот файл нужен для следующих целей:

- **Высокая надежность доступа**. Сервис WebHDFS не поддерживает автоматическое замещение сервера при отказе (High Availability failover). Для обеспечения этой функции наилучшей заменой этого сервиса является сервис HttpFS.
- **Подмена пользователя (impersonation)**. Пользователь, от имени которого работает сервис HDFS, не доступен для подмены. Если вы разрешили подмену пользователя в окружении с ограниченным использованием для этой цели суперпользователя HDFS, вы можете разрешить использование сервиса HttpFS и использовать суперпользователя HttpFS для имперсонации.

> **РЕКОМЕНДАЦИЯ**
>
> Если вы включили сервис HttpFS с использованием высокой надежности доступа (High Availability), исключите этот сервис из использования на главном узле, то есть NameNode, кластера ADH.

За более подробной информацией обратитесь к разделу Параметры httpfs-site.xml.

hadoop-env.sh

Этот файл используют для настройки параметров, относящихся к операционному окружению. К таким относится, например, `Java_HOME`. Hadoop использует JRE, и одной из переменных окружения является `Java_Home` в файле *hadoop-env.sh*.

ranger-hdfs-audit.xml

Этот файл содержит параметры настройки аудиторского слежения и анализа политик в целях более глубокого наблюдения за рабочим окружением.

За более подробной информацией обратитесь к разделу Параметры ranger-hdfs-audit.xml.

ranger-hdfs-security.xml

Путем настройки Ranger можно разрешить работу Apache Ranger plugin для сервиса HDFS. Администратор системы безопасности имеет возможность настраивать уровни управления авторизацией для проверки прав доступа к HDFS с помощью параметров настройки в файле *ranger-hdfs-security.xml*.

Когда вы установите все параметры настройки, механизм авторизации (authorization engine) будет проверять списки управления доступом (Access Control Lists, ACL) HDFS. Если ACL запрещен путем установки соответствующего параметра в `false`, то решение о предоставлении доступа обеспечивает политика Ranger без проверки списков доступа HDFS. Когда пользователь пытается получить доступ к данным через такие сервисы как Hive, происходит проверка политик доступа как самого сервиса, так и HDFS.

За более подробной информацией обратитесь к разделу Параметры ranger-hdfs-security.xml.

Checkpointing в HDFS

Елена Костюченко

Содержание

Назад к содержанию

Обзор
Конфигурация
Команды CLI
Импорт контрольной точки

Обзор

Создание контрольных точек (checkpointing) в HDFS — это функция, оптимизирующая работу с метаданными, которая также может использоваться для аварийного восстановления данных.

Сервис NameNode загружает *fsimage* и применяет изменения из файлов edits только во время запуска. Если HDFS накопит слишком много изменений, edits-файлы займут много дискового пространства, а NameNode будет запускаться дольше.

Чтобы этого не случилось, сервис, создающий контрольные точки, периодически обрабатывает edits-файлы и создает новый *fsimage*. Это помогает сократить размер edits-файлов и ускорить запуск NameNode.

Создание нового *fsimage* создает вычислительную нагрузку на кластер. Чтобы поддерживать доступность NameNode во время этой операции, функция обновления *fsimage* была также реализована в Secondary NameNode и Standby NameNode.

В обычном кластере Hyperwave Secondary NameNode отвечает за обработку edits-файлов и обновление *fsimage*. Чтобы повысить устойчивость кластера к сбоям, вы можете хранить локальную актуальную копию файловой системы, включив либо Checkpointing node, либо Backup node.

> **ВАЖНО**
>
> В кластерах Hyperwave с высокой доступностью (High Availability, HA) *fsimage* периодически создается сервисом Standby NameNode, а edits-файлы обрабатываются и синхронизируются сервисами Quorum Journal Manager. По этой причине в кластерах высокой доступности нет необходимости использовать сервисы Checkpoint node и Backup node.

Для получения дополнительной информации о компонентах HDFS обратитесь к статье Архитектура HDFS.

Конфигурация

Создавать контрольные точки может один из двух сервисов:

- **Checkpoint node** — периодически перерабатывает edits-файлы и обновляет *fsimage* на активном NameNode.
- **Backup node** — выполняет те же функции, что и Checkpoint node, и хранит в памяти актуальную копию пространства имен файловой системы, синхронизированную с активным NameNode.

В HDFS можно использовать только либо Checkpoint node, либо Backup node. Оба сервиса одновременно запустить нельзя.

Сервисы создания контрольных точек имеют те же требования к памяти, что и обычный NameNode. Попытка запустить их на одном хосте, скорее всего, приведет к ошибке.

Параметры, в которых задаются условия создания контрольной точки:

- `dfs.namenode.checkpoint.period` — промежуток времени между контрольными точками. Для указания времени можно использовать сокращение: ms (миллисекунды), s (секунды), m (минуты), h (часы), d (дни). Значение по умолчанию: 3600s.
- `dfs.namenode.checkpoint.txns` — количество записей edits, при достижении которого будет создана контрольная точка. Значение по умолчанию: 1000000.

В значении параметра `dfs.namenode.checkpoint.check. period` можно указать как часто сервис создания контрольных точек должен проверять выполнение любого из этих условий.

Чтобы настроить сервис создания контрольных точек с помощью ADCM, выполните следующие действия:

Checkpoint node

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на HDFS.
3. Включите опцию ***Show Advanced*** и найдите поле ***Custom hdfs-site.xml***.
4. Добавьте параметр `dfs.namenode.backup.address` и его значение `<host>:50100`, где `<host>` — это FQDN хоста для Checkpoint node. В одном кластере можно создать несколько Checkpoint node.
5. Опционально, добавьте параметр `dfs.namenode.backup.http-address` и его значение `<host>:50105`, где `<host>` — это FQDN хоста для Checkpoint node.
6. Перезапустите HDFS с параметром ***Apply configs from ADCM***.
7. Подключитесь к хосту для Checkpoint node через SSH и выполните команду от имени суперпользователя HDFS:

```
$ hdfs namenode -checkpoint
```

Backup node

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на HDFS.
3. Включите параметр ***Show Advanced*** и найдите поле ***Custom hdfs-site.xml***.
4. Добавьте параметр `dfs.namenode.backup.address` и его значение `<host>:50100`, где `<host>` — это FQDN хоста для Backup node. В одном кластере может быть запущен только один Backup node.
5. Опционально, добавьте параметр `dfs.namenode.backup.http-address` и его значение `<host>:50105`, где `<host>` — это FQDN хоста для Backup node.
6. Подтвердите изменения в конфигурации HDFS, нажав ***Save***.
7. В меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.
8. Подключитесь к хосту для Backup node через SSH и выполните команду от имени суперпользователя HDFS:

```
$ hdfs namenode -backup
```

> **РЕКОМЕНДАЦИЯ**
>
> При запуске сервиса создания контрольных точек могут возникнуть ошибки, связанные с отсутствием нужных директорий или прав на их редактирование. Их можно решить создав нужные каталоги и установив на них разрешения для пользователя Hadoop.

Команды CLI

Чтобы создать новую контрольную точку вручную, используйте команду:

```
$ hdfs dfsadmin -saveNamespace
```

Чтобы удалить ненужные контрольные точки, используйте команду expunge.

Подробнее о командах для управления сервисами создания контрольных точек читайте в описании следующих команд:

- secondarynamenode
- namenode

Импорт контрольной точки

Чтобы восстановить утерянный *fsimage* или создать новый NameNode из существующей контрольной точки, вы можете импортировать контрольную точку.

Чтобы импортировать контрольную точку в NameNode:

1. Убедитесь, что директория, указанная в параметре `dfs.namenode.name.dir`, существует и в ней нет файлов.
2. Убедитесь, что параметр `dfs.namenode.checkpoint.dir` содержит путь к директории с контрольной точкой.
3. Запустите NameNode, используя опцию `-importCheckpoint`:

```
$ hdfs namenode -importCheckpoint
```

NameNode загрузит контрольную точку из директории, указанной в `dfs.namenode.checkpoint.dir`, и сохранит ее в директории NameNode, заданной в `dfs.namenode.name.dir`.

Интеграция HDFS с S3

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
- Настройка интеграции с S3
  - Подготовительные шаги
  - Зависимости
  - Аутентификация
  - Анонимное соединение
- Подключение к хранилищу
- Копирование данных

Обзор

Simple Storage Service (S3) — это объектное хранилище, доступ к которому осуществляется по протоколу типа S3. Отдельный контейнер хранения в объектном хранилище S3 называется **бакет**. Бакеты могут быть публичными, что означает, что любой может получить доступ к хранилищу, или частными, требующими аутентификации.

Hadoop поддерживает протокол S3A для подключения к хранилищу S3. Это позволяет использовать инструменты командной строки HDFS для взаимодействия с удаленными объектными хранилищами.

В этой статье представлен пример подключения к внешнему хранилищу S3 для копирования данных в локальную HDFS с помощью команды `distcp`.

Дополнительную информацию о возможных конфигурациях HDFS для работы с хранилищем S3 можно получить в статье Hadoop-AWS module: Integration with Amazon Web Services. Поставщики хранилищ S3 могут поддерживать различные инструменты для подключения к хранилищу. Проконсультируйтесь с вашим провайдером S3 для получения информации о других способах интеграции с хранилищем.

Настройка интеграции с S3

Подготовительные шаги

Убедитесь, что на стороне хранилища создан пользователь для HDFS и у него есть необходимый доступ к файлам. Для прохождения аутентификации потребуются идентификатор ключа доступа (access key ID) и секретный ключ доступа (secret access key).

Если бакет публичный, к нему можно подключиться анонимно.

Зависимости

В ADH есть встроенный коннектор для объектного хранилища — S3A. Для корректной работы коннектора необходимо добавить две дополнительные библиотеки в classpath:

- hadoop-aws-3.1.2.jar
- aws-java-sdk-bundle-1.12.191.jar

Добавьте эти библиотеки в директорию *usr/lib/hadoop/lib/* на хостах кластера.

Аутентификация

Чтобы подключиться к частному бакету S3, необходимо предоставить корректные ключи доступа. При подключении к хранилищу HDFS пробует следующие типы аутентификации в таком порядке:

- аргументы в URI (предупреждает о возможном нарушении безопасности);
- конфигурация HDFS;
- переменные среды AWS;
- Amazon EC2 Instance Metadata Service.

Дополнительную информацию о настройке каждого типа аутентификации можно получить в статье Hadoop-AWS module: Integration with Amazon Web Services.

ВНИМАНИЕ

Потеря учетных данных может поставить под угрозу учетную запись. Убедитесь, что ключи доступа не попадают в логи и баг-репорты. По этой причине избегайте передачи ключей в S3A URL-запросах.

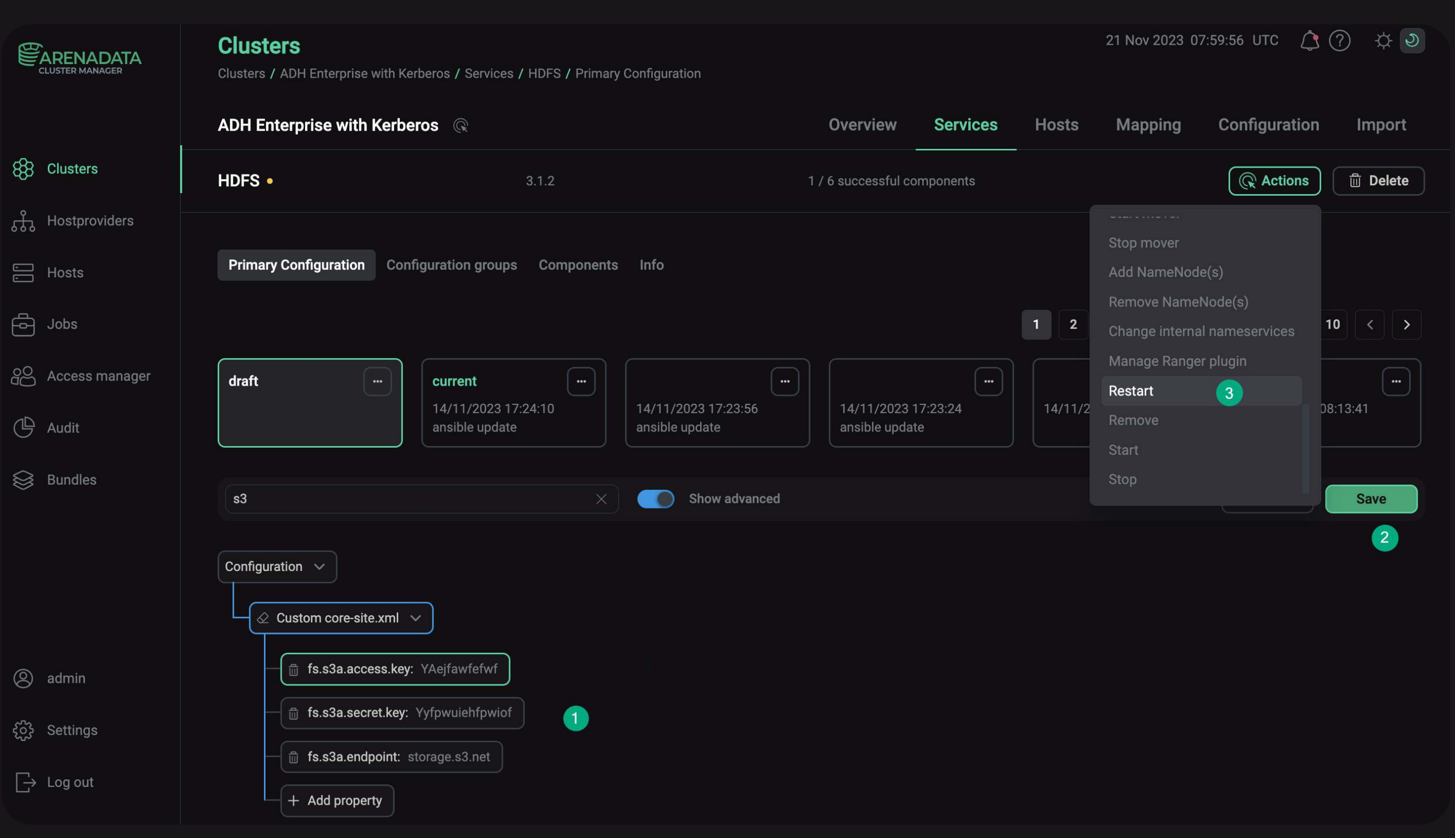
Для аутентификации с использованием конфигурации HDFS добавьте в *core-site.xml* параметры из таблице ниже.

Параметр	Описание	Значение по умолчанию
fs.s3a.access.key	Идентификатор ключа доступа. Не указывать для анонимного подключения	—
fs.s3a.secret.key	Секретный ключ доступа. Не указывать для анонимного подключения	—
fs.s3a.endpoint	S3-эндпоинт для подключения. Обратитесь к своему провайдеру S3 за информацией о том, какой эндпоинт использовать	s3.amazonaws.com

Дополнительные параметры описаны в документации Apache Hadoop — Hadoop-AWS module: Integration with Amazon Web Services.

Чтобы настроить эти параметры с помощью ADCM:

1. Перейдите в раздел ***Clusters*** и выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на HDFS.
3. Включите параметр ***Show advanced***, разверните список свойств ***Custom core-site.xml*** и выберите ***Add property***.
4. Добавьте необходимый параметр с его значением и нажмите ***Apply***.
5. Подтвердите изменения в конфигурации HDFS, нажав ***Save***.
6. В меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters / ADH Enterprise with Kerberos / Services / HDFS / Primary Configuration
21 Nov 2023 07:59:56 UTC
ADH Enterprise with Kerberos
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
HDFS
3.1.2
1 / 6 successful components
Actions
Delete
Primary Configuration
Configuration groups
Components
Info
Stop mover
Add NameNode(s)
Remove NameNode(s)
Change internal nameservices
Manage Ranger plugin
Restart
3
Remove
Start
Stop
1
2
10
draft
current
14/11/2023 17:24:10
ansible update
14/11/2023 17:23:56
ansible update
14/11/2023 17:23:24
ansible update
14/11/2
08:13:41
s3
Show advanced
Save
2
Configuration
Custom core-site.xml
1
Add property

fs.s3a.access.key: YAejfawfefwf
fs.s3a.secret.key: Yyfpwuiehfpwiof
fs.s3a.endpoint: storage.s3.net

Настройка параметров подключения к S3 в ADCM

Анонимное соединение

К публичному бакету S3 можно подключиться анонимно. Для этого укажите `org.apache.hadoop.fs.s3a.AnonymousAWSCredentialsProvider` в качестве значения параметра `fs.s3a.aws.credentials.provider`.

Эту опцию также можно использовать для проверки того, что бакет недоступен для анонимных пользователей. Для этого выполните команду:

```
$ hadoop fs -D fs.s3a.aws.credentials.provider=org.apache.hadoop.fs.s3a.AnonymousAWSCredentialsProvider \
 -ls s3a://<bucket>/
```

Здесь `<bucket>` — это имя бакета, с которым необходимо установить соединение.

Если доступ без аутентификации запрещен, попытка подключения должна завершиться такой ошибкой:

```
ls: s3a://develop-test-bucket/: getFileStatus on s3a://develop-test-bucket/: com.amazonaws.services.s3.model.AmazonS3Exception:
Access Denied (Service: Amazon S3; Status Code: 403; Error Code: AccessDenied; Request ID: dc82c760b5ab707e; S3 Extended Request
ID: null; Proxy: null), S3 Extended Request ID: null:AccessDenied
```

Подключение к хранилищу

Чтобы проверить подключение к хранилищу, выполните команду `-ls`:

```
$ hadoop fs -ls s3a://<bucket>/
```

Здесь `<bucket>` — это имя бакета, с которым необходимо установить соединение.

Пример вывода:

```
Found 3 items
drwxrwxrwx   - admin admin          0 2023-11-14 17:28 s3a://develop-test-bucket/adbm_n1
drwxrwxrwx   - admin admin          0 2023-11-14 17:28 s3a://develop-test-bucket/adbm_n2
drwxrwxrwx   - admin admin          0 2023-11-14 17:28 s3a://develop-test-bucket/repo
```

Копирование данных

Копировать данные между HDFS и S3 можно с помощью команды distcp. Чтобы скопировать данные из S3, выполните команду:

```
$ hadoop distcp \
      s3a://<bucket>/<path>/ \
      hdfs://<host>/<path>/
```

Здесь:

- `<bucket>` — имя бакета, с которым устанавливается соединение;
- `<host>` — nameservice кластера, FQDN или IP-адрес узла NameNode;
- `<path>` — путь к копируемым данным или путь для сохранения данных.

Логирование в HDFS

Елена Костюченко

Содержание

Назад к содержанию

Обзор
Поиск по логам
Настройки логирования
Уровни логирования
Архивирование логов

Обзор

HDFS записывает выходные данные и события процессов файловой системы в текстовые логи. Эта информация может быть полезна при диагностике технических проблем в системе.

Каждый компонент HDFS хранит логи локально, на хосте, на котором он установлен. Логи имеют расширение *.log* и расположены в директории */var/log/hadoop-hdfs/*. Здесь же находятся файлы *.out*, содержащие информацию о запуске компонентов.

Имена файлов с логами имеют следующий формат: `Hadoop-<пользователь>-<компонент>-<хост>.log`.

Здесь:

- `<пользователь>` — пользователь, запускающий Hadoop;
- `<компонент>` — имя компонента, например, DataNode;
- `<хост>` — FQDN хоста компонента.

Просмотреть логи HDFS в браузере можно через веб-интерфейсы NameNodes или DataNodes. Они доступны по ссылкам:

- `http://<HOST>:9864/logs/` — веб-интерфейс DataNode. Здесь `<HOST>` — это IP-адрес или FQDN хоста с DataNode.
- `http://<HOST>/logs/` — веб-интерфейс NameNode. Здесь `<HOST>` — это IP-адрес или FQDN хоста с NameNode.

HDFS использует для логирования библиотеку Log4j. Более подробно о том, как работает библиотека, читайте в документации Apache Logging Services.

Поиск по логам

Чтобы найти в логах интересующую вас информацию, например, сообщения об ошибках, подключитесь к хосту с компонентом, логи которого вы хотите просмотреть, и воспользуйтесь командой `grep`.

Например:

```
$ cat /var/log/hadoop-hdfs/hadoop-hdfs-datanode-elenas-adh3.ru-central1.internal.log | grep -i -A3 -B1 error
```

Эта команда просматривает лог DataNode, расположенной на хосте `elenas-adh3.ru-central1.internal`, и ищет сообщения, содержащие слово `error`. Опция `i` позволяет игнорировать различия в регистре, а опции `-A3 -B1` добавляют к найденной строке с ошибкой одну строчку до нее и три строчки после.

Пример вывода команды:

```
2023-11-29 12:20:12,902 INFO org.apache.hadoop.ipc.Client: Retrying connect to server: elenas-adh2.ru-central1.internal/127.0.0.1:8020. Already tried 2 time(s); maxRetries=45
2023-11-29 12:20:15,588 ERROR org.apache.hadoop.hdfs.server.datanode.DataNode: RECEIVED SIGNAL 15: SIGTERM
2023-11-29 12:20:15,591 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down DataNode at elenas-adh3.ru-central1.internal/127.0.0.1
--
        at org.apache.hadoop.ipc.Client$Connection.run(Client.java:1069)
2023-11-30 12:30:30,031 ERROR org.apache.hadoop.hdfs.server.datanode.DataNode: RECEIVED SIGNAL 15: SIGTERM
2023-11-30 12:30:30,035 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down DataNode at elenas-adh3.ru-central1.internal/127.0.0.1
```

Настройки логирования

Настроить параметры логирования в Hadoop можно в файле конфигурации *log4j.properties*, который расположен на каждом хосте кластера в директории *etc/hadoop/conf/*.

Файл *log4j.properties* содержит общие настройки логирования, а также настройки компонентов логирования: logger и appender.

Параметры, которые вы устанавливаете для logger-компонентов, влияют на то, что программы сообщают системе. Параметры, которые вы устанавливаете для appender-компонентов, влияют на то, какая информация от logger-компонентов попадает в логи.

Файл ADH log4j.properties по умолчанию

```
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements.  See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership.  The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Define some default values that can be overridden by system properties
hadoop.root.logger=INFO,console
hadoop.log.dir=.
hadoop.log.file=hadoop.log

# Define the root logger to the system property "hadoop.root.logger".
log4j.rootLogger=${hadoop.root.logger}, EventCounter

# Logging Threshold
log4j.threshold=ALL

# Null Appender
log4j.appender.NullAppender=org.apache.log4j.varia.NullAppender

#
# Rolling File Appender - cap space usage at 5gb.
#
hadoop.log.maxfilesize=256MB
hadoop.log.maxbackupindex=20
log4j.appender.RFA=org.apache.log4j.RollingFileAppender
log4j.appender.RFA.File=${hadoop.log.dir}/${hadoop.log.file}

log4j.appender.RFA.MaxFileSize=${hadoop.log.maxfilesize}
log4j.appender.RFA.MaxBackupIndex=${hadoop.log.maxbackupindex}

log4j.appender.RFA.layout=org.apache.log4j.PatternLayout

# Pattern format: Date LogLevel LoggerName LogMessage
log4j.appender.RFA.layout.ConversionPattern=%d{ISO8601} %p %c: %m%n
# Debugging Pattern format
#log4j.appender.RFA.layout.ConversionPattern=%d{ISO8601} %-5p %c{2} (%F:%M(%L)) - %m%n


#
# Daily Rolling File Appender
#

log4j.appender.DRFA=org.apache.log4j.DailyRollingFileAppender
log4j.appender.DRFA.File=${hadoop.log.dir}/${hadoop.log.file}

# Rollover at midnight
log4j.appender.DRFA.DatePattern=.yyyy-MM-dd

log4j.appender.DRFA.layout=org.apache.log4j.PatternLayout

# Pattern format: Date LogLevel LoggerName LogMessage
log4j.appender.DRFA.layout.ConversionPattern=%d{ISO8601} %p %c: %m%n
# Debugging Pattern format
#log4j.appender.DRFA.layout.ConversionPattern=%d{ISO8601} %-5p %c{2} (%F:%M(%L)) - %m%n


#
# console
# Add "console" to rootlogger above if you want to use this
#

log4j.appender.console=org.apache.log4j.ConsoleAppender
log4j.appender.console.target=System.err
log4j.appender.console.layout=org.apache.log4j.PatternLayout
log4j.appender.console.layout.ConversionPattern=%d{ISO8601} %p %c{2}: %m%n

#
# TaskLog Appender
#
log4j.appender.TLA=org.apache.hadoop.mapred.TaskLogAppender

log4j.appender.TLA.layout=org.apache.log4j.PatternLayout
log4j.appender.TLA.layout.ConversionPattern=%d{ISO8601} %p %c: %m%n

#
# HDFS block state change log from block manager
#
# Uncomment the following to log normal block state change
# messages from BlockManager in NameNode.
#log4j.logger.BlockStateChange=DEBUG

#
#Security appender
#
hadoop.security.logger=INFO,NullAppender
hadoop.security.log.maxfilesize=256MB
hadoop.security.log.maxbackupindex=20
log4j.category.SecurityLogger=${hadoop.security.logger}
hadoop.security.log.file=SecurityAuth-${user.name}.audit
log4j.appender.RFAS=org.apache.log4j.RollingFileAppender
log4j.appender.RFAS.File=${hadoop.log.dir}/${hadoop.security.log.file}
log4j.appender.RFAS.layout=org.apache.log4j.PatternLayout
log4j.appender.RFAS.layout.ConversionPattern=%d{ISO8601} %p %c: %m%n
log4j.appender.RFAS.MaxFileSize=${hadoop.security.log.maxfilesize}
log4j.appender.RFAS.MaxBackupIndex=${hadoop.security.log.maxbackupindex}

#
# Daily Rolling Security appender
#
log4j.appender.DRFAS=org.apache.log4j.DailyRollingFileAppender
log4j.appender.DRFAS.File=${hadoop.log.dir}/${hadoop.security.log.file}
log4j.appender.DRFAS.layout=org.apache.log4j.PatternLayout
log4j.appender.DRFAS.layout.ConversionPattern=%d{ISO8601} %p %c: %m%n
log4j.appender.DRFAS.DatePattern=.yyyy-MM-dd

#
# hadoop configuration logging
#

# Uncomment the following line to turn off configuration deprecation warnings.
# log4j.logger.org.apache.hadoop.conf.Configuration.deprecation=WARN

#
# hdfs audit logging
#
hdfs.audit.logger=INFO,NullAppender
hdfs.audit.log.maxfilesize=256MB
hdfs.audit.log.maxbackupindex=20
log4j.logger.org.apache.hadoop.hdfs.server.namenode.FSNamesystem.audit=${hdfs.audit.logger}
log4j.additivity.org.apache.hadoop.hdfs.server.namenode.FSNamesystem.audit=false
log4j.appender.RFAAUDIT=org.apache.log4j.RollingFileAppender
log4j.appender.RFAAUDIT.File=${hadoop.log.dir}/hdfs-audit.log
log4j.appender.RFAAUDIT.layout=org.apache.log4j.PatternLayout
log4j.appender.RFAAUDIT.layout.ConversionPattern=%d{ISO8601} %p %c{2}: %m%n
log4j.appender.RFAAUDIT.MaxFileSize=${hdfs.audit.log.maxfilesize}
log4j.appender.RFAAUDIT.MaxBackupIndex=${hdfs.audit.log.maxbackupindex}

#
# NameNode metrics logging.
# The default is to retain two namenode-metrics.log files up to 64MB each.
#
namenode.metrics.logger=INFO,NullAppender
log4j.logger.NameNodeMetricsLog=${namenode.metrics.logger}
log4j.additivity.NameNodeMetricsLog=false
log4j.appender.NNMETRICSRFA=org.apache.log4j.RollingFileAppender
log4j.appender.NNMETRICSRFA.File=${hadoop.log.dir}/namenode-metrics.log
log4j.appender.NNMETRICSRFA.layout=org.apache.log4j.PatternLayout
log4j.appender.NNMETRICSRFA.layout.ConversionPattern=%d{ISO8601} %m%n
log4j.appender.NNMETRICSRFA.MaxBackupIndex=1
log4j.appender.NNMETRICSRFA.MaxFileSize=64MB

#
# DataNode metrics logging.
# The default is to retain two datanode-metrics.log files up to 64MB each.
#
datanode.metrics.logger=INFO,NullAppender
log4j.logger.DataNodeMetricsLog=${datanode.metrics.logger}
log4j.additivity.DataNodeMetricsLog=false
log4j.appender.DNMETRICSRFA=org.apache.log4j.RollingFileAppender
log4j.appender.DNMETRICSRFA.File=${hadoop.log.dir}/datanode-metrics.log
log4j.appender.DNMETRICSRFA.layout=org.apache.log4j.PatternLayout
log4j.appender.DNMETRICSRFA.layout.ConversionPattern=%d{ISO8601} %m%n
log4j.appender.DNMETRICSRFA.MaxBackupIndex=1
log4j.appender.DNMETRICSRFA.MaxFileSize=64MB

# Custom Logging levels

#log4j.logger.org.apache.hadoop.mapred.JobTracker=DEBUG
#log4j.logger.org.apache.hadoop.mapred.TaskTracker=DEBUG
#log4j.logger.org.apache.hadoop.hdfs.server.namenode.FSNamesystem.audit=DEBUG


# AWS SDK & S3A FileSystem
#log4j.logger.com.amazonaws=ERROR
log4j.logger.com.amazonaws.http.AmazonHttpClient=ERROR
#log4j.logger.org.apache.hadoop.fs.s3a.S3AFileSystem=WARN

#
# Event Counter Appender
# Sends counts of logging messages at different severity levels to Hadoop Metrics.
#
log4j.appender.EventCounter=org.apache.hadoop.log.metrics.EventCounter

#
# Job Summary Appender
#
# Use following logger to send summary to separate file defined by
# hadoop.mapreduce.jobsummary.log.file :
# hadoop.mapreduce.jobsummary.logger=INFO,JSA
#
hadoop.mapreduce.jobsummary.logger=${hadoop.root.logger}
hadoop.mapreduce.jobsummary.log.file=hadoop-mapreduce.jobsummary.log
hadoop.mapreduce.jobsummary.log.maxfilesize=256MB
hadoop.mapreduce.jobsummary.log.maxbackupindex=20
log4j.appender.JSA=org.apache.log4j.RollingFileAppender
log4j.appender.JSA.File=${hadoop.log.dir}/${hadoop.mapreduce.jobsummary.log.file}
log4j.appender.JSA.MaxFileSize=${hadoop.mapreduce.jobsummary.log.maxfilesize}
log4j.appender.JSA.MaxBackupIndex=${hadoop.mapreduce.jobsummary.log.maxbackupindex}
log4j.appender.JSA.layout=org.apache.log4j.PatternLayout
log4j.appender.JSA.layout.ConversionPattern=%d{ISO8601} %p %c{2}: %m%n
log4j.logger.org.apache.hadoop.mapred.JobInProgress$JobSummary=${hadoop.mapreduce.jobsummary.logger}
log4j.additivity.org.apache.hadoop.mapred.JobInProgress$JobSummary=false

#
# shuffle connection log from shuffleHandler
# Uncomment the following line to enable logging of shuffle connections
# log4j.logger.org.apache.hadoop.mapred.ShuffleHandler.audit=DEBUG

#
# Yarn ResourceManager Application Summary Log
#
# Set the ResourceManager summary log filename
yarn.server.resourcemanager.appsummary.log.file=rm-appsummary.log
# Set the ResourceManager summary log level and appender
yarn.server.resourcemanager.appsummary.logger=${hadoop.root.logger}
#yarn.server.resourcemanager.appsummary.logger=INFO,RMSUMMARY

# To enable AppSummaryLogging for the RM,
# set yarn.server.resourcemanager.appsummary.logger to
# <LEVEL>,RMSUMMARY in hadoop-env.sh

# Appender for ResourceManager Application Summary Log
# Requires the following properties to be set
#    - hadoop.log.dir (Hadoop Log directory)
#    - yarn.server.resourcemanager.appsummary.log.file (resource manager app summary log filename)
#    - yarn.server.resourcemanager.appsummary.logger (resource manager app summary log level and appender)

log4j.logger.org.apache.hadoop.yarn.server.resourcemanager.RMAppManager$ApplicationSummary=${yarn.server.resourcemanager.appsummary.logger}
log4j.additivity.org.apache.hadoop.yarn.server.resourcemanager.RMAppManager$ApplicationSummary=false
log4j.appender.RMSUMMARY=org.apache.log4j.RollingFileAppender
log4j.appender.RMSUMMARY.File=${hadoop.log.dir}/${yarn.server.resourcemanager.appsummary.log.file}
log4j.appender.RMSUMMARY.MaxFileSize=256MB
log4j.appender.RMSUMMARY.MaxBackupIndex=20
log4j.appender.RMSUMMARY.layout=org.apache.log4j.PatternLayout
log4j.appender.RMSUMMARY.layout.ConversionPattern=%d{ISO8601} %p %c{2}: %m%n

# HS audit log configs
#mapreduce.hs.audit.logger=INFO,HSAUDIT
#log4j.logger.org.apache.hadoop.mapreduce.v2.hs.HSAuditLogger=${mapreduce.hs.audit.logger}
#log4j.additivity.org.apache.hadoop.mapreduce.v2.hs.HSAuditLogger=false
#log4j.appender.HSAUDIT=org.apache.log4j.DailyRollingFileAppender
#log4j.appender.HSAUDIT.File=${hadoop.log.dir}/hs-audit.log
#log4j.appender.HSAUDIT.layout=org.apache.log4j.PatternLayout
#log4j.appender.HSAUDIT.layout.ConversionPattern=%d{ISO8601} %p %c{2}: %m%n
#log4j.appender.HSAUDIT.DatePattern=.yyyy-MM-dd

# Http Server Request Logs
#log4j.logger.http.requests.namenode=INFO,namenoderequestlog
#log4j.appender.namenoderequestlog=org.apache.hadoop.http.HttpRequestLogAppender
#log4j.appender.namenoderequestlog.Filename=${hadoop.log.dir}/jetty-namenode-yyyy_mm_dd.log
#log4j.appender.namenoderequestlog.RetainDays=3

#log4j.logger.http.requests.datanode=INFO,datanoderequestlog
#log4j.appender.datanoderequestlog=org.apache.hadoop.http.HttpRequestLogAppender
#log4j.appender.datanoderequestlog.Filename=${hadoop.log.dir}/jetty-datanode-yyyy_mm_dd.log
#log4j.appender.datanoderequestlog.RetainDays=3

#log4j.logger.http.requests.resourcemanager=INFO,resourcemanagerrequestlog
#log4j.appender.resourcemanagerrequestlog=org.apache.hadoop.http.HttpRequestLogAppender
#log4j.appender.resourcemanagerrequestlog.Filename=${hadoop.log.dir}/jetty-resourcemanager-yyyy_mm_dd.log
#log4j.appender.resourcemanagerrequestlog.RetainDays=3

#log4j.logger.http.requests.jobhistory=INFO,jobhistoryrequestlog
#log4j.appender.jobhistoryrequestlog=org.apache.hadoop.http.HttpRequestLogAppender
#log4j.appender.jobhistoryrequestlog.Filename=${hadoop.log.dir}/jetty-jobhistory-yyyy_mm_dd.log
#log4j.appender.jobhistoryrequestlog.RetainDays=3

#log4j.logger.http.requests.nodemanager=INFO,nodemanagerrequestlog
#log4j.appender.nodemanagerrequestlog=org.apache.hadoop.http.HttpRequestLogAppender
#log4j.appender.nodemanagerrequestlog.Filename=${hadoop.log.dir}/jetty-nodemanager-yyyy_mm_dd.log
#log4j.appender.nodemanagerrequestlog.RetainDays=3


# WebHdfs request log on datanodes
# Specify -Ddatanode.webhdfs.logger=INFO,HTTPDRFA on datanode startup to
# direct the log to a separate file.
#datanode.webhdfs.logger=INFO,console
#log4j.logger.datanode.webhdfs=${datanode.webhdfs.logger}
#log4j.appender.HTTPDRFA=org.apache.log4j.DailyRollingFileAppender
#log4j.appender.HTTPDRFA.File=${hadoop.log.dir}/hadoop-datanode-webhdfs.log
#log4j.appender.HTTPDRFA.layout=org.apache.log4j.PatternLayout
#log4j.appender.HTTPDRFA.layout.ConversionPattern=%d{ISO8601} %m%n
#log4j.appender.HTTPDRFA.DatePattern=.yyyy-MM-dd


# Appender for viewing information for errors and warnings
yarn.ewma.cleanupInterval=300
yarn.ewma.messageAgeLimitSeconds=86400
yarn.ewma.maxUniqueMessages=250
log4j.appender.EWMA=org.apache.hadoop.yarn.util.Log4jWarningErrorMetricsAppender
log4j.appender.EWMA.cleanupInterval=${yarn.ewma.cleanupInterval}
log4j.appender.EWMA.messageAgeLimitSeconds=${yarn.ewma.messageAgeLimitSeconds}
log4j.appender.EWMA.maxUniqueMessages=${yarn.ewma.maxUniqueMessages}

#
# Fair scheduler state dump
#
# Use following logger to dump the state to a separate file

#log4j.logger.org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler.statedump=DEBUG,FSSTATEDUMP
#log4j.additivity.org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler.statedump=false
#log4j.appender.FSSTATEDUMP=org.apache.log4j.RollingFileAppender
#log4j.appender.FSSTATEDUMP.File=${hadoop.log.dir}/fairscheduler-statedump.log
#log4j.appender.FSSTATEDUMP.layout=org.apache.log4j.PatternLayout
#log4j.appender.FSSTATEDUMP.layout.ConversionPattern=%d{ISO8601} %p %c: %m%n
#log4j.appender.FSSTATEDUMP.MaxFileSize=${hadoop.log.maxfilesize}
#log4j.appender.FSSTATEDUMP.MaxBackupIndex=${hadoop.log.maxbackupindex}

# Log levels of third-party libraries
log4j.logger.org.apache.commons.beanutils=WARN
```

Конфигурация *log4j.properties* начинается с параметров по умолчанию для root logger. Он логирует все сообщения всех сервисов, но каждый сервис также имеет собственный логер. Если какие-то значения для сервисных логеров не заданы, они будут унаследованы от root logger.

Пороговое значение логирования (параметр `log4j.threshold`) эквивалентно уровню логирования: оно определяет, какой тип сообщений попадает в логи.

ПРИМЕЧАНИЕ

Пороговое значение логирования влияет на подробность сообщений в выводе некоторых команд.

Настройки appender описывает то, куда выводится информация от logger и как она хранится. В HDFS логи хранятся в формате текстовых файлов.

Более подробно об архитектуре Log4j читайте в документации Apache Logging Services.

Чтобы изменить *log4j.properties* на всех хостах одновременно, используйте ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на HDFS.
3. Включите опцию ***Show Advanced*** и найдите поле ***Custom log4j.properties***.
4. Внесите необходимые изменения в файл по умолчанию.
5. Подтвердите изменения в конфигурации HDFS, нажав ***Save***.
6. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

ПРИМЕЧАНИЕ

Файл *log4j.properties* содержит значения по умолчанию для параметров логирования. Они могут быть перезаписаны компонентами HDFS при перезапуске. Чтобы этого избежать, можно установить значения параметров через переменные среды в файле *hadoop-env.sh*.

Уровни логирования

Hadoop поддерживает следующие уровни логирования (от наименее информативного к наиболее информативному):

1. `FATAL` — сообщает, что операция не может быть выполнена и будет завершена.
2. `ERROR` — уведомляет, что программа работает неправильно или прекратила работу.
3. `WARN` — предупреждает о потенциальных проблемах. Это означает, что программа работает не по стандартному сценарию и в будущем могут возникнуть проблемы.
4. `INFO` — передает информацию о жизненном цикле или состоянии программы.
5. `DEBUG` — выводит отладочную информацию о внутренних состояниях программы.
6. `TRACE` — выводит сообщения, отслеживающие ход выполнения программы.

Logger Log4j также принимает значения уровня логирования: `OFF` — для отключения логирования, и `ALL` — для разрешения всех типов сообщений.

Включение одного уровня логирования включит этот уровень и все уровни до него. Например, если установлен уровень логирования `WARN`, то в логи будут попадать только `WARN`, `ERROR` и `FATAL`, но не `INFO`, `DEBUG` и `TRACE`.

Чтобы узнать текущий уровень логирования для компонента или установить другой, используйте команду daemonlog.

Например, чтобы установить уровень логирования `WARN` для NameNode, выполните:

```
$ hadoop daemonlog -setlevel 127.0.0.1:9870 org.apache.hadoop.hdfs.server.namenode.NameNode WARN
```

Установка уровня логирования таким способом будет действовать до перезапуска сервиса.

Архивирование логов

Некоторые сервисы Hadoop, такие как YARN и Spark, хранят свои логи в HDFS. Чтобы уменьшить количество мелких файлов в системе, можно архивировать логи с помощью архивов Hadoop. Такие логи все еще могут быть прочитаны компонентом Job History Server и командой yarn logs.

Чтобы заархивировать логи YARN, используйте команду archive-logs.

Подробнее об архивах Hadoop читайте в статье Hadoop Archive Logs Guide.

Использование Observer NameNode

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
- Запуск Observer NameNode
  - Обновление конфигурации HDFS
  - Перевод NameNode в состояние Observer
  - Проверка состояния Observer NameNode

Обзор

Традиционно пара NameNode (Active и Standby) управляют всеми клиентскими взаимодействиями: Active NameNode обрабатывает все операции с пространством имен и запросы метаданных, а Standby остается синхронизированным, но не обслуживает клиентов. В кластере ADH с высокой доступностью (High Availability, HA) добавление роли Observer NameNode решает задачи масштабируемости чтения и обеспечения согласованности при чтении метаданных.

В больших кластерах с высокой нагрузкой на метаданные (например, при частых операциях просмотра каталогов, получении расположения блоков, параллельных чтениях) Active NameNode часто становится узким местом производительности. Роль Observer NameNode реализована для переноса на нее операций чтения (read-only), что позволяет перераспределять поток запросов на чтение с Active NameNode на другие.

При этом сохраняется согласованность данных за счет считывания edits-файлов, а любые случайные запросы на запись отклоняются или перенаправляются на активную NameNode. Такая архитектура обеспечивает согласованность чтения, более равномерное распределение нагрузки и уменьшение задержек при большом объеме операций с метаданными.

Использование роли Observer позволяет кластерам HDFS обрабатывать большие объемы запросов на чтение метаданных (например, от BI-инструментов, при просмотре каталогов или через веб-интерфейсы) без нагрузки на Active NameNode, повышая производительность и отказоустойчивость.

Запуск Observer NameNode

Переведите одну из Standby NameNode в состояние Observer, следуя приведенным ниже инструкциям.

ВАЖНО

Перед назначением роли Observer одной из NameNode убедитесь, что в кластере ADH предварительно развернуты как минимум три NameNode. Это необходимо для обеспечения работоспособности кластера в случае сбоев с Active NameNode.

Обновление конфигурации HDFS

Чтобы начать использовать Observer NameNode в ADH, обновите конфигурацию HDFS через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите ***HDFS***.
3. В разделе ***hdfs-site.xml*** установите параметры ***dfs.namenode.state.context.enabled*** и ***dfs.ha.tail-edits.in-progress*** в значение `true`.
4. Подтвердите изменения, нажав ***Save***.
5. В выпадающем меню ***Actions*** выберите ***Restart***, убедитесь, что опция ***Apply configs from ADCM*** установлена в `true`, и нажмите ***Run***.

Перевод NameNode в состояние Observer

Чтобы перевести Standby NameNode в состояние Observer:

1. На хосте с NameNode проверьте ее текущее состояние с помощью команды:

```
$ hdfs haadmin -getAllServiceState
```

Пример вывода:

```
elenas-adh-4-2.ru-central1.internal:8020 active
elenas-adh-4-1.ru-central1.internal:8020 standby
elenas-adh-4-3.ru-central1.internal:8020 standby
```

2. Выполните следующую команду, чтобы перевести NameNode в состояние Observer:

```
$ sudo -u hdfs hdfs haadmin -transitionToObserver <service_id> --forcemanual
```

где `<service_id>` — это идентификатор Standby NameNode. Уточнить идентификатор можно в интерфейсе HDFS.

Пример вывода:

```
2025-10-21 08:30:33,203 WARN ha.HAAdmin: Proceeding with manual HA state management even though
automatic failover is enabled for NameNode at elenas-adh-4-1.ru-central1.internal/10.92.43.218:8020
```

Проверка состояния Observer NameNode

Чтобы проверить, изменилось ли состояние Standby NameNode, выполните:

```
$ hdfs haadmin -getAllServiceState
```

Пример вывода:

```
elenas-adh-4-2.ru-central1.internal:8020 active
elenas-adh-4-1.ru-central1.internal:8020 observer
elenas-adh-4-3.ru-central1.internal:8020 standby
```

РЕКОМЕНДАЦИЯ

Вы также можете проверить состояние NameNode через HDFS UI.

Управление сервисом HDFS через ADCM

Елена Костюченко

Содержание

Назад к содержанию

Обзор

Balancer

Disk balancer

Mover

Internal nameservice

Обзор

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом HDFS. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса HDFS, перечислены в таблице ниже.

Действие	Описание
Start balancer	Запускает HDFS Balancer
Stop balancer	Останавливает HDFS Balancer
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете где вы можете добавить, удалить или перераспределить компоненты HDFS
DataNode Maintenance/Decommission	Позволяет переводить DataNode в режим обслуживания (maintenance), выводить их из эксплуатации (decommission) или восстанавливать работу DataNode. Данные выведенного из эксплуатации DataNode будут реплицированы на другие DataNode. Этот режим можно использовать для безопасного удаления DataNode или его отключения для долгосрочного обслуживания. Для краткосрочного вывода из эксплуатации используйте режим обслуживания. В режиме обслуживания DataNode не принимает изменения и не реплицирует и не удаляет блоки
Check disk balancer	Получает текущий статус балансировщика дисков на указанных DataNode. Чтобы просмотреть отчет, перейдите на страницу ***Jobs***
Report disk balancer	Сообщает информацию о занятом дисковом пространстве из указанных DataNode. Чтобы просмотреть отчет, перейдите на страницу ***Jobs***
Start disk balancer	Запускает Disk balancer
Stop disk balancer	Останавливает Disk balancer
Start mover	Запускает Mover. При выборе этого действия укажите директории, для которых нужно проверить выполнение политик хранения данных
Stop mover	Останавливает Mover
Change internal nameservices	Позволяет изменить internal nameservices. Используйте только латинские буквы и цифры
Manage Ranger plugin	Включает или отключает плагин Ranger для HDFS
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются. Сервис поддерживает функцию ***Rolling restart***, которая позволяет перезапускать компоненты сервиса по одному, а не все сразу (компоненты DataNode могут перезапускаться партиями по несколько компонентов за раз). Это помогает избежать простоя сервиса и позволяет сохранить работоспособность кластера при рестарте. Опция ***Rolling restart*** имеет следующие параметры: ▪ ***batch_size*** — количество компонентов DataNode, которые необходимо перезапустить за одну итерацию. Рестарт компонентов партиями эффективен только в случае, если настроен rack awareness, а минимальное количество стоек равно $\mathrm{replication\_factor} - 1$. Рекомендуется устанавливать размер партии не больше количества хостов в стойке, чтобы одновременно перезапускались хосты только одной стойки, а остальные оставались активными. Если функция rack awareness не настроена, рекомендуется использовать размер партии равный `1`. ▪ ***batch_delay*** — задержка в секундах между перезапусками партий компонентов. ▪ ***health_checks*** — указывает, нужно ли выполнять проверку компонентов после рестарта. ▪ ***max_failed_batches_number*** — максимально допустимое количество партий с ошибками при рестарте. При достижении этого значения ADCM-действие ***Restart*** завершается с ошибкой. Действие ***Rolling restart*** перезапускает компоненты HDFS в следующем порядке: 1. HDFS DataNode 2. HDFS JournalNode 3. HDFS ZKFC и Standby NameNode 4. HDFS ZKFC и Active NameNode 5. HDFS HttpFS

Действия с компонентами HDFS

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Balancer

Balancer позволяет управлять нагрузкой DataNodes в кластере. Он может помочь при неравномерном распределении данных между DataNode, например, если был создан новый узел DataNode. Balancer останавливается, когда нагрузка узлов достигает допустимого порога (Threshold) или ниже.

Порог указывается в процентах от дискового пространства и отражает то, насколько нагрузка конкретного DataNode может отличаться от нагрузки всего кластера.

После выбора действия ***Start balancer*** в появившемся окне заполните следующие поля (или оставьте их пустыми, чтобы использовать значения по умолчанию):

- ***Threshold*** — процентное значение от 1 до 100. Значение по умолчанию — 10%. Меньшие значения сделают кластер более сбалансированным, но балансировка займет больше времени. Если значение слишком мало и нагрузка узлов постоянно меняется, кластер может не достичь сбалансированного состояния.
- ***Hosts to exclude*** — FDQN хостов, узлы которых должны игнорироваться балансировщиком.
- ***Hosts to include*** — FDQN хостов, узлы которых должны быть включены в процесс балансировки. По умолчанию включены все хосты.
- ***Source hosts*** — FDQN хостов, узлы которых требуют балансировки в первую очередь. Балансировщик будет перемещать блоки только из указанных узлов данных. По умолчанию используются все хосты.
- ***Idle iterations*** — количество итераций, в течение которых балансировщик может оставаться бездействующим, прежде чем он остановится. Значение по умолчанию — 5.

Вы можете запустить Balancer с дополнительными параметрами, используя CLI-команду balancer или изменив его параметры в файле конфигурации hdfs-site.xml.

Disk balancer

Disk balancer позволяет равномерно распределять нагрузку по дискам внутри одного узла. Вы можете добавить дополнительные диски в параметре `dfs.datanode.data.dir`.

Вы можете запустить балансировщик дисков с дополнительными параметрами, используя CLI-команду disk balancer или изменив его параметры в файле конфигурации hdfs-site.xml.

Mover

Mover — это инструмент переноса данных, который проверяет, соответствуют ли данные в указанном каталоге политике хранения, и, если это не так, перемещает реплики в другое хранилище, чтобы выполнить требование политики хранения.

После изменения политики хранения данных она не применяется автоматически. Используйте действие перемещения, чтобы обеспечить выполнение новой политики хранения данных.

Вы можете запустить Mover с дополнительными параметрами, используя CLI-команду mover или изменив его параметры в файле конфигурации hdfs-site.xml.

Internal nameservice

Internal nameservice — это дополнительное (внутреннее) имя для кластера HDFS, позволяющее запрашивать другой кластер HDFS из текущего. Например, для передачи данных между кластерами или создания задач.

Вы можете запрашивать любой Internal nameservice, указанный в параметре `dfs.internal.nameservices` файла конфигурации hdfs-site.xml. Узлы этого кластера будут отчитываться перед всеми службами имен в этом списке.

Использование Beeline shell с Hive

Константин Алпашкин, Михаил Серов

Содержание

Назад к содержанию

- Примеры
  - Базовое использование Beeline
  - Beeline с соединением NoSASL
- Форматы вывода Beeline
- Отмена запроса
- Фоновый запрос в скрипте из терминала

HiveServer2 поддерживает Beeline shell. Это клиент JDBC, который основан на командном интерфейсе SQLLine. Документация SQLLine также применима и к Beeline.

Вы можете найти информацию о замене реализации командного интерфейса Hive на Beeline и о причинах такой замены в документации Apache Hive.

Beeline shell может работать как во встроенном (embedded), так и в удалённом (remote) режиме. Во встроенном режиме запускается встроенный Hive (аналогичный командному интерфейсу Hive), а в удалённом режиме Beeline подключается к отдельному процессу HiveServer2 через Thrift. Начиная с Hive 0.14, при использовании Beeline с HiveServer2, лог-сообщения HiveServer2 для выполняющихся запросов также выводятся в стандартный поток ошибок (STDERR). Удалённый режим рекомендуется для использования в production-среде, так как он более безопасен и не требует предоставления пользователям прямого доступа к HDFS/metastore.

В удалённом режиме HiveServer2 принимает только валидные запросы Thrift — даже при использовании HTTP тело сообщения должно содержать информацию (payloads) от Thrift.

Примеры

Базовое использование Beeline

Данный раздел показывает базовые операции с Beeline.

Чтобы открыть Beeline shell, выполните команду:

```
$ bin/beeline
```

Пример вывода команды можно увидеть ниже.

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Beeline version 3.1.1 by Apache Hive
```

Чтобы присоединиться к Hive, выполните следующую команду в Beeline shell:

```
!connect jdbc:hive2://localhost:10000 scott tiger
```

Пример вывода команды можно увидеть ниже.

```
Connecting to jdbc:hive2://localhost:10000
Connected to: Apache Hive (version 3.1.1)
Driver: Hive JDBC (version 3.1.1)
Transaction isolation: TRANSACTION_REPEATABLE_READ
```

ПРИМЕЧАНИЕ

Если Hive работает в режиме высокой доступности, необходимо использовать специальный формат JDBC-строки.

Чтобы увидеть все Hive-таблицы, выполните следующую команду в Beeline shell:

```
show tables;
```

Данная команда выводит следующее:

```
+--------------------+
|     tab_name       |
+--------------------+
| primitives         |
| src                |
| src1               |
| src_json           |
| src_sequencefile   |
| src_thrift         |
| srcbucket          |
| srcbucket2         |
| srcpart            |
+--------------------+
9 rows selected (1.079 seconds)
```

Вы также можете указать параметры соединения в командной строке. Это значит, что можно найти команду со строкой соединения в истории вашей оболочки UNIX.

```
$ beeline -u jdbc:hive2://localhost:10000/default -n scott -w password_file
```

Beeline с соединением NoSASL

Если вы хотите установить соединение, используя NoSASL, нужно указать способ аутентификации в явном виде:

```
!connect jdbc:hive2://<host>:<port>/<db>;auth=noSasl hiveuser pass
```

Информация о доступных командах Beeline доступна на странице Командная строка Beeline.

Форматы вывода Beeline

В Beeline результат может отображаться в разных форматах. Формат устанавливается с помощью параметра `outputformat`.

Типы форматов вывода

Тип	Описание
table	Результат отображается в виде таблицы. Строка результата является строкой таблицы, а значения в одной строке отображаются в разных колонках таблицы. Данный формат выбран по умолчанию
vertical	Каждая строка результата отображается в виде блока пар ключ/значение, где ключи являются названиями колонок
xmlattr	Результат отображается в формате XML, где каждая строка представлена XML-элементом `result`. Значения строки отображаются в виде атрибутов для элемента `result`. Названия атрибутов являются названиями колонок
xmlelements	Результат отображается в формате XML, где каждая строка представлена XML-элементом `result`. Значения строки отображаются в виде дочерних элементов элемента `result`
json	(Hive 4.0) Результат отображается в формате JSON, где каждая строка представлена элементом `result` в JSON-массиве `resultset`
jsonfile	(Hive 4.0) Результат отображается в формате JSON, где каждая строка является отдельным JSON-объектом. Это соответствует таблицам, созданным в формате JSONFILE
sv output formats	Результат отображается в виде строк с использованием разделителей (delimiters). Доступны пять форматов с разделителями: csv, tsv, csv2, tsv2 и dsv

Отмена запроса

При нажатии `CTRL+C` в Beeline shell во время выполняющегося запроса, Beeline пытается отменить запрос, закрывая сокет к HiveServer2. Это поведение Beeline доступно только если `hive.server2.close.session.on.disconnect` присвоено значение `true`. Начиная с Hive 2.2.0 Beeline shell не закрывается при отмене выполняемого запроса после нажатия `CTRL+C`. Если вы хотите закрыть Beeline shell, нажмите `CTRL+C` еще раз, пока запрос все еще отменяется. Однако, если нет выполняющегося запроса, то первый ввод `CTRL+C` повлечёт выход из Beeline shell. Это поведение аналогично командной строке Hive при обработке `CTRL+C`.

РЕКОМЕНДАЦИЯ

Для выхода из Beeline shell рекомендуется использовать команду `!quit`.

Фоновый запрос в скрипте из терминала

Вы можете запускать Beeline без терминала для групповой обработки (batch processing) и автоматизации скриптов. Используйте команды `nohup` и `disown`.

Некоторые версии клиента Beeline могут требовать дополнительной настройки, чтобы команда `nohup` могла корректно переводить процесс Beeline в фоновый режим, не останавливая его.

Следующие переменные окружения могут быть изменены:

```
export HADOOP_CLIENT_OPTS="$HADOOP_CLIENT_OPTS -Djline.terminal=jline.UnsupportedTerminal"
```

Запуск с nohangup (`nohup`) и амперсандом (`&`) делает процесс фоновым и позволяет терминалу отсоединиться, сохраняя процесс Beeline запущенным.

```
nohup beeline --silent=true --showHeader=true --outputformat=dsv -f query.hql </dev/null > /tmp/output.log 2> /tmp/error.log &
```

Подключение к Hive через JDBC

Константин Алпашкин

Содержание

Назад к содержанию

Строка JDBC-подключения

Соотношение типов данных JDBC

Примеры

- Java
- Подключение с помощью DBeaver

В данной статье показаны способы подключения к HiveServer2 с использованием интерфейса JDBC, который является рекомендуемым методом клиентского взаимодействия с Hive. В статье представлены примеры подключения с помощью простой Java-программы, а также с использованием DBeaver — средства работы с базами данных, которое использует JDBC.

Для подключения к Hive в керберизированном кластере необходимо выполнить действия Kerberos, такие как установка клиента Kerberos, получение тикета, настройка *krb5.conf* и так далее. Больше подробной информации доступно в статье Обзор Kerberos.

Строка JDBC-подключения

Ниже доступны примеры строк JDBC-подключения в зависимости от используемых в ADH-кластере механизмов безопасности. Актуальная для вашего ADH-кластера строка JDBC-подключения доступна на странице Hive ***Info*** в ADCM (***Clusters*** → ***Services*** → ***Hive*** → ***Info***).

Без SSL/Kerberos

Устанавливает соединение в режиме высокой доступности, соединение не защищено:

```
jdbc:hive2://<cluster_host_0>:2181,<cluster_host_1>:2181,
<cluster_host_N>:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/<cluster_id>/<namespace>
```

Режим HTTP

Устанавливает соединение в HTTP-режиме, соединение не защищено:

```
jdbc:hive2://<cluster_host_0>:2181,<cluster_host_1>:2181,<cluster_host_N>:2181/;transportMode=http;httpPath=<hive_endpoint>
```

Где `<hive_endpoint>` — это HTTP endpoint, установленный параметром `hive.server2.thrift.http.path` в *hive-site.xml*.

SSL

Устанавливает соединение в режиме высокой доступности, соединение защищено с помощью SSL:

```
jdbc:hive2://<cluster_host_0>:2181,<cluster_host_1>:2181,
<cluster_host_N>:2181;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/<cluster_id>/<namespace>;ssl=true;ss
lTrustStore=/tmp/truststore.jks;trustStorePassword=bigdata
```

SSL+Kerberos

Устанавливает соединение в режиме высокой доступности, соединение защищено с помощью SSL+Kerberos:

```
jdbc:hive2://<cluster_host_0>:2181,<cluster_host_1>:2181,
<cluster_host_N>:2181;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/<cluster_id>/<namespace>;ssl=true;ss
lTrustStore=/tmp/truststore.jks;trustStorePassword=bigdata;principal=hive/_HOST@EXAMPLE.COM
```

Где `EXAMPLE.COM` — ваш Kerberos realm, например `RU-CENTRAL1.INTERNAL`.

Представленные выше примеры JDBC-подключений предназначены для соединения с HiveServer2 в режиме высокой доступности (High Availability, HA). Однако вы также можете подключиться к HiveServer2 напрямую, в non-HA режиме, используя порт сервера Thrift (по умолчанию `10000`). В таком случае строка JDBC-подключения имеет вид `jdbc:hive2://<cluster_host>:10000/`.

Соотношение типов данных JDBC

В таблице ниже описано соответствие типов данных Hive и Java.

Тип данных Hive	Тип данных Java	Примечание
TINYINT	byte	Знаковое/беззнаковое целое число (1 байт)
SMALLINT	short	Знаковое целое число (2 байта)
INT	int	Знаковое целое число (4 байта)
BIGINT	long	Знаковое целое число (8 байт)
FLOAT	double	Число одинарной точности
DOUBLE	double	Число двойной точности
DECIMAL	java.math.BigDecimal	Число с фиксированной точностью
BOOLEAN	boolean	1 бит (0 или 1)
STRING	String	Последовательность символов произвольной длины
TIMESTAMP	java.sql.Timestamp	Значения даты/времени
BINARY	String	Бинарные данные
ARRAY	String (json-encoded)	Значения одного типа
MAP	String (json-encoded)	Пары ключ/значение
STRUCT	String (json-encoded)	Структурированные данные

Примеры

Java

Следующий Java-код реализует подключение к HiveServer2 с помощью стандартных средств JDBC. В программе используется класс драйвера `org.apache.hive.jdbc.HiveDriver` из *hive-jdbc-3.1.1-arenadata-standalone.jar*, который можно загрузить из Maven-репозитория. Данный standalone JAR содержит все зависимости, необходимые для использования драйвера на хостах за пределами кластера ADH.

```java
import java.sql.SQLException;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.DriverManager;

public class HiveJdbcClient {
  private static String driverName = "org.apache.hive.jdbc.HiveDriver"; ①

  public static void main(String[] args) throws SQLException {
      try {
      Class.forName(driverName);
    } catch (ClassNotFoundException e) {
      e.printStackTrace();
      System.exit(1);
    }

    Connection con = DriverManager.getConnection("jdbc:hive2://ka-adh-1.ru-central1.internal:2181," +
"ka-adh-3.ru-central1.internal:2181,ka-adh-2.ru-central1.internal:2181/;serviceDiscoveryMode=zooKeeper;" +
"zooKeeperNamespace=arenadata/cluster/2/hiveserver2", "hive", ""); ②

    // Connection con = DriverManager.getConnection("jdbc:hive2://ka-adh-1.ru-central1.internal:10000/test_db", "hive", ""); ③

    Statement stmt = con.createStatement();
    String tableName = "testHiveDriverTable";
    stmt.execute("drop table if exists " + tableName);
    stmt.execute("create table " + tableName + " (key int, value string)");
    // show tables
    String sql = "show tables '" + tableName + "'";
    System.out.println("Running query: " + sql);
    ResultSet res = stmt.executeQuery(sql);
    while (res.next()) {
      System.out.println(res.getString(1));
    }

    // other Hive queries
  }
}
```

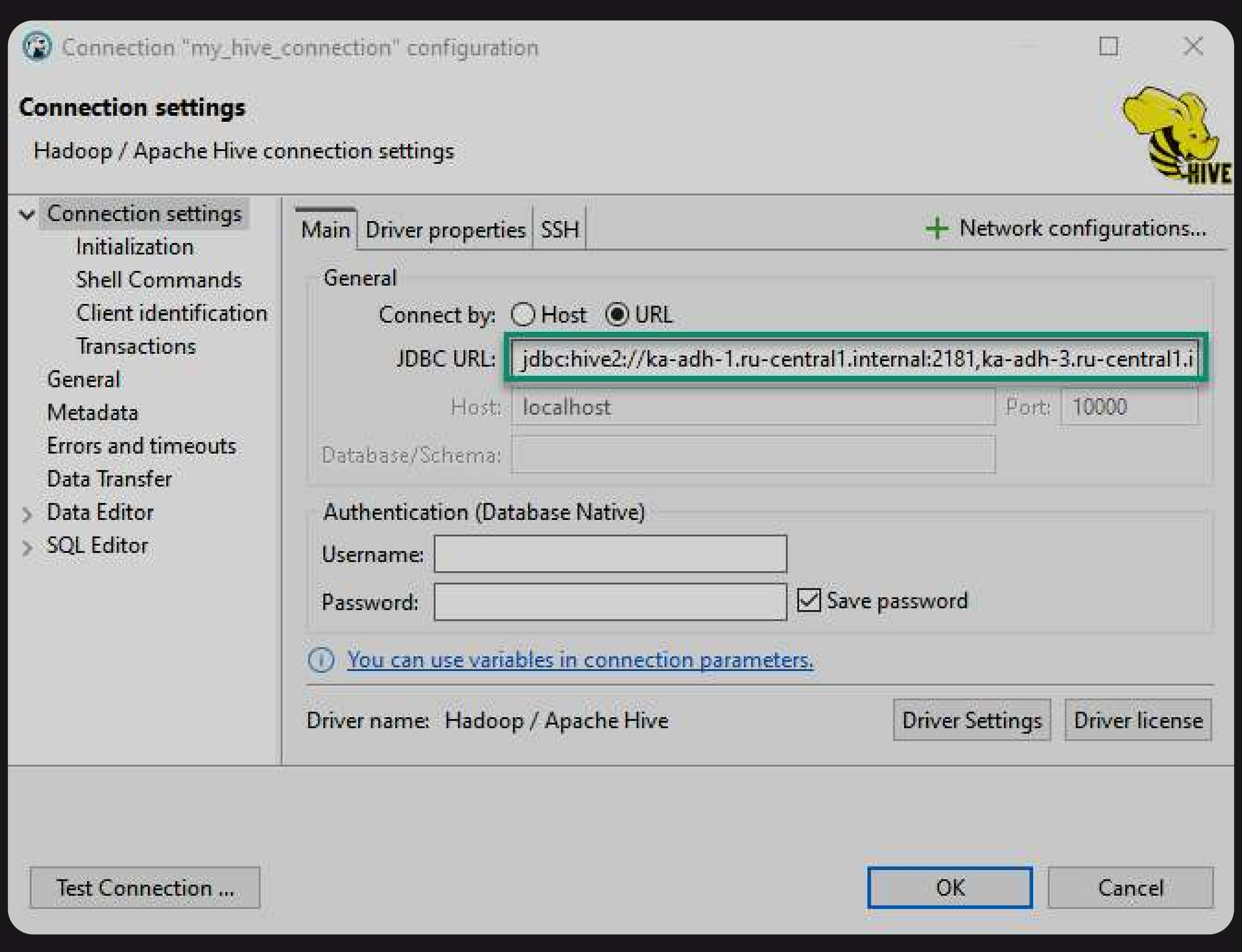
① Имя класса драйвера.

② Строка подключения JDBC для соединения с Hive в режиме высокой доступности.

③ Строка подключения JDBC для соединения с Hive в non-HA режиме, через порт Thrift-сервера.

Подключение с помощью DBeaver

DBeaver — это инструмент для работы с базами данных с открытым исходным кодом, который использует интерфейс JDBC для подключения к Hive. Для подключения к HiveServer2 из DBeaver достаточно указать соответствующую строку подключения JDBC, и Hive автоматически загрузит драйвер по умолчанию `org.apache.hive.jdbc.HiveDriver`.


Connection "my_hive_connection" configuration

Connection settings

Hadoop / Apache Hive connection settings

Connect by: ○ Host ● URL

JDBC URL: jdbc:hive2://ka-adh-1.ru-central1.internal:2181,ka-adh-3.ru-central1.i

Host: localhost

Port: 10000

Database/Schema:

Username:

Password:

☑ Save password

You can use variables in connection parameters.

Driver name: Hadoop / Apache Hive

Настройки подключения к Hive

Обзор HiveServer2 UI

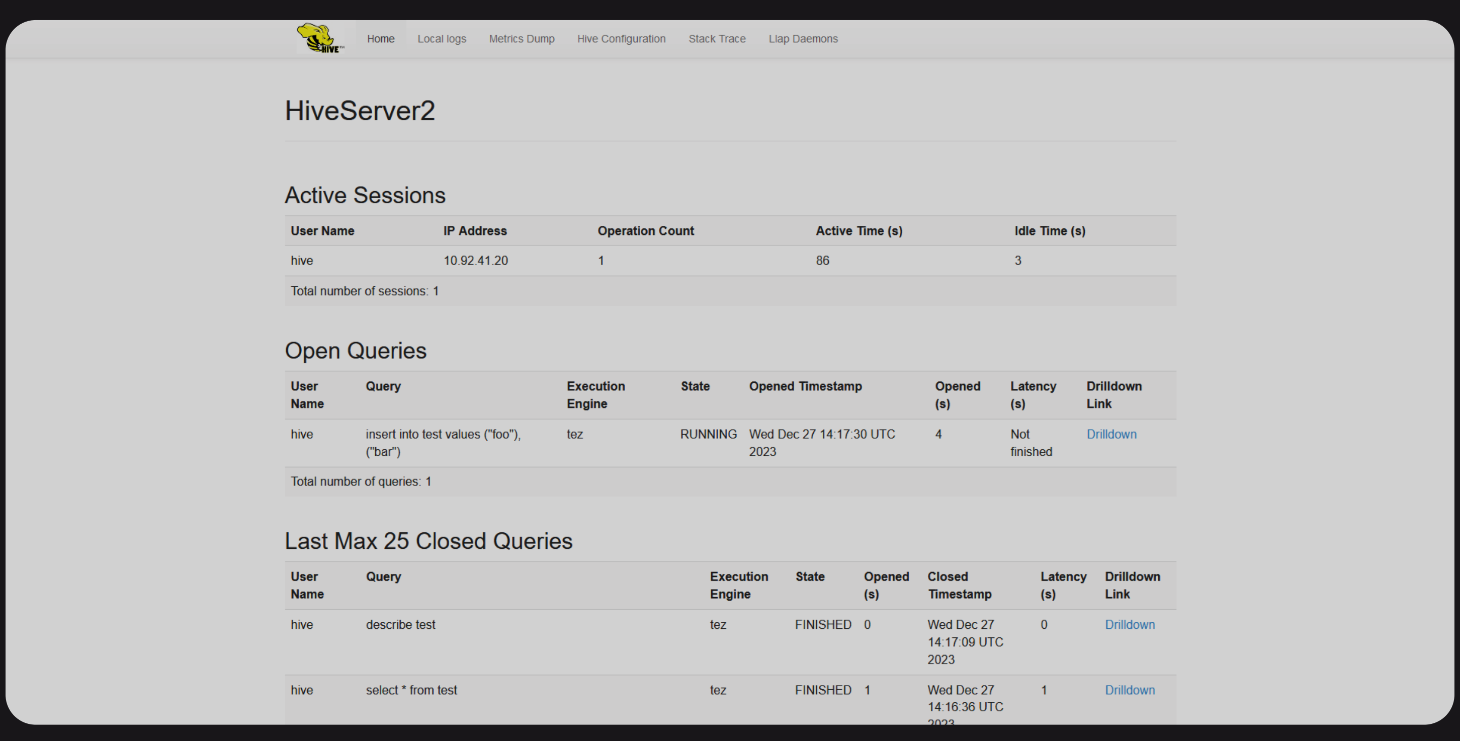
Константин Алпашкин

Содержание

Назад к содержанию

- Home
- Local logs
- Metrics Dump
- Hive Configuration
- Stack Trace
- Llap Daemons

HiveServer2 предоставляет веб-интерфейс, где вы можете просматривать конфигурацию Hive, логи, метрики, а также информацию об активных сессиях. Актуальная ссылка на веб-интерфейс доступна в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Hive*** → ***Info***). Веб-интерфейс HiveServer2 имеет следующий вид.



HiveServer2

Active Sessions

User Name	IP Address	Operation Count	Active Time (s)	Idle Time (s)
hive	10.92.41.20	1	86	3
Total number of sessions: 1				

Open Queries

User Name	Query	Execution Engine	State	Opened Timestamp	Opened (s)	Latency (s)	Drilldown Link
hive	insert into test values ("foo"), ("bar")	tez	RUNNING	Wed Dec 27 14:17:30 UTC 2023	4	Not finished	Drilldown
Total number of queries: 1							

Last Max 25 Closed Queries

User Name	Query	Execution Engine	State	Opened (s)	Closed Timestamp	Latency (s)	Drilldown Link
hive	describe test	tez	FINISHED	0	Wed Dec 27 14:17:09 UTC 2023	0	Drilldown
hive	select * from test	tez	FINISHED	1	Wed Dec 27 14:16:36 UTC	1	Drilldown

HiveServer2 UI

По умолчанию веб-интерфейс HiveServer2 доступен на порте `10002`. Чтобы изменить номер порта:

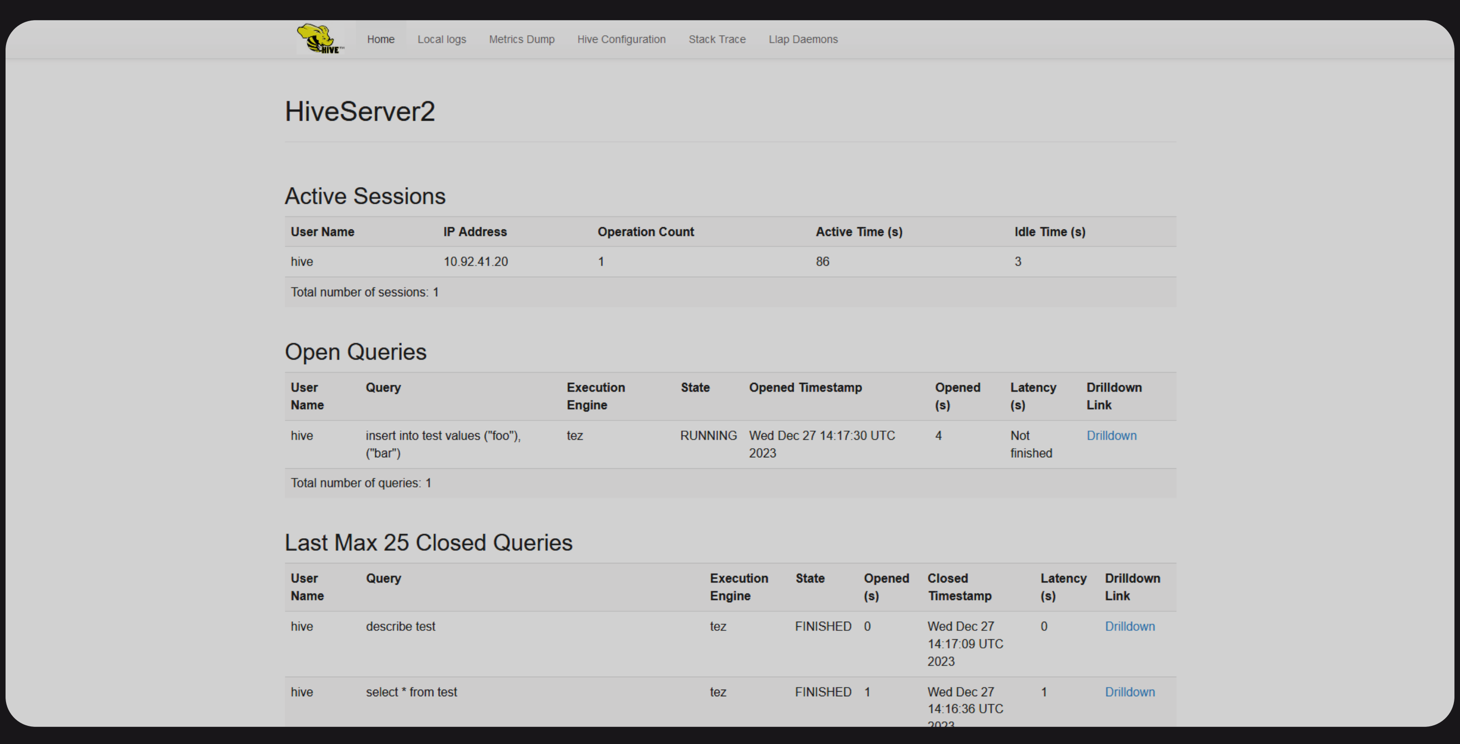
1. В ADCM перейдите на страницу ***Clusters*** → ***<clusterName>*** → ***Services*** → ***Hive*** → ***Primary configuration*** и активируйте опцию ***Show advanced***.
2. В разделе ***Custom hive-site.xml*** добавьте новый параметр `hive.server2.webui.port=<port>`.
3. Сохраните конфигурацию и выполните рестарт сервиса Hive. Веб-интерфейс станет доступным на новом порте после рестарта.

HiveServer2 UI включает следующие страницы:

- Home
- Local Logs
- Metrics Dump
- Hive Configuration
- Stack Trace
- Llap Daemons

Home

Страница ***Home*** имеет следующий вид.



HiveServer2

Active Sessions

User Name	IP Address	Operation Count	Active Time (s)	Idle Time (s)
hive	10.92.41.20	1	86	3
Total number of sessions: 1				

Open Queries

User Name	Query	Execution Engine	State	Opened Timestamp	Opened (s)	Latency (s)	Drilldown Link
hive	insert into test values ("foo"), ("bar")	tez	RUNNING	Wed Dec 27 14:17:30 UTC 2023	4	Not finished	Drilldown
Total number of queries: 1							

Last Max 25 Closed Queries

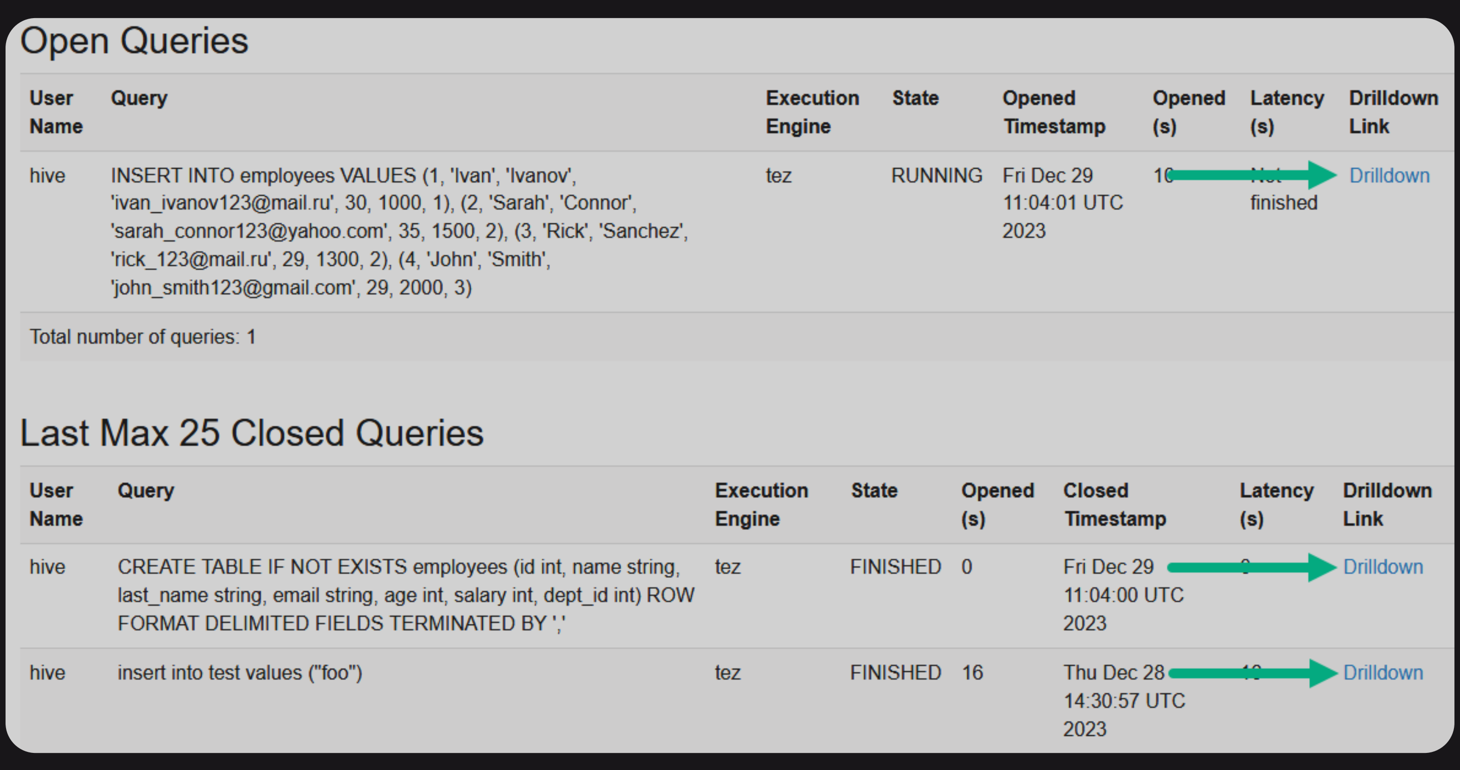
User Name	Query	Execution Engine	State	Opened (s)	Closed Timestamp	Latency (s)	Drilldown Link
hive	describe test	tez	FINISHED	0	Wed Dec 27 14:17:09 UTC 2023	0	Drilldown
hive	select * from test	tez	FINISHED	1	Wed Dec 27 14:16:36 UTC	1	Drilldown

Страница Home

На этой странице вы найдете информацию о:

- активных сессиях;
- запросах, которые в данные момент обрабатываются в Hive;
- последних запросах, отправленных на выполнение;
- Hive-атрибутах, таких как версия, номер билда, время работы и так далее.

Чтобы увидеть подробную информацию о текущих и уже выполненных Hive-запросах, используйте ссылку ***Drilldown*** на соответствующем запросе, как показано на рисунке ниже.



Open Queries

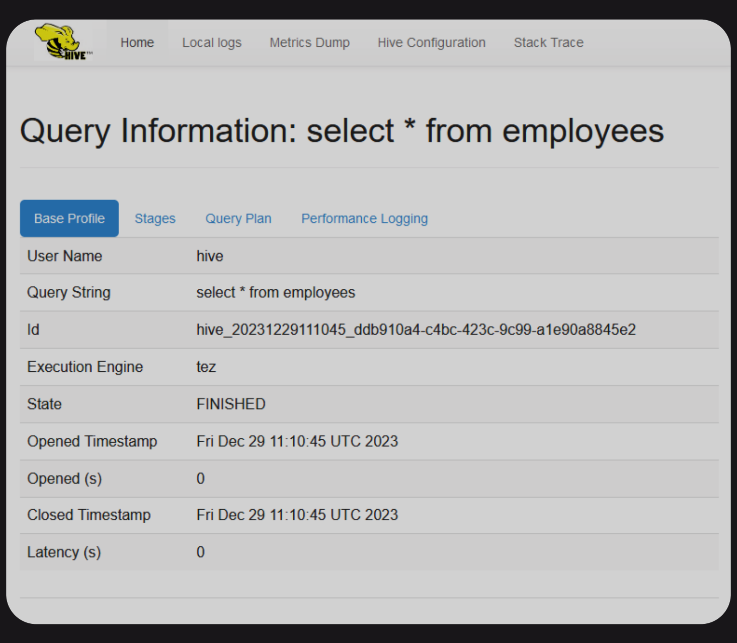
User Name	Query	Execution Engine	State	Opened Timestamp	Opened (s)	Latency (s)	Drilldown Link
hive	INSERT INTO employees VALUES (1, 'Ivan', 'Ivanov', 'ivan_ivanov123@mail.ru', 30, 1000, 1), (2, 'Sarah', 'Connor', 'sarah_connor123@yahoo.com', 35, 1500, 2), (3, 'Rick', 'Sanchez', 'rick_123@mail.ru', 29, 1300, 2), (4, 'John', 'Smith', 'john_smith123@gmail.com', 29, 2000, 3)	tez	RUNNING	Fri Dec 29 11:04:01 UTC 2023	10	Not finished	Drilldown
Total number of queries: 1							

Last Max 25 Closed Queries

User Name	Query	Execution Engine	State	Opened (s)	Closed Timestamp	Latency (s)	Drilldown Link
hive	CREATE TABLE IF NOT EXISTS employees (id int, name string, last_name string, email string, age int, salary int, dept_id int) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','	tez	FINISHED	0	Fri Dec 29 11:04:00 UTC 2023	0	Drilldown
hive	insert into test values ("foo")	tez	FINISHED	16	Thu Dec 28 14:30:57 UTC 2023	[illegible]	Drilldown

Drilldown-ссылки

Ниже показана страница ***Query Information***.



Query Information: select * from employees

User Name	hive
Query String	select * from employees
Id	hive_20231229111045_ddb910a4-c4bc-423c-9c99-a1e90a8845e2
Execution Engine	tez
State	FINISHED
Opened Timestamp	Fri Dec 29 11:10:45 UTC 2023
Opened (s)	0
Closed Timestamp	Fri Dec 29 11:10:45 UTC 2023
Latency (s)	0

Страница Query Information

Детали запроса представлены на следующих вкладках:

- ***Base Profile***. Отображает общую информацию о запросе Hive, такую как идентификатор, статус, время выполнения и так далее.
- ***Stages***. Выводит информацию о Map/Reduce-этапах запроса.
- ***Query Plan***. Отображает план выполнения запроса (execution plan, результат выполнения `EXPLAIN <query>`).
- ***Performance Logging***. Отображает метрики производительности различных Hive-логгеров.

Local logs

На этой странице вы можете просматривать логи, сгенерированные компонентами Hive Metastore/HiveServer2.

Metrics Dump

На этой странице отображаются "сырые" метрики Hive в JSON-формате.

Hive Configuration

На этой странице перечислены все параметры конфигурации из *hive-site.xml*, а также актуальные параметры Hadoop.

Stack Trace

На этой странице выводится Java-стектрейс, информирующий об активности компонентов Hive.

Llap Daemons

Отображает список демонов LLAP (если таковые имеются).

Работа с таблицами Hive

Константин Алпашкин

Содержание

Назад к содержанию

Запуск Hive

Создание таблицы

Вставка значений в таблицу

Выборка данных

Удаление таблицы

Существует несколько способов работы с таблицами Hive.

Вы можете использовать DML-запросы (Data Manipulation Language, DML) для импорта или добавления данных в таблицу. Также вы можете напрямую перенести таблицу в Hive, используя HDFS-команды. В примерах ниже мы используем SQL для загрузки и вставки данных в Hive-таблицу.

Запуск Hive

Перед началом работы убедитесь, что ваш Hive-сервер запущен. Оболочка Hive доступна на машине, где установлен компонент *Hive HiveServer2*.

1. Используйте веб-интерфейс ADCM, чтобы узнать имя хоста, на котором установлен компонент *Hive HiveServer2*.
2. Присоединитесь к машине с Hive-сервером по SSH.
3. Выполните команду `hive`:

```
$ sudo -u hive hive
```

Пример вывода команды представлен ниже.

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-client.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-hive.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-pig.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-thin-client.jar!/org/slf4j/impl/StaticLoggerBinder.
class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = c93e27de-24b5-4a3c-8488-925926d54d78

Logging initialized using configuration in jar:file:/usr/lib/hive/lib/hive-common-3.1.1.jar!/hive-log4j2.properties Async: true
Hive Session ID = 88e81126-ac59-497e-bac2-709a665d4cf2
hive>
```

Создание таблицы

Используйте команду `CREATE` для создания таблицы. Следующая команда создает таблицу `solar_objects`, используя символ `,` в качестве разделителя для колонок:

```
CREATE TABLE solar_objects(
    Name STRING,
    Mass INT,
    Diameter INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

Пример вывода представлен ниже:

```
OK
Time taken: 0.074 seconds
```

Вставка значений в таблицу

Используйте команду `INSERT` для наполнения таблицы данными:

```
INSERT INTO TABLE
    solar_objects
VALUES
    ('Sun',1989100000,1392000),
    ('Mercury',330,4879),
    ('Venus',4867,12104),
    ('Earth',5972,12742),
    ('Mars',642,6780),
    ('Jupiter',1898187,139822),
    ('Saturn',568317,116464),
    ('Uranus',86813,50724),
    ('Neptune',102413,49244);
```

Пример вывода представлен ниже:

```
Query ID = aai_20211116141418_4a40a2c4-1d21-422c-80ee-cf0f95f35000
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1637045461848_0005)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 5,42 s
----------------------------------------------------------------------------------------------
Loading data to table default.solar_objects
OK
Time taken: 15.12 seconds
```

Выборка данных

Используйте `SELECT` для выборки данных из таблицы:

```
SELECT * FROM solar_objects;
```

Пример вывода представлен ниже:

```
OK
Sun     1989100000      1392000
Mercury 330     4879
Venus   4867    12104
Earth   5972    12742
Mars    642     6780
Jupiter 1898187 139822
Saturn  568317  116464
Uranus  86813   50724
Neptune 102413  49244
Time taken: 0.17 seconds, Fetched: 9 row(s)
```

Удаление таблицы

Чтобы удалить таблицу, используйте команду `DROP`:

```
DROP TABLE solar_objects;
```

Пример вывода представлен ниже:

```
OK
Time taken: 1.991 seconds
```

Пример вычислений Hive

Константин Алпашкин

Содержание

Назад к содержанию

Набор данных

Подготовка

Выполнение вычислений

В данной статье мы рассмотрим обработку криминальной статистики, основанную на реальных отчетах о происшествиях Бостонского управления полиции. Набор данных предоставлен Analyze Boston. Актуальный dataset доступен на странице Kaggle.

Набор данных

Набор данных (Dataset) содержит записи с 14 июня 2015 по 3 сентября 2018 года. Каждая строка представляет запись о преступлении, включая вид преступления, дату, время и место.

Программа принимает следующие файлы на входе:

- *crime.csv* — основной файл с записями о преступлениях;
- *offense_codes.csv* — файл, содержащий коды преступлений.

В результате выполнения программы мы получим следующую информацию:

- наиболее частые виды преступлений;
- распределение преступлений по районам;
- частота совершения преступления по видам.

Подготовка

Перед началом работы выполните следующие шаги:

1. На машине, где установлен Hive-сервер, создайте директории для файлов с данными:

```
$ hdfs dfs -mkdir /user/user1/crime
$ hdfs dfs -mkdir /user/user1/offense_codes
```

2. Скопируйте файлы с данными в созданные директории:

```
$ hdfs dfs -put ~/crime.csv /user/user1/crime
$ hdfs dfs -put ~/offense_codes.csv /user/user1/offense_codes
```

Выполнение вычислений

1. Создайте таблицу `offense_codes`:

```
CREATE EXTERNAL TABLE offense_codes (
  CODE int,
  NAME string
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/user/user1/offense_codes' tblproperties ("skip.header.line.count"="1");
```

2. Создайте таблицу `crime`:

```
CREATE EXTERNAL TABLE crime (
    INCIDENT_NUMBER string,
    OFFENSE_CODE int,
    OFFENSE_CODE_GROUP string,
    OFFENSE_DESCRIPTION string,
    DISTRICT string,
    REPORTING_AREA string,
    SHOOTING string,
    OCCURRED_ON_DATE string,
    YEAR int,
    MONTH int,
    DAY_OF_WEEK string,
    HOUR int,
    UCR_PART string,
    STREET string,
    Lat DECIMAL,
    Long DECIMAL,
    Location string
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION '/user/user1/crime' tblproperties ("skip.header.line.count"="1");
```

РЕКОМЕНДАЦИЯ

Не забудьте изменить путь, где хранятся ваши таблицы и файлы с данными. В данном примере используется путь */user/user1/*.

3. Запустите вычисления с помощью следующего SQL-скрипта:

```
with filtered_crimes as (
    select *
    from crime cr
    where cr.district is not null
),
     crimes_w_codes as (
         select cr.*,
                oc.*,
                trim(case
                    when instr(oc.name, '-') != 0 then
                        substr(oc.name, 1, instr(oc.name, '-') - 1)
                    else oc.name end) as crime_type
         from filtered_crimes cr
                  join offense_codes oc
                       on cr.offense_code = oc.code
     ),
     crimes_district_analytics as (
         select cr.district, count(incident_number) as crimes_total, avg(cr.lat) as lat, avg(cr.long) as lng
         from filtered_crimes cr
         group by cr.district
     ),
     crimes_by_district_by_month as (
         select cr.district, cr.month, count(cr.incident_number) as crimes_cnt
         from filtered_crimes cr
         group by cr.district, cr.month
     ),
     crimes_district_median as (
         select cr.district, percentile(cr.crimes_cnt,0.5) as crimes_monthly
         from crimes_by_district_by_month cr
         group by cr.district
     ),
     crimes_by_district_by_crime_type as (
         select cr.district, cr.crime_type, count(cr.INCIDENT_NUMBER) as crimes_cnt
         from crimes_w_codes cr
         group by cr.district, cr.crime_type
     ),
     crimes_sort as (
         select pre.district, collect_list(pre.crime_type) as frequent_crime_types
         from (
                  select cr.district,
                         cr.crime_type,
                         row_number() over (partition by cr.district order by cr.crimes_cnt desc) as rn
                  from crimes_by_district_by_crime_type cr) pre
         where pre.rn <= 3
         group by pre.district
     )
select cda.district, cda.crimes_total, cdm.crimes_monthly, cs.frequent_crime_types, cda.lat, cda.lng
from crimes_district_analytics cda
         join crimes_district_median cdm on cda.district = cdm.district
         join crimes_sort cs on cda.district = cs.district;
```

Где:

- `crimes_total` — общее количество преступлений, совершенных в указанном районе.
- `crimes_monthly` — усредненное количество преступлений в месяц, совершенных в указанном районе.
- `frequent_crime_types` — три наиболее частых вида преступлений в указанном районе за всю историю наблюдений. Значения разделены запятой и пробелом (`, `) и упорядочены по нисходящей от наиболее частых к редким.
- `crime_type` — первая часть поля `NAME` из таблицы *offense_codes.csv*, отсеченная разделителем `-`. Например, если поле `NAME` содержит `BURGLARY - COMMERICAL - ATTEMPT`, то `crime_type` определяется как `BURGLARY`.
- `lat` — широта координат района, средняя по всем преступлениям.
- `lng` — долгота координат района, средняя по всем преступлениям.

Результат выполнения скрипта имеет следующий вид:

```
No Stats for default@crime, Columns: district, incident_number, lat, long
No Stats for default@crime, Columns: month
No Stats for default@crime, Columns: offense_code
No Stats for default@offense_codes, Columns: code, name
Query ID = user1_20211112125439_22d654c1-caeb-47f8-adc9-8e2028602653
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1636705252031_0039)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      3          3        0        0       0       0
Map 4 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 5 ...... container     SUCCEEDED      3          3        0        0       0       0
Map 10 ......... container     SUCCEEDED      1          1        0        0       0       0
Map 6 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 7 ...... container     SUCCEEDED      3          3        0        0       0       0
Reducer 8 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 9 ...... container     SUCCEEDED      3          3        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      3          3        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 10/10  [==========================>>] 100%  ELAPSED TIME: 9,21 s
----------------------------------------------------------------------------------------------
OK
cda.district    cda.crimes_total    cdm.crimes_monthly  cs.frequent_crime_types cda.lat cda.lng
 ETC"   1522    369.5   ["\"WEAPON","FIREARM/WEAPON"]   NULL    41.9364
 ETC. " 1514    382.0   ["\"DRUGS"] NULL    41.8329
A7  13151   1062.5  ["SICK/INJURED/MEDICAL","INVESTIGATE PERSON","VANDALISM"]   41.9898 -70.9834
B2  48669   3876.0  ["VERBAL DISPUTE","SICK/INJURED/MEDICAL","INVESTIGATE PERSON"]  41.9943 -70.9907
C11 41460   3212.5  ["SICK/INJURED/MEDICAL","INVESTIGATE PERSON","VERBAL DISPUTE"]  41.9925 -70.9877
D4  41020   3221.5  ["PROPERTY","INVESTIGATE PERSON","SICK/INJURED/MEDICAL"]    41.9977 -70.9963
E5  12966   1022.5  ["SICK/INJURED/MEDICAL","INVESTIGATE PERSON","PROPERTY"]    41.9178 -70.8662
 BRUSH  525 525.0   ["\"FIRE REPORT"]   NULL    NULL
 BUILDING   1269    1269.0  ["\"FIRE REPORT"]   NULL    NULL
A15 6378    491.5   ["INVESTIGATE PERSON","VANDALISM","SICK/INJURED/MEDICAL"]   41.8010 -70.6760
    1693    107.0   ["M/V ACCIDENT","M/V","INVESTIGATE PROPERTY"]   24.8485 -43.0789
 ETC."  2597    680.0   ["\"DRUGS"] NULL    41.8254
 FEET   1   1.0 ["\"A&B HANDS"] NULL    NULL
 NON-NEGLIGIENT MANSLAUGHTER"   161 40.0    ["\"MURDER","MURDER NON"]   NULL    41.7190
A1  34834   2698.5  ["PROPERTY","SICK/INJURED/MEDICAL","WARRANT ARREST"]    41.9741 -70.9578
B3  34618   2696.0  ["VERBAL DISPUTE","INVESTIGATE PERSON","MISSING PERSON"]    41.9961 -70.9936
C6  22903   1826.5  ["SICK/INJURED/MEDICAL","INVESTIGATE PERSON","TOWED MOTOR VEHICLE"] 41.8786 -70.8023
D14 19785   1581.0  ["TOWED MOTOR VEHICLE","SICK/INJURED/MEDICAL","INVESTIGATE PERSON"] 41.9931 -70.9888
E13 17010   1323.0  ["SICK/INJURED/MEDICAL","INVESTIGATE PERSON","TOWED MOTOR VEHICLE"] 41.9947 -70.9913
E18 16997   1289.5  ["SICK/INJURED/MEDICAL","INVESTIGATE PERSON","VERBAL DISPUTE"]  42.0000 -71.0000
Time taken: 12.691 seconds, Fetched: 20 row(s)
```

Многотабличная вставка данных в Hive

Константин Алпашкин

Содержание

Назад к содержанию

- Тестовая база данных
- Синтаксис
- Примеры
  - Запись в несколько таблиц Hive
  - Запись в партиции
  - Запись в несколько файлов

Hive поддерживает многотабличную вставку данных, что позволяют записывать данные в несколько таблиц Hive или в несколько HDFS-файлов одновременно. Это эффективный с точки зрения оптимизации прием, поскольку исходная таблица сканируется лишь один раз, а полученные данные используются для записи в несколько таблиц/файлов.

Тестовая база данных

Для демонстрации примеров в данной статье используется Hive-таблица `transactions`, структура которой описана ниже.

```
SELECT * FROM transactions;
+----------------------+----------------------+--------------------------+------------------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  |
+----------------------+----------------------+--------------------------+------------------------+
| 1                    | 1002                 | 10.00                    | 2023-01-01             |
| 8                    | 1003                 | 50.00                    | 2023-01-01             |
| 3                    | 1002                 | 30.00                    | 2023-01-02             |
| 4                    | 1001                 | 100.50                   | 2023-01-02             |
| 2                    | 1002                 | 20.00                    | 2023-01-03             |
| 6                    | 1001                 | 200.50                   | 2023-01-03             |
| 7                    | 1003                 | 50.00                    | 2023-01-03             |
| 5                    | 1001                 | 150.50                   | 2023-01-04             |
| 9                    | 1003                 | 75.00                    | 2023-01-04             |
+----------------------+----------------------+--------------------------+------------------------+
```

SQL для создания таблицы

Для создания и наполнения тестовой таблицы Hive выполните следующий SQL с помощью */bin/beeline*.

```
CREATE DATABASE IF NOT EXISTS test_multi_inserts;
USE test_multi_inserts;

DROP TABLE IF EXISTS transactions;

CREATE TABLE transactions(
    txn_id int,
    acc_id int,
    txn_amount decimal(10,2),
    txn_date date)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

INSERT INTO transactions VALUES
(1, 1002, 10.00, '2023-01-01'),
(2, 1002, 20.00, '2023-01-03'),
(3, 1002, 30.00, '2023-01-02'),
(4, 1001, 100.50, '2023-01-02'),
(5, 1001, 150.50, '2023-01-04'),
(6, 1001, 200.50, '2023-01-03'),
(7, 1003, 50.00, '2023-01-03'),
(8, 1003, 50.00, '2023-01-01'),
(9, 1003, 75.00, '2023-01-04');
```

Синтаксис

Существует две формы синтаксиса многотабличной вставки, оба варианта показаны ниже.

```
FROM <src_tbl>
INSERT OVERWRITE
    (TABLE <target_tbl> [PARTITION(<partition_spec>)] | DIRECTORY [LOCAL] '<path/to/hdfs>')
<SELECT ... >


INSERT OVERWRITE
    (TABLE <target_tbl> [PARTITION(<partition_spec>)] | DIRECTORY [LOCAL] '<path/to/hdfs')
<SELECT ... FROM <src_tbl>. ... >
```

Где:

- `<src_tbl>` — исходная Hive-таблица для чтения данных.
- `<SELECT … >` — предложение `SELECT` для выборки данных из `<src_tbl>`.
- `<target_tbl>` — целевая таблица, в которую будут записаны данные. Целевые таблицы не создаются автоматически и должны быть предварительно созданы вручную.
- `<partition_spec>` — указывает партицию таблицы Hive, в которую необходимо записать данные.
- `<path/to/hdfs>` — путь к HDFS-файлу, в который необходимо записать данные. Если указан относительный путь, Hive сохраняет файлы внутри HDFS-директории */user/hive/*. С помощью ключевого слова `LOCAL` можно записать данные в локальной файловой системе.

Примеры

Запись в несколько таблиц Hive

В следующем примере запрос вычисляет минимальные и максимальные суммы транзакций для каждого аккаунта и записывает результаты в соответствующие Hive-таблицы.

ПРИМЕЧАНИЕ

Целевые таблицы для записи должны быть созданы вручную перед выполнением `INSERT`.

```
CREATE TABLE max_txn_per_acc (account integer, max_txn_value decimal(10,2));
CREATE TABLE min_txn_per_acc (account integer, min_txn_value decimal(10,2));

FROM transactions
    INSERT INTO TABLE max_txn_per_acc
        SELECT acc_id, MAX(txn_amount)
        GROUP BY acc_id
    INSERT INTO TABLE min_txn_per_acc
        SELECT acc_id, MIN(txn_amount)
        GROUP BY acc_id;
```

Для проверки результатов вставки используйте запросы:

```
SELECT * FROM max_txn_per_acc;
SELECT * FROM min_txn_per_acc;
```

Результат:

```
SELECT * FROM max_txn_per_acc;
+--------------------------+--------------------------------+
| max_txn_per_acc.account  | max_txn_per_acc.max_txn_value  |
+--------------------------+--------------------------------+
| 1001                     | 200.50                         |
| 1002                     | 30.00                          |
| 1003                     | 75.00                          |
+--------------------------+--------------------------------+

SELECT * FROM min_txn_per_acc;
+--------------------------+--------------------------------+
| min_txn_per_acc.account  | min_txn_per_acc.min_txn_value  |
+--------------------------+--------------------------------+
| 1001                     | 100.50                         |
| 1002                     | 10.00                          |
| 1003                     | 50.00                          |
+--------------------------+--------------------------------++
```

Запись в партиции

Hive позволяет записывать данные из исходной таблицы в несколько партиций в рамках одного запроса. Следующий пример вставляет данные из столбцов исходной таблицы в заранее известные (статические) партиции.

```
CREATE TABLE txns_partitioned_acc_date(txn_id int, txn_amount decimal(10, 2))
PARTITIONED BY (account_id int, txn_date date);

FROM transactions txns
    INSERT OVERWRITE TABLE txns_partitioned_acc_date
    PARTITION(account_id='1001', txn_date='2023-01-02')
SELECT txn_id, txn_amount
WHERE txns.acc_id=1001 AND txns.txn_date='2023-01-02';

SELECT * FROM txns_partitioned_acc_date;
```

Результат:

```
+-----------------------------------+---------------------------------------+---------------------------------------+------------
-------------------------+
| txns_partitioned_acc_date.txn_id  | txns_partitioned_acc_date.txn_amount  | txns_partitioned_acc_date.account_id  |
txns_partitioned_acc_date.txn_date  |
+-----------------------------------+---------------------------------------+---------------------------------------+------------
-------------------------+
| 4                                 | 100.50                                | 1001                                  | 2023-01-02
|
+-----------------------------------+---------------------------------------+---------------------------------------+------------
-------------------------+
```

По умолчанию в предложении `PARTITION(<spec>)` должна быть указана минимум одна статическая партиция (`PARTITION(account_id='1001')` как в примере выше). Чтобы использовать динамические партиции при многотабличной вставке (например, `PARTITION('acount_id')`) и тем самым позволить Hive автоматически определять необходимую партицию для записи, установите следующие параметры:

```
SET hive.exec.dynamic.partition=true;
SET hive.exec.dynamic.partition.mode=nonstrict;
```

После этого вы можете использовать динамические партиции, как показано в следующем примере.

```
CREATE TABLE txns_partitioned_acc(txn_amount decimal(10, 2))
PARTITIONED BY (account_id int);

SET hive.exec.dynamic.partition.mode=nonstrict;

FROM transactions txns
    INSERT OVERWRITE TABLE txns_partitioned_acc
    PARTITION(account_id)
SELECT txn_amount, acc_id;

SELECT * FROM txns_partitioned_acc;
```

Из вывода следует, что Hive автоматически записал данные в нужную партицию.

```
+----------------------------------+----------------------------------+
| txns_partitioned_acc.txn_amount  | txns_partitioned_acc.account_id  |
+----------------------------------+----------------------------------+
| 100.50                           | 1001                             |
| 150.50                           | 1001                             |
| 200.50                           | 1001                             |
| 10.00                            | 1002                             |
| 20.00                            | 1002                             |
| 30.00                            | 1002                             |
| 50.00                            | 1003                             |
| 50.00                            | 1003                             |
| 75.00                            | 1003                             |
+----------------------------------+----------------------------------+
```

Запись в несколько файлов

Hive позволяет записывать данные из таблицы в несколько HDFS/локальных файлов в рамках одного запроса. Ниже показана запись в несколько HDFS-файлов.

```
FROM transactions
    INSERT OVERWRITE DIRECTORY 'multi-insert-demo/transactions_20230101'
    SELECT acc_id, txn_amount WHERE txn_date='2023-01-01'
    INSERT OVERWRITE DIRECTORY 'multi-insert-demo/transactions_20230102'
    SELECT acc_id, txn_amount WHERE txn_date='2023-01-02';
```

Для проверки HDFS-директории на наличие новых файлов используйте команду:

```
$ hdfs dfs -ls /user/hive/multi-insert-demo
```

В выводе отображаются две новых HDFS-директории с данными каждой выборки.

```
...
drwxr-xr-x   - hive hadoop          0 2023-11-29 10:51 /user/hive/multi-insert-demo/transactions_20230101
drwxr-xr-x   - hive hadoop          0 2023-11-29 10:51 /user/hive/multi-insert-demo/transactions_20230102
```

Для записи в файлы, расположенные в локальной файловой системе, а не в HDFS, используйте ключевое слово `LOCAL` как показано ниже.

```
FROM transactions
    INSERT OVERWRITE LOCAL DIRECTORY '/tmp/test.out'
SELECT transactions.*;
```

Анализ запросов в Hive

Константин Алпашкин

Содержание

Назад к содержанию

- Тестовая база данных
- EXPLAIN
  - EXPLAIN
  - EXPLAIN EXTENDED
  - EXPLAIN DEPENDENCY
  - EXPLAIN AUTHORIZATION
  - EXPLAIN VECTORIZATION
- Сбор статистики с помощью ANALYZE
  - Сбор статистики

Перед выполнением каждого запроса оптимизатор Hive (Cost-Based Optimizer, CBO) создает **план выполнения** — последовательность задач, которые должны быть выполнены, чтобы получить нужные данные наиболее эффективным способом. Используя команды Hive `EXPLAIN` и `ANALYZE`, вы можете просматривать планы выполнения и получать подробную статистику о работе Hive для оптимизации ваших запросов.

Тестовая база данных

Для демонстрации примеров в данной статье используются Hive-таблицы `transactions` и `accounts`, структура которых описана ниже.

```
SELECT * FROM transactions;
+----------------------+----------------------+--------------------------+-------------------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  |
+----------------------+----------------------+--------------------------+-------------------------+
| 1                    | 1002                 | 10.00                    | 2023-01-01              |
| 8                    | 1003                 | 50.00                    | 2023-01-01              |
| 3                    | 1002                 | 30.00                    | 2023-01-02              |
| 4                    | 1001                 | 100.50                   | 2023-01-02              |
| 2                    | 1002                 | 20.00                    | 2023-01-03              |
| 6                    | 1001                 | 200.50                   | 2023-01-03              |
| 7                    | 1003                 | 50.00                    | 2023-01-03              |
| 5                    | 1001                 | 150.50                   | 2023-01-04              |
| 9                    | 1003                 | 75.00                    | 2023-01-04              |
+----------------------+----------------------+--------------------------+-------------------------+

SELECT * FROM accounts;
+--------------+---------------------+
| accounts.id  | accounts.full_name  |
+--------------+---------------------+
| 1001         | John Smith          |
| 1002         | Sarah Connor        |
| 1003         | Rick Sanchez        |
+--------------+---------------------+
```

SQL для создания тестовых таблиц

Для создания и наполнения тестовых таблиц Hive выполните следующий SQL с помощью */bin/beeline*.

```sql
CREATE DATABASE IF NOT EXISTS explain_demo;
USE explain_demo;

DROP TABLE IF EXISTS transactions;
CREATE TABLE transactions(txn_id int, acc_id int, txn_amount decimal(10,2)) PARTITIONED BY (txn_date date);
INSERT INTO transactions VALUES
(1, 1002, 10.00, '2023-01-01'),
(2, 1002, 20.00, '2023-01-03'),
(3, 1002, 30.00, '2023-01-02'),
(4, 1001, 100.50, '2023-01-02'),
(5, 1001, 150.50, '2023-01-04'),
(6, 1001, 200.50, '2023-01-03'),
(7, 1003, 50.00, '2023-01-03'),
(8, 1003, 50.00, '2023-01-01'),
(9, 1003, 75.00, '2023-01-04');

DROP TABLE IF EXISTS accounts;
CREATE TABLE accounts(id int, full_name string);
INSERT INTO accounts VALUES
(1001, 'John Smith'),
(1002, 'Sarah Connor'),
(1003, 'Rick Sanchez');
```

EXPLAIN

Команда `EXPLAIN` выводит план выполнения (execution plan) Hive-запроса. Вывод включает в себя подробное описание того, какими этапами Hive планирует выполнить запрос.

Синтаксис имеет следующий вид:

```
EXPLAIN [EXTENDED|DEPENDENCY|AUTHORIZATION|VECTORIZATION|FORMATTED] <query>
```

Где:

- `EXTENDED` — возвращает расширенную информацию о плане выполнения.
- `DEPENDENCY` — содержит подробную информацию о таблицах и партициях, которые задействованы в запросе.
- `AUTHORIZATION` — возвращает информацию о всех объектах, требующих авторизации, включая проблемы с правами доступа (если таковые имеются).
- `VECTORIZATION` — возвращает информацию о тех Map/Reduce-задачах, для которых не применялась векторизация.
- `FORMATTED` — форматирует вывод в JSON-формате.
- `<query>` — запрос, который необходимо анализировать.

EXPLAIN

Команда `EXPLAIN` возвращает общую информацию о плане выполнения запроса.

```sql
CREATE TABLE IF NOT EXISTS top_txns_per_acc(acc_id int, max_txn_value decimal(10,2));

EXPLAIN
    FROM transactions INSERT OVERWRITE TABLE top_txns_per_acc
    SELECT acc_id, MAX(txn_amount) GROUP BY acc_id;
```

Результат:

```
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
| Plan optimized by CBO.                             |
|                                                    |
| Vertex dependency in root stage                    |
| Reducer 2 <- Map 1 (SIMPLE_EDGE)                   |
| Reducer 3 <- Reducer 2 (CUSTOM_SIMPLE_EDGE)        |
|                                                    |
| Stage-3                                            |
|   Stats Work{}                                     |
|     Stage-0                                        |
|       Move Operator                                |
|         table:{"name:":"explain_demo.top_txns_per_acc"} |
|         Stage-2                                    |
|           Dependency Collection{}                  |
|             Stage-1                                |
|               Reducer 3                            |
|               File Output Operator [FS_13]         |
|                 Group By Operator [GBY_11] (rows=1 width=552) |
|                   Output:["_col0","_col1"],aggregations:["compute_stats(VALUE._col0)","compute_stats(VALUE._col1)"] |
|                 <-Reducer 2 [CUSTOM_SIMPLE_EDGE]   |
|                   File Output Operator [FS_6]      |
|                     table:{"name:":"explain_demo.top_txns_per_acc"} |
|                     Group By Operator [GBY_4] (rows=4 width=12) |
|                       Output:["_col0","_col1"],aggregations:["max(VALUE._col0)"],keys:KEY._col0 |
|                     <-Map 1 [SIMPLE_EDGE] vectorized |
|                       SHUFFLE [RS_16]              |
|                         PartitionCols:_col0        |
|                         Group By Operator [GBY_15] (rows=9 width=12) |
|                           Output:["_col0","_col1"],aggregations:["max(txn_amount)"],keys:acc_id |
|                           Select Operator [SEL_14] (rows=9 width=12) |
|                             Output:["acc_id","txn_amount"] |
|                             TableScan [TS_0] (rows=9 width=12) |
|                               explain_demo@transactions,transactions,Tbl:COMPLETE,Col:NONE,Output:["acc_id","txn_amount"] |
|                   PARTITION_ONLY_SHUFFLE [RS_10]   |
|                     Group By Operator [GBY_9] (rows=1 width=536) |
|                       Output:["_col0","_col1"],aggregations:["compute_stats(acc_id, 'hll')","compute_stats(max_txn_value,
'hll')"] |
|                       Select Operator [SEL_8] (rows=4 width=12) |
|                         Output:["acc_id","max_txn_value"] |
|                          Please refer to the previous Group By Operator [GBY_4] |
|                                                    |
+----------------------------------------------------+
```

Вывод `EXPLAIN` может значительно отличаться в зависимости от запроса, но основными информационными блоками являются следующие:

- **Stage dependencies**. Содержит список этапов (stages), которые вместе формируют общий план выполнения. Отдельный этап может являться Map/Reduce-задачей, операцией чтения/записи в HDFS, обращением к Hive Metastore и так далее.
- **Stage plans**. Каждый блок описывает детали отдельного этапа, включая задействованные операторы, порядок сортировки и так далее.

EXPLAIN EXTENDED

Команда `EXPLAIN EXTENDED` выводит более подробную информацию о плане выполнения Hive. Вывод включает в себя дополнительную информацию об операциях ввода-вывода HDFS, форматах данных, бакетах, расширенную статистику и так далее.

```sql
EXPLAIN EXTENDED
    SELECT * FROM transactions
    WHERE txn_amount > 100;
```

Пример вывода EXPLAIN EXTENDED

```
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
| STAGE DEPENDENCIES:                                |
|   Stage-0 is a root stage                          |
|                                                    |
| STAGE PLANS:                                       |
|   Stage: Stage-0                                   |
|     Fetch Operator                                 |
|       limit: -1                                    |
|       Partition Description:                       |
|           Partition                                |
|             input format: org.apache.hadoop.mapred.TextInputFormat |
|             output format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|             partition values:                      |
|               txn_date 2023-01-01                  |
|             properties:                            |
|               COLUMN_STATS_ACCURATE {"BASIC_STATS":"true","COLUMN_STATS":
{"acc_id":"true","txn_amount":"true","txn_id":"true"}} |
|               bucket_count -1                      |
|               column.name.delimiter ,              |
|               columns txn_id,acc_id,txn_amount     |
|               columns.comments                     |
|               columns.types int:int:decimal(10,2)  |
|               file.inputformat org.apache.hadoop.mapred.TextInputFormat |
|               file.outputformat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|               location hdfs://adh/apps/hive/warehouse/explain_demo.db/transactions/txn_date=2023-01-01 |
|               name explain_demo.transactions       |
|               numFiles 1                           |
|               numRows 2                            |
|               partition_columns txn_date           |
|               partition_columns.types date         |
|               rawDataSize 24                       |
|               serialization.ddl struct transactions { i32 txn_id, i32 acc_id, decimal(10,2) txn_amount} |
|               serialization.format 1               |
|               serialization.lib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|               totalSize 26                         |
|               transient_lastDdlTime 1702847838     |
|             serde: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|                                                    |
|               input format: org.apache.hadoop.mapred.TextInputFormat |
|               output format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|               properties:                          |
|                 bucket_count -1                    |
|                 bucketing_version 2                |
|                 column.name.delimiter ,            |
|                 columns txn_id,acc_id,txn_amount   |
|                 columns.comments                   |
|                 columns.types int:int:decimal(10,2) |
|                 file.inputformat org.apache.hadoop.mapred.TextInputFormat |
|                 file.outputformat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|                 location hdfs://adh/apps/hive/warehouse/explain_demo.db/transactions |
|                 name explain_demo.transactions     |
|                 partition_columns txn_date         |
|                 partition_columns.types date       |
|                 serialization.ddl struct transactions { i32 txn_id, i32 acc_id, decimal(10,2) txn_amount} |
|                 serialization.format 1             |
|                 serialization.lib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|                 transient_lastDdlTime 1702846232   |
|               serde: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|               name: explain_demo.transactions      |
|             name: explain_demo.transactions        |
|           Partition                                |
|             input format: org.apache.hadoop.mapred.TextInputFormat |
|             output format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|             partition values:                      |
|               txn_date 2023-01-02                  |
|             properties:                            |
|               COLUMN_STATS_ACCURATE {"BASIC_STATS":"true","COLUMN_STATS":
{"acc_id":"true","txn_amount":"true","txn_id":"true"}} |
|               bucket_count -1                      |
|               column.name.delimiter ,              |
|               columns txn_id,acc_id,txn_amount     |
|               columns.comments                     |
|               columns.types int:int:decimal(10,2)  |
|               file.inputformat org.apache.hadoop.mapred.TextInputFormat |
|               file.outputformat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|               location hdfs://adh/apps/hive/warehouse/explain_demo.db/transactions/txn_date=2023-01-02 |
|               name explain_demo.transactions       |
|               numFiles 1                           |
|               numRows 2                            |
|               partition_columns txn_date           |
|               partition_columns.types date         |
|               rawDataSize 25                       |
|               serialization.ddl struct transactions { i32 txn_id, i32 acc_id, decimal(10,2) txn_amount} |
|               serialization.format 1               |
|               serialization.lib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|               totalSize 27                         |
|               transient_lastDdlTime 1702847838     |
|             serde: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|                                                    |
|               input format: org.apache.hadoop.mapred.TextInputFormat |
|               output format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|               properties:                          |
|                 bucket_count -1                    |
|                 bucketing_version 2                |
|                 column.name.delimiter ,            |
|                 columns txn_id,acc_id,txn_amount   |
|                 columns.comments                   |
|                 columns.types int:int:decimal(10,2) |
|                 file.inputformat org.apache.hadoop.mapred.TextInputFormat |
|                 file.outputformat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|                 location hdfs://adh/apps/hive/warehouse/explain_demo.db/transactions |
|                 name explain_demo.transactions     |
|                 partition_columns txn_date         |
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
|                 partition_columns.types date       |
|                 serialization.ddl struct transactions { i32 txn_id, i32 acc_id, decimal(10,2) txn_amount} |
|                 serialization.format 1             |
|                 serialization.lib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|                 transient_lastDdlTime 1702846232   |
|               serde: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|               name: explain_demo.transactions      |
|             name: explain_demo.transactions        |
|           Partition                                |
|             input format: org.apache.hadoop.mapred.TextInputFormat |
|             output format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|             partition values:                      |
|               txn_date 2023-01-03                  |
|             properties:                            |
|               COLUMN_STATS_ACCURATE {"BASIC_STATS":"true","COLUMN_STATS":
{"acc_id":"true","txn_amount":"true","txn_id":"true"}} |
|               bucket_count -1                      |
|               column.name.delimiter ,              |
|               columns txn_id,acc_id,txn_amount     |
|               columns.comments                     |
|               columns.types int:int:decimal(10,2)  |
|               file.inputformat org.apache.hadoop.mapred.TextInputFormat |
|               file.outputformat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|               location hdfs://adh/apps/hive/warehouse/explain_demo.db/transactions/txn_date=2023-01-03 |
|               name explain_demo.transactions       |
|               numFiles 1                           |
|               numRows 3                            |
|               partition_columns txn_date           |
|               partition_columns.types date         |
|               rawDataSize 37                       |
|               serialization.ddl struct transactions { i32 txn_id, i32 acc_id, decimal(10,2) txn_amount} |
|               serialization.format 1               |
|               serialization.lib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|               totalSize 40                         |
|               transient_lastDdlTime 1702847838     |
|             serde: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|                                                    |
|               input format: org.apache.hadoop.mapred.TextInputFormat |
|               output format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|               properties:                          |
|                 bucket_count -1                    |
|                 bucketing_version 2                |
|                 column.name.delimiter ,            |
|                 columns txn_id,acc_id,txn_amount   |
|                 columns.comments                   |
|                 columns.types int:int:decimal(10,2) |
|                 file.inputformat org.apache.hadoop.mapred.TextInputFormat |
|                 file.outputformat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|                 location hdfs://adh/apps/hive/warehouse/explain_demo.db/transactions |
|                 name explain_demo.transactions     |
|                 partition_columns txn_date         |
|                 partition_columns.types date       |
|                 serialization.ddl struct transactions { i32 txn_id, i32 acc_id, decimal(10,2) txn_amount} |
|                 serialization.format 1             |
|                 serialization.lib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|                 transient_lastDdlTime 1702846232   |
|               serde: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|               name: explain_demo.transactions      |
|             name: explain_demo.transactions        |
|           Partition                                |
|             input format: org.apache.hadoop.mapred.TextInputFormat |
|             output format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|             partition values:                      |
|               txn_date 2023-01-04                  |
|             properties:                            |
|               COLUMN_STATS_ACCURATE {"BASIC_STATS":"true","COLUMN_STATS":
{"acc_id":"true","txn_amount":"true","txn_id":"true"}} |
|               bucket_count -1                      |
|               column.name.delimiter ,              |
|               columns txn_id,acc_id,txn_amount     |
|               columns.comments                     |
|               columns.types int:int:decimal(10,2)  |
|               file.inputformat org.apache.hadoop.mapred.TextInputFormat |
|               file.outputformat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|               location hdfs://adh/apps/hive/warehouse/explain_demo.db/transactions/txn_date=2023-01-04 |
|               name explain_demo.transactions       |
|               numFiles 1                           |
|               numRows 2                            |
|               partition_columns txn_date           |
|               partition_columns.types date         |
|               rawDataSize 25                       |
|               serialization.ddl struct transactions { i32 txn_id, i32 acc_id, decimal(10,2) txn_amount} |
|               serialization.format 1               |
|               serialization.lib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|               totalSize 27                         |
|               transient_lastDdlTime 1702847838     |
|             serde: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|                                                    |
|               input format: org.apache.hadoop.mapred.TextInputFormat |
|               output format: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|               properties:                          |
|                 bucket_count -1                    |
|                 bucketing_version 2                |
|                 column.name.delimiter ,            |
|                 columns txn_id,acc_id,txn_amount   |
|                 columns.comments                   |
|                 columns.types int:int:decimal(10,2) |
|                 file.inputformat org.apache.hadoop.mapred.TextInputFormat |
|                 file.outputformat org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat |
|                 location hdfs://adh/apps/hive/warehouse/explain_demo.db/transactions |
|                 name explain_demo.transactions     |
|                 partition_columns txn_date         |
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
|                 partition_columns.types date       |
|                 serialization.ddl struct transactions { i32 txn_id, i32 acc_id, decimal(10,2) txn_amount} |
|                 serialization.format 1             |
|                 serialization.lib org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|                 transient_lastDdlTime 1702846232   |
|               serde: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe |
|               name: explain_demo.transactions      |
|             name: explain_demo.transactions        |
|       Processor Tree:                              |
|         TableScan                                  |
|           alias: transactions                      |
|           GatherStats: false                       |
|           Filter Operator                          |
|             isSamplingPred: false                  |
|             predicate: (txn_amount > 100) (type: boolean) |
|             Select Operator                        |
|               expressions: txn_id (type: int), acc_id (type: int), txn_amount (type: decimal(10,2)), txn_date (type: date) |
|               outputColumnNames: _col0, _col1, _col2, _col3 |
|               ListSink                             |
|                                                    |
+----------------------------------------------------+
```

EXPLAIN DEPENDENCY

Команда `EXPLAIN DEPENDENCY` возвращает дополнительную информацию о всех источниках входных данных (таблицы Hive, партиции, HDFS-файлы), к которым осуществляется доступ при выполнении запроса.

```sql
EXPLAIN DEPENDENCY
    SELECT t.txn_id, a.full_name
    FROM transactions t
    JOIN accounts a ON (t.acc_id=a.id)
```

Результат:

```
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
| {"input_tables":[{"tablename":"explain_demo@transactions","tabletype":"MANAGED_TABLE"},
{"tablename":"explain_demo@accounts","tabletype":"MANAGED_TABLE"}],"input_partitions":
[{"partitionName":"explain_demo@transactions@txn_date=2023-01-01"},{"partitionName":"explain_demo@transactions@txn_date=2023-01-
02"},{"partitionName":"explain_demo@transactions@txn_date=2023-01-03"},{"partitionName":"explain_demo@transactions@txn_date=2023-
01-04"}]} |
+----------------------------------------------------+
```

EXPLAIN AUTHORIZATION

Используя `EXPLAIN AUTHORIZATION`, можно получить информацию о всех ошибках авторизации, включая проблемы с правами доступа (если таковые имеются). Например:

```sql
EXPLAIN AUTHORIZATION
    SELECT a.full_name, t.txn_date, t.txn_id from accounts a
    JOIN transactions t ON (a.id=t.acc_id);
```

Результат:

```
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
| INPUTS:                                            |
|   explain_demo@accounts                            |
|   explain_demo@transactions                        |
|   explain_demo@transactions@txn_date=2023-01-01    |
|   explain_demo@transactions@txn_date=2023-01-02    |
|   explain_demo@transactions@txn_date=2023-01-03    |
|   explain_demo@transactions@txn_date=2023-01-04    |
| OUTPUTS:                                           |
|   hdfs://adh/tmp/hive/hive/de6b68f3-0c62-40e9-84ac-8e34916558c9/hive_2023-12-18_09-26-13_342_1170854594803480484-104/-mr-10001
|
| CURRENT_USER:                                      |
|   hive                                             |
| OPERATION:                                         |
|   QUERY                                            |
+----------------------------------------------------+
```

EXPLAIN VECTORIZATION

Ключевое слово `VECTORIZATION` добавляет в вывод `EXPLAIN` информацию, указывающую, почему для определенных Map/Reduce-задач не использовалась векторизация (если таковые задачи имеются).

Синтаксис имеет следующий вид:

```
EXPLAIN VECTORIZATION [ONLY] [SUMMARY|OPERATOR|EXPRESSION|DETAIL]
```

Где:

- `ONLY` — скрывает большинство элементов, не связанных с векторизацией.
- `SUMMARY` — отображает, включена ли векторизация, а также добавляет краткую информацию о работе Map/Reduce-задач.
- `OPERATOR` — показывает информацию о векторизации для операторов.
- `EXPRESSION` — показывает информацию о векторизации для выражений.
- `DETAIL` — наиболее подробный уровень информации. Включает в себя все вышеперечисленное.

Сбор статистики с помощью ANALYZE

При обработке запросов оптимизатор CBO активно использует статистику Hive для выбора оптимального плана выполнения запроса с точки зрения системных ресурсов. Эта статистика (количество строк, количество файлов, размер сырых данных) хранится в Hive Metastore и по умолчанию генерируется автоматически при добавлении новых данных в таблицы Hive (`hive.stats.autogather=true`).

Сбор статистики

Для сбора статистики определенной таблицы или партиции в любой момент времени используйте команду `ANALYZE`. Эта команда вычисляет статистику по запросу и записывает ее в Hive Metastore. Hive поддерживает статистику на уровне таблиц, партиций и столбцов.

Синтаксис выглядит следующим образом:

```
ANALYZE TABLE <tbl_name> [PARTITION(<partition>)] COMPUTE STATISTICS
[FOR COLUMNS c1, c2, ...]
[NOSCAN];
```

Где:

- `FOR COLUMNS` — вычисляет статистику либо для всех столбцов таблицы (если имена столбцов не указаны), либо только для указанных столбцов.
- `NOSCAN` — если параметр указан, команда не сканирует все файлы. Вместо этого в статистике будет содержаться информация о количестве байт и физическом размере в байтах.

> **ПРИМЕЧАНИЕ**
>
> По умолчанию сбор статистики включен для вновь созданных таблиц/партиций, поэтому нет необходимости вычислять статистику вручную, если только не было изменено дефолтное значение свойства `hive.stats.autogather=true`.

После сохранения статистики в Hive Metastore вы можете просмотреть статистику, выполнив команду `DESCRIBE`. Синтаксис команды следующий:

```
DESCRIBE [EXTENDED|FORMATTED] <tbl_name> [PARTITION(<partition>)]
```

В следующем примере показан сбор и вывод статистики Hive для всей таблицы:

```sql
ANALYZE TABLE transactions COMPUTE STATISTICS;
DESCRIBE EXTENDED transactions;
```

Вывод:

```
+-----------------------------+-----------------------------------------------------+----------+
|          col_name           |                      data_type                      | comment  |
+-----------------------------+-----------------------------------------------------+----------+
| txn_id                      | int                                                 |          |
| acc_id                      | int                                                 |          |
| txn_amount                  | decimal(10,2)                                       |          |
| txn_date                    | date                                                |          |
|                             | NULL                                                | NULL     |
| # Partition Information     | NULL                                                | NULL     |
| # col_name                  | data_type                                           | comment  |
| txn_date                    | date                                                |          |
|                             | NULL                                                | NULL     |
| Detailed Table Information  | Table(tableName:transactions, explain_demo, owner:hive, createTime:1702839122, lastAccessTime:0,
retention:0, sd:StorageDescriptor(cols:[FieldSchema(name:txn_id, type:int, comment:null), FieldSchema(name:acc_id, type:int,
comment:null), FieldSchema(name:txn_amount, type:decimal(10,2), comment:null), FieldSchema(name:txn_date, type:date,
comment:null)], location:hdfs://adh/apps/hive/warehouse/transactions, inputFormat:org.apache.hadoop.mapred.TextInputFormat,
outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compressed:false, numBuckets:-1,
serdeInfo:SerDeInfo(name:null, serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{serialization.
format=1}), bucketCols:[], sortCols:[], parameters:{}, skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[],
skewedColValueLocationMaps:{}), storedAsSubDirectories:false), partitionKeys:[FieldSchema(name:txn_date, type:date,
comment:null)], parameters:{totalSize=120, numRows=9, rawDataSize=111, COLUMN_STATS_ACCURATE={\"BASIC_STATS\":\"true\"},
numFiles=4, numPartitions=4, transient_lastDdlTime=1702839122, bucketing_version=2}, viewOriginalText:null,
viewExpandedText:null, tableType:MANAGED_TABLE, rewriteEnabled:false, catName:hive, ownerType:USER) |          |
+-----------------------------+-----------------------------------------------------+----------+
```

Для отображения статистики определенной партиции в Hive-таблице используйте следующий пример:

```sql
ANALYZE TABLE transactions PARTITION(txn_date='2023-01-03') COMPUTE STATISTICS;
DESCRIBE EXTENDED transactions PARTITION(txn_date='2023-01-03');
```

Подробная статистика партиции содержится в блоке Detailed Partition Information.

```
+---------------------------------+-----------------------------------------------------+----------+
|            col_name             |                      data_type                      | comment  |
+---------------------------------+-----------------------------------------------------+----------+
| txn_id                          | int                                                 |          |
| acc_id                          | int                                                 |          |
| txn_amount                      | decimal(10,2)                                       |          |
| txn_date                        | date                                                |          |
|                                 | NULL                                                | NULL     |
| # Partition Information         | NULL                                                | NULL     |
| # col_name                      | data_type                                           | comment  |
| txn_date                        | date                                                |          |
|                                 | NULL                                                | NULL     |
| Detailed Partition Information  | Partition(values:[2023-01-03], dbName:explain_demo, tableName:transactions, createTime:0,
lastAccessTime:0, sd:StorageDescriptor(cols:[FieldSchema(name:txn_id, type:int, comment:null), FieldSchema(name:acc_id, type:int,
comment:null), FieldSchema(name:txn_amount, type:decimal(10,2), comment:null), FieldSchema(name:txn_date, type:date,
comment:null)], location:hdfs://adh/apps/hive/warehouse/transactions/txn_date=2023-01-03, inputFormat:org.apache.hadoop.mapred.
TextInputFormat, outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compressed:false, numBuckets:-1,
serdeInfo:SerDeInfo(name:null, serializationLib:org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{serialization.
format=1}), bucketCols:[], sortCols:[], parameters:{}, skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[],
skewedColValueLocationMaps:{}), storedAsSubDirectories:false), parameters:{totalSize=40, numRows=3, rawDataSize=37,
COLUMN_STATS_ACCURATE={"BASIC_STATS":"true","COLUMN_STATS":{"acc_id":"true","txn_amount":"true","txn_id":"true"}}, numFiles=1,
transient_lastDdlTime=1702844180}, catName:hive) |          |
+---------------------------------+-----------------------------------------------------+----------+
```

Подзапросы в Hive

Константин Алпашкин

Содержание

Назад к содержанию

- Тестовая база данных
- Подзапросы в предложении WHERE
  - Подзапросы с IN
  - Подзапросы с NOT IN
  - Подзапросы с EXISTS
  - Подзапросы с агрегатными функциями
- Подзапросы в предложении FROM
  - Пример вложенного подзапроса SELECT

Подзапрос в Hive — это вложенное выражение HiveQL, которое возвращает набор данных при выполнении родительского (внешнего) запроса. На момент выхода версии ADH 3.1.2 Hive поддерживает использование подзапросов внутри предложений FROM и WHERE.

При обработке запросов с вложенным подзапросом сначала выполняется подзапрос, а затем его результаты могут быть использованы в родительском запросе. Механизм подзапросов включен по умолчанию и не требует ручной активации.

Тестовая база данных

В данной статье для демонстрации подзапросов используются Hive-таблицы `employees` и `departments`, структура которых описана ниже.

```
SELECT * FROM employees;
+--------------+----------------+---------------------+----------------------------+---------------+-------------------+
| employees.id | employees.name | employees.last_name |      employees.email       | employees.age | employees.dept_id |
+--------------+----------------+---------------------+----------------------------+---------------+-------------------+
| 1            | Ivan           | Ivanov              | ivan_ivanov123@mail.ru     | 30            | 1                 |
| 2            | Sarah          | Connor              | sarah_connor123@yahoo.com  | 35            | 2                 |
| 3            | Rick           | Sanchez             | rick_123@mail.ru           | 29            | 2                 |
| 4            | John           | Smith               | john_smith123@gmail.com    | 50            | 3                 |
+--------------+----------------+---------------------+----------------------------+---------------+-------------------+

SELECT * FROM departments;
+-----------------+-----------------------+-------------------------+------------------------+
| departments.id  | departments.dep_name  | departments.dep_office  | departments.dep_phone  |
+-----------------+-----------------------+-------------------------+------------------------+
| 1               | sales                 | 110                     | 9379992                |
| 2               | it                    | 115                     | 12345678               |
| 3               | support               | 125                     | 880080080              |
+-----------------+-----------------------+-------------------------+------------------------+
```

SQL для создания тестовых таблиц

Для создания и наполнения тестовых таблиц Hive выполните следующий SQL с помощью */bin/beeline*.

```sql
CREATE DATABASE hr;

USE hr;

CREATE TABLE IF NOT EXISTS hr.employees (id int, name string, last_name string, email string, age int, salary int, dept_id int)
COMMENT 'Employee Table'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

INSERT INTO hr.employees VALUES
(1, 'Ivan', 'Ivanov', 'ivan_ivanov123@mail.ru', 30, 1000, 1),
(2, 'Sarah', 'Connor', 'sarah_connor123@yahoo.com', 35, 1500, 2),
(3, 'Rick', 'Sanchez', 'rick_123@mail.ru', 29, 1300, 2),
(4, 'John', 'Smith', 'john_smith123@gmail.com', 29, 2000, 3);

CREATE TABLE IF NOT EXISTS hr.departments (id int, dep_name string, dep_office int, dep_phone string)
COMMENT 'Departments Table'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

INSERT INTO hr.departments VALUES
(1, 'sales', 110, '9379992'),
(2, 'it', 115, '12345678'),
(3, 'support', 125, '880080080');
```

Подзапросы в предложении WHERE

Внутри предложения `WHERE` допускается только один подзапрос. Подзапрос считается **коррелированным (correlated)**, если он содержит предикат с оператором `=` и одна сторона оператора ссылается хотя бы на один столбец из родительского запроса, а другая сторона — хотя бы на один столбец из подзапроса. При использовании внутри `WHERE` результат выполнения подзапроса часто передается операторам `IN` / `EXIST` для проверки на вхождение/наличие.

Подзапросы с IN

В примере ниже подзапрос получает идентификатор департамента по номеру офиса, а затем внешним запросом извлекаются данные о всех сотрудниках, работающих в этом департаменте.

```sql
SELECT e.name, e.last_name, e.email FROM employees e
WHERE e.dept_id IN
    (SELECT id FROM departments
    WHERE dep_office=110);
```

Результат:

```
+---------+--------------+-------------------------+
| e.name  | e.last_name  |         e.email         |
+---------+--------------+-------------------------+
| Ivan    | Ivanov       | ivan_ivanov123@mail.ru  |
+---------+--------------+-------------------------+
```

Подзапросы с NOT IN

В следующем примере подзапрос возвращает внутренний идентификатор департамента по номеру офиса. Далее внешний запрос возвращает информацию о всех сотрудниках, которые не находятся в данном офисе.

```sql
 SELECT e.name, e.last_name, e.email
 FROM employees e
 WHERE e.dept_id NOT IN
       (SELECT d.id
        FROM departments d
        WHERE d.dep_office=110)
```

Результат:

```
+---------+--------------+----------------------------+
| e.name  | e.last_name  |           e.email          |
+---------+--------------+----------------------------+
| Sarah   | Connor       | sarah_connor123@yahoo.com  |
| Rick    | Sanchez      | rick_123@mail.ru           |
| John    | Smith        | john_smith123@gmail.com    |
+---------+--------------+----------------------------+
```

Подзапросы с EXISTS

Следующий запрос возвращает номера офисов, в которых находятся сотрудники определенного возраста. Сначала подзапрос получает список сотрудников с указанным возрастом, а затем полученный результат передается оператору `EXISTS`. Для каждой строки, возвращаемой подзапросом, внешний запрос извлекает соответствующий номер офиса из таблицы `departments`.

```sql
SELECT d.dep_office
FROM departments d
WHERE EXISTS (
    SELECT * FROM employees e
    WHERE e.dept_id=d.id AND e.age=29
    );
```

Результат:

```
+---------------+
| d.dep_office  |
+---------------+
| 115           |
| 125           |
+---------------+
```

Подзапросы с агрегатными функциями

Hive предоставляет ограниченную поддержку агрегатных функций в подзапросах. Следующий пример возвращает список сотрудников, чья заработная плата выше среднего значения.

```sql
SELECT e.last_name, e.salary FROM employee e
WHERE salary > (
    SELECT AVG(salary)
    FROM employee);
```

Результат:

```
+--------------+-----------+
| e.last_name  | e.salary  |
+--------------+-----------+
| Smith        | 2000      |
| Connor       | 1500      |
+--------------+-----------+
```

Подзапросы в предложении FROM

Количество подзапросов, вложенных в предложение `FROM`, может быть произвольным. Подзапросы `SELECT` в предложениях `FROM` должны быть именованными, как и таблицы внутри предложения `FROM` имеют свое имя. Все столбцы в выборке подзапроса должны иметь уникальные имена; данные столбцов, полученные в результате выполнения подзапроса, доступны во внешнем запросе под этими именами как обычные столбцы таблицы. Например:

```sql
SELECT t.name, t.email
FROM (SELECT e.name, e.email
      FROM employees e
      WHERE e.age=35) t; ①
```

① Алиас (`t`) необходим для идентификации подзапроса `SELECT`. По этому идентификатору можно получить значение столбца подзапроса из внешнего запроса.

Пример вложенного подзапроса SELECT

В следующем примере подзапрос выполняет выборку данных из явно соединенных таблиц `employees` и `departments`. Затем во внешнем запросе формируется новый почтовый адрес для каждого сотрудника на основе данных, полученных в подзапросе.

```sql
SELECT t.old_email, concat(t.name, t.last_name, '_', t.department_name, '@new.domain.org')
FROM
    (SELECT e.name, e.last_name, e.email AS old_email, d.dep_name AS department_name
     FROM employees e
     JOIN departments d ON e.dept_id=d.id
     ) t;
```

Результат:

```
+----------------------------+-----------------------------------+
|        t.old_email         |            new_email              |
+----------------------------+-----------------------------------+
| sarah_connor123@yahoo.com  | SarahConnor_it@new.domain.org     |
| rick_123@mail.ru           | RickSanchez_it@new.domain.org     |
| john_smith123@gmail.com    | JohnSmith_support@new.domain.org  |
| ivan_ivanov123@mail.ru     | IvanIvanov_sales@new.domain.org   |
+----------------------------+-----------------------------------+
```

Агрегатные функции в Hive

Константин Алпашкин

Содержание

Назад к содержанию

- Тестовая база данных
- Базовая агрегация
  - Другие агрегатные функции Hive
- Расширенная агрегация
  - GROUPING SETS
  - WITH CUBE
  - WITH ROLLUP

Агрегатные функции Hive — это встроенные функции, которые принимают несколько значений и возвращают одно агрегированное значение. В данной статье приведены примеры и рекомендации по использованию базовых и расширенных агрегатных функций Hive. Полный список агрегатных функций, поддерживаемых в Hive по умолчанию, доступен в документации Apache Hive.

Тестовая база данных

В данной статье для демонстрации агрегатных функций используются Hive-таблицы `employees` и `transactions`, структура которых описана ниже.

```
SELECT * FROM employees;
+----------------+------------------+----------------------+-----------------------------+----------------+-------------------+-----
---------------+
| employees.id  | employees.name  | employees.last_name  |      employees.email       | employees.age  | employees.salary  |
employees.dept_id  |
+----------------+------------------+----------------------+-----------------------------+----------------+-------------------+-----
---------------+
| 1             | Ivan            | Ivanov               | ivan_ivanov123@mail.ru     | 30             | 1000              | 1
|
| 2             | Sarah           | Connor               | sarah_connor123@yahoo.com  | 35             | 1500              | 2
|
| 3             | Rick            | Sanchez              | rick_123@mail.ru           | 29             | 1300              | 2
|
| 4             | John            | Smith                | john_smith123@gmail.com    | 29             | 2000              | 3
|
+----------------+------------------+----------------------+-----------------------------+----------------+-------------------+-----
---------------+


SELECT * FROM transactions;
+-----------------------+----------------------+---------------------------+-------------------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  |
+-----------------------+----------------------+---------------------------+-------------------------+
| 1                    | 101                  | 10.20                    | 2023-10-26             |
| 2                    | 101                  | 102.30                   | 2023-10-26             |
| 3                    | 102                  | 95.00                    | 2023-10-26             |
| 4                    | 103                  | 900.50                   | 2023-10-27             |
| 5                    | 102                  | 910.50                   | 2023-10-27             |
| 6                    | 101                  | 850.00                   | 2023-10-27             |
+-----------------------+----------------------+---------------------------+-------------------------+
```

SQL для создания тестовых таблиц

Для создания и наполнения тестовых таблиц Hive выполните следующий SQL с помощью */bin/beeline*.

```sql
CREATE TABLE IF NOT EXISTS employees (id int, name string, last_name string, email string, age int, salary int, dept_id int)
COMMENT 'Employee Table'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

INSERT INTO employees VALUES
(1, 'Ivan', 'Ivanov', 'ivan_ivanov123@mail.ru', 30, 1000, 1),
(2, 'Sarah', 'Connor', 'sarah_connor123@yahoo.com', 35, 1500, 2),
(3, 'Rick', 'Sanchez', 'rick_123@mail.ru', 29, 1300, 2),
(4, 'John', 'Smith', 'john_smith123@gmail.com', 29, 2000, 3);


CREATE TABLE IF NOT EXISTS transactions(txn_id int, acc_id int, txn_amount decimal(10,2), txn_date date)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe';

INSERT INTO transactions VALUES
(1, 101, 10.20, '2023-10-26'),
(2, 101, 102.30, '2023-10-26'),
(3, 102, 95.00, '2023-10-26'),
(4, 103, 900.50, '2023-10-27'),
(5, 102, 910.50, '2023-10-27'),
(6, 101, 850.00, '2023-10-27');
```

Базовая агрегация

Функция `COUNT(col)` возвращает общее количество строк в результирующем наборе данных.

```sql
SELECT COUNT(id) AS emp_total_cnt FROM employees;
```

Результат:

```
+-----------------+
| emp_total_cnt  |
+-----------------+
| 4              |
+-----------------+
```

РЕКОМЕНДАЦИЯ

С помощью ключевого слова `DISTINCT` можно получить агрегированное значение только для уникальных значений в столбце.

Функции `MIN()` и `MAX()` возвращают минимальное/максимальное значение числового столбца.

```sql
SELECT MIN(salary) from employees;
```

Результат:

```
+-------+
|  _c0  |
+-------+
| 1000  |
+-------+
```

Функция `AVG()` возвращает среднее значение числового столбца. Функция не может использоваться внутри предложения `WHERE`. Например, следующий запрос приведет к ошибке:

```sql
SELECT e.salary, e.age
FROM employees e
WHERE e.age > AVG(e.age);
```

Текст ошибки:

```
Error: Error while compiling statement: FAILED: SemanticException [Error 10128]: Line 3:14 Not yet supported place for UDAF 'avg'
(state=42000,code=10128)
```

В этом конкретном случае в качестве обходного пути можно вложить `AVG()` внутрь подзапроса как показано ниже.

```sql
SELECT e.age, e.last_name
FROM employees e
WHERE e.age > (SELECT AVG(age) FROM employees);
```

Результат успешного выполнения запроса:

```
+--------+--------------+
| e.age  | e.last_name  |
+--------+--------------+
| 35     | Connor       |
+--------+--------------+
```

Другие агрегатные функции Hive

Помимо стандартных агрегатных функций SQL, таких как `SUM`, `AVG` и прочих, Hive предоставляет дополнительные функции, которые могут пригодиться при анализе наборов данных. Например, с помощью `collect_*(col)` можно упаковать все значения столбца `col` в список или множество.

```sql
SELECT collect_set(age) FROM employees; ①
SELECT d.dep_name AS department_name, collect_list(e.age) AS ages ②
FROM employees e
    JOIN departments d
    ON e.dept_id=d.id
GROUP BY d.dep_name;
```

① Возвращает единственную строку с множеством уникальных значений.

② Возвращает список значений для каждого департамента.

Результат:

```
+-------------+
|     _c0     |
+-------------+
| [30,35,29]  |
+-------------+

+------------------+----------+
| department_name  |   ages   |
+------------------+----------+
| it               | [35,29]  |
| sales            | [30]     |
| support          | [29]     |
+------------------+----------+
```

Hive также предоставляет специальные функции для вычисления математических показателей, таких как отклонение, дисперсия и другие. Примеры использования некоторых из них показаны ниже.

```sql
SELECT variance(salary) FROM employees; ①
SELECT var_pop(salary) FROM employees; ①
SELECT var_samp(salary) FROM employees; ②
SELECT stddev_pop(salary) FROM employees; ③
SELECT stddev_samp(salary) FROM employees; ④
```

① Запросы возвращают показатель дисперсии для числового столбца `salary`.

② Возвращает показатель несмещенной выборочной дисперсии (unbiased sample variance) для столбца.

③ Возвращает статистическое стандартное отклонение для значений в столбце.

④ Возвращает показатель несмещенного выборочного стандартного отклонения (unbiased sample standard deviation) для числового столбца.

Результат:

```
SELECT variance(salary) FROM employees;
+-----------+
|    _c0    |
+-----------+
| 132500.0  |
+-----------+

SELECT var_pop(salary) FROM employees;
+-----------+
|    _c0    |
+-----------+
| 132500.0  |
+-----------+

SELECT var_samp(salary) FROM employees;
+---------------------+
|         _c0         |
+---------------------+
| 176666.66666666666  |
+---------------------+

SELECT stddev_pop(salary) FROM employees;
+--------------------+
|        _c0         |
+--------------------+
| 364.0054944640259  |
+--------------------+

SELECT stddev_samp(salary) FROM employees;
+---------------------+
|         _c0         |
+---------------------+
| 420.31734043061635  |
+---------------------+
```

Полный список поддерживаемых агрегатных функций доступен в документации Apache Hive.

Расширенная агрегация

GROUPING SETS

Предложение `GROUPING SETS`, используемое в паре с `GROUP BY`, позволяет указывать несколько критериев `GROUP BY` в одном запросе. Использование нескольких группирующих наборов (например, `GROUPING SETS (a,b)`) эквивалентно выполнению нескольких запросов с разными выражениями `GROUP BY a`, `GROUP BY b` и объединенных с помощью `UNION`. Стоит отметить, что выборка с помощью группирующих наборов выполняется за одно сканирование таблицы, а не в несколько итераций, как в случае с отдельными запросами, объединенными `UNION`.

В следующем примере используется только один группирующий набор (`GROUPING SETS (acc_id)`), который группирует результаты по аккаунту, но без агрегации по дате.

```sql
SELECT acc_id, txn_date, SUM(txn_amount) as total_paid
FROM transactions
GROUP BY acc_id, txn_date
GROUPING SETS (acc_id);
```

Результатом такого запроса является общая сумма, оплаченная каждым аккаунтом. Значения `NULL` в столбце `txn_date` указывают на отсутствие агрегации по дате.

```
+---------+-----------+-------------+
| acc_id  | txn_date  | total_paid  |
+---------+-----------+-------------+
| 101     | NULL      | 962.50      |
| 102     | NULL      | 1005.50     |
| 103     | NULL      | 900.50      |
+---------+-----------+-------------+
```

В следующем примере показано использование нескольких группирующих наборов (`GROUPING SETS (acc_id, txn_date)`).

```sql
SELECT acc_id, txn_date, SUM(txn_amount) as total_paid, grouping__id
FROM transactions
GROUP BY acc_id, txn_date
GROUPING SETS (acc_id, txn_date);
```

В этом случае результат выборки содержит наборы строк, сгруппированных по `txn_date` и `acc_id`:

```
+---------+-------------+-------------+
| acc_id  |  txn_date   | total_paid  |
+---------+-------------+-------------+
| NULL    | 2023-10-26  | 207.50      |
| NULL    | 2023-10-27  | 2661.00     |
| 101     | NULL        | 962.50      |
| 102     | NULL        | 1005.50     |
| 103     | NULL        | 900.50      |
+---------+-------------+-------------+
```

Значения `NULL` указывают, что агрегация по данному столбцу не выполнялась. Обратите внимание, что полученный результат представляет собой `UNION` двух результатов, полученных с помощью следующих запросов:

```sql
SELECT NULL, txn_date, SUM(txn_amount)
FROM transactions
GROUP BY txn_date
UNION
SELECT acc_id, NULL, SUM(txn_amount)
FROM transactions
GROUP BY acc_id;
```

ВАЖНО

Выполнение отдельных запросов с `UNION` запускает две задачи вместо одной и потенциально может занять больше времени.

WITH CUBE

Функция `CUBE` используется вместе с `GROUP BY` для вычисления всех возможных комбинаций агрегации столбцов, указанных в `GROUP BY`. Следующие выражения HiveQL тождественны:

```sql
... GROUP BY a, b, c WITH CUBE
-- эквивалентно:
... GROUP BY a, b, c GROUPING SETS (
    (a, b, c), (a, b), (b, c), (a, c), (a), (b), (c), ( )
)
```

Ниже показан пример использования `WITH CUBE`.

```sql
SELECT acc_id, txn_date, SUM(txn_amount) as total_sum
FROM transactions
GROUP BY acc_id, txn_date WITH CUBE;
```

Результат содержит комбинацию всех возможных способов группировки данных:

```
+---------+-------------+-------------+
| acc_id  |  txn_date   | total_sum   |
+---------+-------------+-------------+
| NULL    | NULL        | 2868.50     |
| NULL    | 2023-10-26  | 207.50      |
| NULL    | 2023-10-27  | 2661.00     |
| 101     | NULL        | 962.50      |
| 101     | 2023-10-26  | 112.50      |
| 101     | 2023-10-27  | 850.00      |
| 102     | NULL        | 1005.50     |
| 102     | 2023-10-26  | 95.00       |
| 102     | 2023-10-27  | 910.50      |
| 103     | NULL        | 900.50      |
| 103     | 2023-10-27  | 900.50      |
+---------+-------------+-------------+
```

В этом примере использование конструкции `WITH CUBE` эквивалентно следующему запросу:

```sql
SELECT acc_id, txn_date, SUM(txn_amount) as total_sum
FROM transactions
GROUP BY acc_id, txn_date
GROUPING SETS ((acc_id, txn_date), acc_id, txn_date, ( ));
```

WITH ROLLUP

Функция `ROLLUP` используется вместе с `GROUP BY` для вычисления нескольких уровней промежуточных результатов по заданной группе измерений. Функция вычисляет промежуточные результаты агрегации от самого детального уровня до самого общего, следуя списку критериев, указанных в `GROUP BY`. Следующие конструкции HiveQL тождественны:

```sql
... GROUP BY a, b, c, WITH ROLLUP
-- эквивалентно:
... GROUP BY a, b, c GROUPING SETS ( (a, b, c), (a, b), (a), ( ))
```

Ниже показан пример использования `WITH ROLLUP`.

```sql
SELECT acc_id, txn_date, SUM(txn_amount) as total_sum
FROM transactions
GROUP BY acc_id, txn_date WITH ROLLUP;
```

Результат содержит записи с различными уровнями агрегации — от наиболее детальной (сумма, оплаченная определенным аккаунтом в конкретный день) до наиболее общих показателей (общая сумма уплаченных денег).

```
+---------+-------------+------------+
| acc_id  |  txn_date   | total_sum  |
+---------+-------------+------------+
| NULL    | NULL        | 2868.50    |
| 101     | NULL        | 962.50     |
| 101     | 2023-10-26  | 112.50     |
| 101     | 2023-10-27  | 850.00     |
| 102     | NULL        | 1005.50    |
| 102     | 2023-10-26  | 95.00      |
| 102     | 2023-10-27  | 910.50     |
| 103     | NULL        | 900.50     |
| 103     | 2023-10-27  | 900.50     |
+---------+-------------+------------+
```

В этом примере использование конструкции `ROLLUP` эквивалентно следующему запросу:

```sql
SELECT acc_id, txn_date, SUM(txn_amount) as total_sum
FROM transactions
GROUP BY acc_id, txn_date
GROUPING SETS ((acc_id, txn_date), acc_id, ( ));
```

Оконные и аналитические функции в Hive

Константин Алапшкин

Содержание

Назад к содержанию

- Тестовая база данных
- Синтаксис
  - Описание окна
- Ключевое слово OVER
  - OVER + PARTITION BY
  - OVER + ORDER BY
- Оконные функции
  - LEAD и LAG
  - ROW_NUMBER
  - FIRST_VALUE и LAST_VALUE
- Аналитические функции
  - RANK и DENSE_RANK
  - PERCENT_RANK
  - NTILE

Оконные и аналитические функции Hive позволяют производить вычисления над выделенным набором строк (окном, партицией) в исходной выборке `SELECT`. В отличие от агрегатных функций, которые сводят несколько значений столбцов к одному значению, результатом работы оконной функции является один или более столбцов с результатами вычислений, добавленных к исходному набору результатов. Оконные и аналитические функции очень полезны для решения аналитических задач и являются незаменимым инструментом для таких запросов как "получить топ X проданных товаров по категориям", "получить значение максимальной зарплаты по отделам", "получить первые N строк из результирующего набора" и так далее.

Все функции можно условно разделить на группы:

- Оконные функции, такие как `LEAD`, `LAG`, `FIRST_VALUE` и прочие.
- Аналитические функции, например `RANK`, `ROW_NUMBER`, `NTILE` и другие.

В статье приведены примеры и рекомендации по использованию основных оконных и аналитических функций Hive. Более детальная информация о поддерживаемых функциях Hive доступна в документации Apache Hive.

Тестовая база данных

Для демонстрации примеров в данной статье используются Hive-таблица `transactions`, структура которой описана ниже.

```
SELECT * FROM transactions;
+-----------------------+-----------------------+--------------------------+------------------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  |
+-----------------------+-----------------------+--------------------------+------------------------+
| 1                     | 1002                  | 10.00                    | 2023-01-01             |
| 2                     | 1002                  | 20.00                    | 2023-01-03             |
| 3                     | 1002                  | 30.00                    | 2023-01-02             |
| 4                     | 1001                  | 100.50                   | 2023-01-02             |
| 5                     | 1001                  | 150.50                   | 2023-01-04             |
| 6                     | 1001                  | 200.50                   | 2023-01-03             |
| 7                     | 1003                  | 50.00                    | 2023-01-03             |
| 8                     | 1003                  | 50.00                    | 2023-01-01             |
| 9                     | 1003                  | 75.00                    | 2023-01-04             |
+-----------------------+-----------------------+--------------------------+------------------------+
```

SQL для создания тестовой таблицы

Для создания и наполнения тестовой таблицы Hive выполните следующий SQL с помощью */bin/beeline*.

```
DROP TABLE IF EXISTS `transactions`;
CREATE TABLE `transactions`(`txn_id` int, `acc_id` int, `txn_amount` decimal(10,2), `txn_date` date);
INSERT INTO transactions VALUES
(1, 1002, 10.00, '2023-01-01'),
(2, 1002, 20.00, '2023-01-03'),
(3, 1002, 30.00, '2023-01-02'),
(4, 1001, 100.50, '2023-01-02'),
(5, 1001, 150.50, '2023-01-04'),
(6, 1001, 200.50, '2023-01-03'),
(7, 1003, 50.00, '2023-01-03'),
(8, 1003, 50.00, '2023-01-01'),
(9, 1003, 75.00, '2023-01-04');
```

Синтаксис

Базовый синтаксис оконных и аналитических функций имеет следующий вид:

```
SELECT <func>([[, ...]]) OVER ([<window_spec>]) [<window_name>]
FROM <table_name>;
```

Где:

- `<func>` — стандартная агрегатная функция SQL (например, `AVG`) или специальная оконная/аналитическая функция.
- `OVER` — означает, что функция должна выполняться для выделенных окон в исходном наборе результатов.
- `<window_spec>` — выражение, определяющее критерий формирования окон в наборе результатов. Подробности использования описаны ниже.
- `<window_name>` — алиас для столбца с результатами выполнения функции.

Описание окна

Описание окна (window specification) позволяет указывать границы и размеры окон в результатах выборки. Пустое описание означает, что весь результирующий набор является одним большим окном и функция должна выполняться для всех строк. Синтаксис описания окна показан ниже:

```
[PARTITION BY <col> [, ...]]
[ORDER BY <col> [ASC|DESC] [, ...]]
(ROWS | RANGE) BETWEEN (UNBOUNDED | [num]) PRECEDING AND ([num] PRECEDING | CURRENT ROW | (UNBOUNDED | [num]) FOLLOWING)
```

Где:

- `PARTITION BY <col>` — разбивает исходный результирующий набор на партиции, используя значения столбца `<col>`.
- `ORDER BY <col>` — упорядочивает строки в каждой партиции перед выполнением функции.
- `ROWS | RANGE …` — указывает верхний и нижний пределы фрейма окна.

Также существует альтернативный синтаксис с отдельным указанием предложения `WINDOW`:

```
SELECT <func>([[, ...]]) OVER <window_name>
FROM <table_name>
WINDOW <window_name> AS ([<window_spec>])
```

Ключевое слово OVER

В самом простом виде оконная функция объявляется путем добавления `OVER` в запрос `SELECT`. При указании столбцов для выборки могут использоваться оконные/аналитические функции, а также стандартные агрегатные функции SQL (`SUM`, `AVG` и прочие).

В следующем примере пустое предложение `OVER()` означает, что границы окна явно не указаны. Поэтому, весь результирующий набор, который вернул `SELECT`, является одним большим окном для вычисления средней суммы транзакций.

```
SELECT *, AVG(txn_amount)
OVER() AS average_txn
FROM transactions;
```

Результат содержит дополнительный столбец `average_txn`, в котором хранится среднее значение для всех транзакций:

```
+-----------------------+-----------------------+--------------------------+------------------------+--------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  | average_txn  |
+-----------------------+-----------------------+--------------------------+------------------------+--------------+
| 1                     | 1002                  | 10.00                    | 2023-01-01             | 76.277778    |
| 2                     | 1002                  | 20.00                    | 2023-01-03             | 76.277778    |
| 3                     | 1002                  | 30.00                    | 2023-01-02             | 76.277778    |
| 4                     | 1001                  | 100.50                   | 2023-01-02             | 76.277778    |
| 5                     | 1001                  | 150.50                   | 2023-01-04             | 76.277778    |
| 6                     | 1001                  | 200.50                   | 2023-01-03             | 76.277778    |
| 7                     | 1003                  | 50.00                    | 2023-01-03             | 76.277778    |
| 8                     | 1003                  | 50.00                    | 2023-01-01             | 76.277778    |
| 9                     | 1003                  | 75.00                    | 2023-01-04             | 76.277778    |
+-----------------------+-----------------------+--------------------------+------------------------+--------------+
```

OVER + PARTITION BY

Использование конструкции `OVER (PARTITION BY col)` позволяет условно разделить результирующий набор на партиции (partitions), чтобы функция выполнялась для каждой отдельной партиции.

Например, следующий запрос вычисляет среднюю сумму транзакций для каждого аккаунта.

```
SELECT *, AVG(txn_amount)
OVER (PARTITION BY acc_id) AS average_txn_per_acc
FROM transactions;
```

Теперь в результатах содержатся средние значения, рассчитанные с привязкой к определенному аккаунту:

```
+-----------------------+-----------------------+--------------------------+------------------------+----------------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  | average_txn_per_acc  |
+-----------------------+-----------------------+--------------------------+------------------------+----------------------+
| 4                     | 1001                  | 100.50                   | 2023-01-02             | 150.500000           |
| 5                     | 1001                  | 150.50                   | 2023-01-04             | 150.500000           |
| 6                     | 1001                  | 200.50                   | 2023-01-03             | 150.500000           |
| 1                     | 1002                  | 10.00                    | 2023-01-01             | 20.000000            |
| 2                     | 1002                  | 20.00                    | 2023-01-03             | 20.000000            |
| 3                     | 1002                  | 30.00                    | 2023-01-02             | 20.000000            |
| 7                     | 1003                  | 50.00                    | 2023-01-03             | 58.333333            |
| 8                     | 1003                  | 50.00                    | 2023-01-01             | 58.333333            |
| 9                     | 1003                  | 75.00                    | 2023-01-04             | 58.333333            |
+-----------------------+-----------------------+--------------------------+------------------------+----------------------+
```

OVER + ORDER BY

С помощью ключевого слова `ORDER` можно реализовать сортировку в функции.

Например, следующий запрос вычисляет общую сумму, оплаченную каждым аккаунтом в определенный день, сортируя результаты по дате.

```
SELECT *, SUM(txn_amount)
OVER (PARTITION BY acc_id ORDER BY txn_date) AS running_total_paid
FROM transactions;
```

Теперь для каждого аккаунта строки отсортированы по дате транзакции. Обратите внимание, что столбец `running_total_paid` содержит значения нарастающего итога. Это происходит потому, что появление `ORDER BY` в описании окна приводит к неявному использованию `RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW` для обозначения предела фрейма окна. Таким образом, при подсчете среднего значения для каждой строки функция учитывает значение текущей строки + значения всех предыдущих строк в окне. Без `ORDER BY` общая сумма в столбце `running_total_paid` была бы одинаковой для всех записей.

```
+-----------------------+-----------------------+--------------------------+------------------------+---------------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  | running_total_paid  |
+-----------------------+-----------------------+--------------------------+------------------------+---------------------+
| 4                     | 1001                  | 100.50                   | 2023-01-02             | 100.50              |
| 6                     | 1001                  | 200.50                   | 2023-01-03             | 301.00              |
| 5                     | 1001                  | 150.50                   | 2023-01-04             | 451.50              |
| 1                     | 1002                  | 10.00                    | 2023-01-01             | 10.00               |
| 3                     | 1002                  | 30.00                    | 2023-01-02             | 40.00               |
| 2                     | 1002                  | 20.00                    | 2023-01-03             | 60.00               |
| 8                     | 1003                  | 50.00                    | 2023-01-01             | 50.00               |
| 7                     | 1003                  | 50.00                    | 2023-01-03             | 100.00              |
| 9                     | 1003                  | 75.00                    | 2023-01-04             | 175.00              |
+-----------------------+-----------------------+--------------------------+------------------------+---------------------+
```

Оконные функции

LEAD и LAG

Функции `LEAD` / `LAG` позволяют извлекать данные из следующей/предыдущей строки соответственно. Такие функции полезны для сравнения значений соседних строк или для вычисления разницы между строками.

Следующий запрос группирует результаты выборки по аккаунтам и для каждой строки отображает ID следующей транзакции, относящейся к данной группе.

```
SELECT txn_id, txn_amount, txn_date, LEAD(txn_id)
OVER (PARTITION BY acc_id ORDER BY txn_date) AS next_txn_id
FROM transactions;
```

Результат показан ниже. Значения `NULL` в столбце `next_txn_id` указывают на выход за пределы партиции, то есть в данной группе больше нет следующих строк.

```
+---------+-------------+-------------+--------------+
| txn_id  | txn_amount  |  txn_date   | next_txn_id  |
+---------+-------------+-------------+--------------+
| 4       | 100.50      | 2023-01-02  | 6            |
| 6       | 200.50      | 2023-01-03  | 5            |
| 5       | 150.50      | 2023-01-04  | NULL         |
| 1       | 10.00       | 2023-01-01  | 3            |
| 3       | 30.00       | 2023-01-02  | 2            |
| 2       | 20.00       | 2023-01-03  | NULL         |
| 8       | 50.00       | 2023-01-01  | 7            |
| 7       | 50.00       | 2023-01-03  | 9            |
| 9       | 75.00       | 2023-01-04  | NULL         |
+---------+-------------+-------------+--------------+
```

По умолчанию функции `LEAD` и `LAG` возвращают значения столбцов, которые находятся в 1 шаге от текущей строки, а также возвращают `NULL`, если достигнута граница партиции. Это поведение можно переопределить, используя следующую сигнатуру:

```
LEAD(col, i, j)
LAG(col, i, j)
```

Где:

- `col` — имя столбца, для которого необходимо получить следующее/предыдущее значение.
- `i` — шаг смещения. Количество строк до следующего/предыдущего результата.
- `j` — значение, возвращаемое вместо `NULL` при достижении границ партиции.

ROW_NUMBER

Функция `ROW_NUMBER()` присваивает уникальный порядковый номер для каждой строки в партиции в соответствии с предложениями `PARTITION BY` и `ORDER BY`.

В следующем примере запрос получает первые 2 транзакции, совершенные каждым аккаунтом.

```
SELECT * FROM
     (SELECT *, ROW_NUMBER()
      OVER (PARTITION BY acc_id ORDER BY txn_date) AS row_num
      FROM transactions) tx
WHERE tx.row_num < 3;
```

В итоге столбец `tx.row_num` отображает первую и вторую транзакции, совершенные каждым аккаунтом в хронологическом порядке:

```
+------------+------------+----------------+--------------+-------------+
| tx.txn_id  | tx.acc_id  | tx.txn_amount  | tx.txn_date  | tx.row_num  |
+------------+------------+----------------+--------------+-------------+
| 4          | 1001       | 100.50         | 2023-01-02   | 1           |
| 6          | 1001       | 200.50         | 2023-01-03   | 2           |
| 1          | 1002       | 10.00          | 2023-01-01   | 1           |
| 3          | 1002       | 30.00          | 2023-01-02   | 2           |
| 8          | 1003       | 50.00          | 2023-01-01   | 1           |
| 7          | 1003       | 50.00          | 2023-01-03   | 2           |
+------------+------------+----------------+--------------+-------------+
```

FIRST_VALUE и LAST_VALUE

Функции `FIRST_VALUE` / `LAST_VALUE` возвращают значение столбца из первой и последней строки партиции соответственно.

В следующем примере запрос возвращает наименьшее значение ID транзакций для каждой партиции аккаунтов.

```
SELECT txn_id, acc_id, FIRST_VALUE(txn_id)
OVER (PARTITION BY acc_id) lowest_id_per_acc
FROM transactions;
```

Результат:

```
+---------+---------+--------------------+
| txn_id  | acc_id  | lowest_id_per_acc  |
+---------+---------+--------------------+
| 4       | 1001    | 4                  |
| 5       | 1001    | 4                  |
| 6       | 1001    | 4                  |
| 1       | 1002    | 1                  |
| 2       | 1002    | 1                  |
| 3       | 1002    | 1                  |
| 7       | 1003    | 7                  |
| 8       | 1003    | 7                  |
| 9       | 1003    | 7                  |
+---------+---------+--------------------+
```

Обе функции `FIRST_VALUE` / `LAST_VALUE` принимают необязательный Boolean-параметр, который по умолчанию равен `false`. Если указать значение `true`, то при подсчете первой/последней строки Hive будет игнорировать строки со значениями столбцов `NULL`.

Аналитические функции

RANK и DENSE_RANK

Функция `RANK` присваивает ранг каждой строке в партиции. Если есть повторяющиеся значения, функция возвращает одинаковый ранг для таких строк, пропуская при этом следующий числовой ранг.

В следующем примере `RANK` вычисляет ранг каждой транзакции среди всех остальных транзакций (предложение `PARTITION BY` отсутствует). Для ранжирования транзакций по каждому отдельному аккаунту необходимо добавить предложение `PARTITION BY acc_id`.

```
SELECT txn_amount, acc_id, RANK()
OVER (ORDER BY txn_amount DESC) rank_global
-- ранжирование внутри аккаунта:
-- OVER (PARTITION BY acc_id ORDER BY txn_amount DESC) rank_per_acc
FROM transactions;
```

Результат:

```
+-------------+---------+--------------+
| txn_amount  | acc_id  | rank_global  |
+-------------+---------+--------------+
| 200.50      | 1001    | 1            |
| 150.50      | 1001    | 2            |
| 100.50      | 1001    | 3            |
| 75.00       | 1003    | 4            |
| 50.00       | 1003    | 5            |
| 50.00       | 1003    | 5            |
| 30.00       | 1002    | 7            |
| 20.00       | 1002    | 8            |
| 10.00       | 1002    | 9            |
+-------------+---------+--------------+
```

Обратите внимание, что в итоговом наборе результатов есть две транзакции с одинаковым рангом (`5`) в силу равных значений `txn_amount`. Поскольку этим двум строкам был присвоен одинаковый ранг `5`, ранг `6` отсутствует в наборе результатов, а следующий доступный ранг — `7`. Чтобы избежать подобных "пробелов" в последовательности рангов, используйте функцию `DENSE_RANK`. Если в предыдущем примере заменить `RANK` на `DENSE_RANK`, результат будет следующим:

```
+-------------+---------+--------------+
| txn_amount  | acc_id  | rank_global  |
+-------------+---------+--------------+
| 200.50      | 1001    | 1            |
| 150.50      | 1001    | 2            |
| 100.50      | 1001    | 3            |
| 75.00       | 1003    | 4            |
| 50.00       | 1003    | 5            |
| 50.00       | 1003    | 5            |
| 30.00       | 1002    | 6            |
| 20.00       | 1002    | 7            |
| 10.00       | 1002    | 8            |
+-------------+---------+--------------+
```

PERCENT_RANK

Функция `PERCENT_RANK` вычисляет ранг в процентном выражении (число от 0 до 1 включительно) для каждой строки в партиции. Указание `ORDER BY` в предложении `OVER` является обязательным. Функция добавляет к исходному набору результатов столбец с возрастающей числовой последовательностью, которая отражает позицию каждой строки внутри партиции. Первая строка в партиции всегда имеет значение `0`, а последняя строка всегда равна `1`.

Например, следующий запрос ранжирует транзакции, осуществленные с участием одного и того же аккаунта.

```
SELECT txn_id, acc_id, txn_amount, PERCENT_RANK()
OVER (PARTITION BY acc_id ORDER BY txn_amount) position_within_acc
FROM transactions;
```

Результат содержит столбец `position_within_acc`, который отражает позицию транзакции в партиции:

```
+---------+---------+-------------+-------------------+
| txn_id  | acc_id  | txn_amount  |position_within_acc|
+---------+---------+-------------+-------------------+
| 4       | 1001    | 100.50      | 0.0               |
| 5       | 1001    | 150.50      | 0.5               |
| 6       | 1001    | 200.50      | 1.0               |
| 1       | 1002    | 10.00       | 0.0               |
| 2       | 1002    | 20.00       | 0.5               |
| 3       | 1002    | 30.00       | 1.0               |
| 7       | 1003    | 50.00       | 0.0               |
| 8       | 1003    | 50.00       | 0.0               |
| 9       | 1003    | 75.00       | 1.0               |
+---------+---------+-------------+-------------------+
```

NTILE

Функция `NTILE(n)` разбивает отсортированный набор результатов на `n` примерно равных групп (buckets), а каждой строке присваивается число, указывающее, к какой группе принадлежит данная строка. Указание `ORDER BY` в предложении `OVER` является обязательным. С помощью этой универсальной функции можно вычислять такие статистические понятия, как медиана, процентиль, квартиль и так далее.

Следующий пример разбивает все транзакции на 4 группы (квартили) по их сумме и определяет, в какой квартиль попадает каждая транзакция. Обратите внимание, что партиция сортируется по столбцу `txn_amount` от наименьшего к большему, таким образом, самые маленькие транзакции попадают в первый квартиль.

```
SELECT txn_id, txn_amount, NTILE(4)
OVER (ORDER BY txn_amount) AS quartile
FROM transactions;
```

Результат:

```
+---------+-------------+-----------+
| txn_id  | txn_amount  | quartile  |
+---------+-------------+-----------+
| 1       | 10.00       | 1         |
| 2       | 20.00       | 1         |
| 3       | 30.00       | 1         |
| 7       | 50.00       | 2         |
| 8       | 50.00       | 2         |
| 9       | 75.00       | 3         |
| 4       | 100.50      | 3         |
| 5       | 150.50      | 4         |
| 6       | 200.50      | 4         |
+---------+-------------+-----------+
```

Материализованные представления в Hive

Константин Алпашкин

Содержание

Назад к содержанию

- Тестовая база данных
- Синтаксис
- Переписывание запросов
- Обновление материализованного представления
- Пример использования

Материализованное представление (Materialized View, MV) — это объект, хранящий результаты выполнения определенного запроса, которые в будущем используются вместо чтения данных из базы. Обычно MV используются для повышения производительности путем "кеширования" результатов больших запросов со сложными агрегациями, объединениями, фильтрами и так далее.

В данном разделе описаны основные операции с MV, а также примеры их использования. Более детальная информация о работе с MV в Hive доступна в документации Apache Hive.

Тестовая база данных

Для демонстрации примеров в данной статье используются Hive-таблицы `employees` и `departments`, структура которых описана ниже.

```
SELECT * FROM employees;
+---------------+-----------------+----------------------+----------------------------+----------------+-------------------+-----
---------------+
| employees.id  | employees.name  | employees.last_name  |      employees.email       | employees.age  | employees.salary  |
employees.dept_id  |
+---------------+-----------------+----------------------+----------------------------+----------------+-------------------+-----
---------------+
| 1             | Ivan            | Ivanov               | ivan_ivanov123@mail.ru     | 30             | 1000              | 1
|
| 2             | Sarah           | Connor               | sarah_connor123@yahoo.com  | 35             | 1500              | 2
|
| 3             | Rick            | Sanchez              | rick_123@mail.ru           | 29             | 1300              | 2
|
| 4             | John            | Smith                | john_smith123@gmail.com    | 29             | 2000              | 3
|
+---------------+-----------------+----------------------+----------------------------+----------------+-------------------+-----
---------------+

SELECT * FROM departments;
+-----------------+-----------------------+-------------------------+------------------------+
| departments.id  | departments.dep_name  | departments.dep_office  | departments.dep_phone  |
+-----------------+-----------------------+-------------------------+------------------------+
| 1               | sales                 | 110                     | 9379992                |
| 2               | it                    | 115                     | 12345678               |
| 3               | support               | 125                     | 880080080              |
+-----------------+-----------------------+-------------------------+------------------------+
```

SQL для создания тестовых таблиц

Для создания и наполнения тестовых таблиц Hive выполните следующий SQL с помощью */bin/beeline*.

```sql
CREATE DATABASE IF NOT EXISTS mv_demo;
USE mv_demo;

DROP TABLE IF EXISTS employees;
CREATE TABLE employees (id int, name string, last_name string, email string, age int, salary int, dept_id int)
 STORED AS ORC
 TBLPROPERTIES ('transactional'='true');

INSERT INTO employees VALUES
(1, 'Ivan', 'Ivanov', 'ivan_ivanov123@mail.ru', 30, 1000, 1),
(2, 'Sarah', 'Connor', 'sarah_connor123@yahoo.com', 35, 1500, 2),
(3, 'Rick', 'Sanchez', 'rick_123@mail.ru', 29, 1300, 2),
(4, 'John', 'Smith', 'john_smith123@gmail.com', 29, 2000, 3);

DROP TABLE IF EXISTS departments;
CREATE TABLE departments (id int, dep_name string, dep_office int, dep_phone string)
STORED AS ORC
TBLPROPERTIES ('transactional'='true');

INSERT INTO departments VALUES
(1, 'sales', 110, '9379992'),
(2, 'it', 115, '12345678'),
(3, 'support', 125, '880080080');
```

Синтаксис

Поскольку технически MV — это обычная таблица Hive (хотя и с некоторыми ограничениями), синтаксис для создания нового MV аналогичен синтаксису создания таблиц Hive. Для создания MV используется следующий синтаксис:

```sql
CREATE MATERIALIZED VIEW [IF NOT EXISTS] [<db_name>.]<mv_name>
  [DISABLE REWRITE]
  [COMMENT <comment>]
  [PARTITIONED ON (<col_name>, ...)]
  [CLUSTERED ON (<col_name>, ...) | DISTRIBUTED ON (<col_name>, ...) SORTED ON (<col_name>, ...)]
  [
    [ROW FORMAT <row_format>]
    [STORED AS <file_format>] | STORED BY '<storage.handler.class.name>' [WITH SERDEPROPERTIES (...)]
  ]
  [LOCATION <hdfs_path>]
  [TBLPROPERTIES (<property_name>=<property_value>, ...)]
AS
<query>;
```

Где:

- `<query>` — запрос для наполнения MV данными.
- `DISABLE REWRITE` — отключает автоматическое переписывание запросов для данного MV.
- `PARTITIONED ON` — позволяет дробить MV на партиции по столбцу для лучшей производительности.
- `CLUSTERED ON` — используется для разделения MV на примерно равные бакеты, что облегчает обслуживание и увеличивает скорость работы. Каждый бакет хранится в HDFS в виде отдельного файла.
- `ROW FORMAT` — определяет форматирование строк, используя либо разделители, либо кастомный SerDe-класс.
- `STORED AS` — устанавливает формат хранения данных. По умолчанию используется формат ORC.
- `STORED BY` — указывает имя класса-обработчика для сохранения данных MV в ненативном формате.
- `LOCATION` — указывает путь в HDFS для сохранения данных MV.
- `TBLPROPERTIES` — список пар ключ/значение для настройки параметров MV.

Переписывание запросов

После создания MV оптимизатор Hive автоматически переписывает входящие запросы таким образом, чтобы задействовать MV, если посчитает, что это улучшит производительность. Механизм переписывания запросов включен по умолчанию и может быть отключен при создании MV (предложение `DISABLE REWRITE`) или с помощью следующей команды:

```sql
ALTER MATERIALIZED VIEW <mv_name> DISABLE REWRITE;
```

Для отключения функции переписывания запросов на глобальном уровне установите следующее свойство:

```sql
SET hive.materializedview.rewriting=false;
```

Если MV устаревает (содержимое MV не обновляется в течение длительного времени), Hive перестает использовать этот MV для переписывания запросов. Время, по истечении которого MV считается устаревшим, можно задать с помощью свойства `hive.materializedview.rewriting.time.window"="Nmin"` в разделе ***Custom hive-site.xml*** настроек ADCM.

ВАЖНО

Функция автоматического переписывания запросов недоступна, если MV был создан из нетранзакционной таблицы. Исходные таблицы Hive должны быть managed и ACID-таблицами.

Обновление материализованного представления

После создания MV любые изменения в исходных таблицах автоматически не отражаются в MV. Процесс обновления MV необходимо запускать вручную с помощью команды:

```sql
ALTER MATERIALIZED VIEW <mv_name> REBUILD;
```

Если новые данные были добавлены в исходные таблицы с помощью `INSERT`, Hive выполняет инкрементное обновление. Для операций `UPDATE` и `DELETE` выполняется полное обновление.

Пример использования

Далее показаны основные операции с MV:

1. Создайте новый MV, используя тестовые таблицы.

```sql
CREATE MATERIALIZED VIEW mv_emp_dp
  AS SELECT e.name, e.last_name, e.email, d.dep_name
  FROM employees e JOIN departments d
  ON (e.dept_id = d.id);
```

2. Проверьте, что MV был успешно создан:

```sql
SHOW MATERIALIZED VIEWS;
```

Вывод:

```
+------------+
|  tab_name  |
+------------+
| mv_emp_dp  |
+------------+
```

Также проверьте директорию хранилища HDFS на наличие новых файлов:

```
$ hdfs dfs -ls /apps/hive/warehouse/mv_demo.db/mv_emp_dp
```

В хранилище Hive доступен новый ORC-файл с данными MV:

```
-rw-r--r--   3 hive hadoop        795 2023-11-20 13:23 /apps/hive/warehouse/mv_demo.db/mv_emp_dp/000000_0
```

3. Выполните запрос, аналогичный тому, что использовался для создания MV. Такой запрос получает результаты непосредственно из MV вместо соединения и сканирования двух исходных таблиц.

```sql
SELECT e.last_name, e.email, d.dep_name
FROM employees e JOIN departments d
ON (e.dept_id = d.id)
WHERE d.dep_name='sales';
```

Результат:

```
+--------------+-------------------------+-------------+
| e.last_name  |         e.email         | d.dep_name  |
+--------------+-------------------------+-------------+
| Ivanov       | ivan_ivanov123@mail.ru  | sales       |
+--------------+-------------------------+-------------++
```

4. Выполните тот же запрос с предложением `EXPLAIN EXTENDED` для проверки переписывания запроса.

```sql
EXPLAIN EXTENDED SELECT e.last_name, e.email, d.dep_name
FROM employees e JOIN departments d
ON (e.dept_id = d.id)
WHERE d.dep_name='sales';
```

Вывод показан ниже. Обратите внимание на `alias: mv_demo.mv_emp_dp` — данная запись означает, что результаты были получены из MV `mv_demo.mv_emp_dp` одним сканированием.

```
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
| STAGE DEPENDENCIES:                                |
|   Stage-0 is a root stage                          |
|                                                    |
| STAGE PLANS:                                       |
|   Stage: Stage-0                                   |
|     Fetch Operator                                 |
|       limit: -1                                    |
|       Processor Tree:                              |
|         TableScan                                  |
|           alias: mv_demo.mv_emp_dp                 |
|           GatherStats: false                       |
|           Filter Operator                          |
|             isSamplingPred: false                  |
|             predicate: (dep_name = 'sales') (ty... |
|             Select Operator                        |
|               expressions: last_name (type: str... |
|               outputColumnNames: _col0, _col1, ... |
|               ListSink                             |
|                                                    |
+----------------------------------------------------+
```

5. Обновите исходную таблицу.

```sql
INSERT INTO employees VALUES
(5,'Anna','Petrova','ann_pett@yandxed.ru', 32, 1700, 1);
```

6. Еще раз выполните запрос с `EXPLAIN EXTENDED`:

```sql
EXPLAIN EXTENDED SELECT e.last_name, e.email, d.dep_name
FROM employees e JOIN departments d
ON (e.dept_id = d.id)
WHERE d.dep_name='sales';
```

На этот раз план выполнения содержит этапы объединения и полного сканирования таблиц, так как содержимое `mv_emp_dp` MV отличается от исходных таблиц.

```
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
| STAGE DEPENDENCIES:                                |
|   Stage-1 is a root stage                          |
|   Stage-0 depends on stages: Stage-1               |
|                                                    |
| STAGE PLANS:                                       |
|   Stage: Stage-1                                   |
|     Tez                                            |
|       DagId: hive_20231120144209_2c92e022-a865-... |
|       Edges:                                       |
|         Map 1 <- Map 2 (BROADCAST_EDGE)            |
|       DagName: hive_20231120144209_2c92e022-a86... |
|       Vertices:                                    |
|         Map 1                                      |
|             Map Operator Tree:                     |
|                 TableScan                          |
|                   alias: e                         |
|                   Statistics: Num rows: 50 Dat ... |
|                   ...                              |
+----------------------------------------------------+
```

7. Чтобы синхронизировать изменения в исходных таблицах и MV, перестройте MV:

```sql
ALTER MATERIALIZED VIEW mv_emp_dp REBUILD;
```

8. Выполните запрос с `EXPLAIN EXTENDED`:

```sql
EXPLAIN EXTENDED SELECT e.last_name, e.email, d.dep_name
FROM employees e JOIN departments d
ON (e.dept_id = d.id)
WHERE d.dep_name='sales';
```

Теперь в плане выполнения содержится информация об использовании MV для получения данных.

```
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
| STAGE DEPENDENCIES:                                |
|   Stage-0 is a root stage                          |
|                                                    |
| STAGE PLANS:                                       |
|   Stage: Stage-0                                   |
|     Fetch Operator                                 |
|       limit: -1                                    |
|       Processor Tree:                              |
|         TableScan                                  |
|           alias: mv_demo.mv_emp_dp                 |
|           GatherStats: false                       |
|           Filter Operator                          |
|             isSamplingPred: false                  |
|             predicate: (dep_name = 'sales') (ty... |
|             Select Operator                        |
|               expressions: last_name (type: str... |
|               outputColumnNames: _col0, _col1, ... |
|               ListSink                             |
|                                                    |
+----------------------------------------------------+
```

9. Для удаления MV используйте команду:

```sql
DROP MATERIALIZED VIEW mv_emp_dp;
```

Hive on Spark

Константин Алпашкин

Содержание

Назад к содержанию

Настройка

Hive on Spark позволяет Hive использовать Apache Spark в качестве основного механизма выполнения. По умолчанию Hive использует Tez в качестве основного механизма выполнения.

Настройка

Наиболее простой способ активировать движок Spark — использовать ADCM.

Для этого на странице ***Hive Configuration*** необходимо изменить значение параметра `hive.execution.engine`. Также этот параметр можно изменить в файле *hive-site.xml*, который находится в директории с настройками Hive по умолчанию: */etc/hive/conf/*. Параметр `hive.execution.engine` представлен ниже.

```
<property>
    <name>hive.execution.engine</name>
    <value>spark</value>
</property>
```

Более детальную информацию о настройке Spark вы можете увидеть на странице Spark и Hive.

Домашняя директория Spark содержит конфигурационный файл Hive (*/etc/spark3/conf/hive-site.xml*). Вы можете изменять или добавлять параметры в этот файл, чтобы кастомизировать работу вашего Spark-приложения и Hive.

Файл настроек по умолчанию

```
<?xml version="1.0"?>
<configuration>
        <property>
                <name>hive.cbo.enable</name>
                <value>True</value>
        </property>
        <property>
                <name>hive.compute.query.using.stats</name>
                <value>False</value>
        </property>
        <property>
                <name>hive.execution.engine</name>
                <value>spark</value>
        </property>
        <property>
                <name>hive.log.explain.output</name>
                <value>True</value>
        </property>
        <property>
                <name>hive.metastore.event.db.notification.api.auth</name>
                <value>False</value>
        </property>
        <property>
                <name>hive.metastore.kerberos.keytab.file</name>
                <value>/etc/security/keytabs/hive.service.keytab</value>
        </property>
        <property>
                <name>hive.metastore.kerberos.principal</name>
                <value>hive/_HOST@EXAMPLE.COM</value>
        </property>
        <property>
                <name>hive.metastore.sasl.enabled</name>
                <value>False</value>
        </property>
        <property>
                <name>hive.metastore.uris</name>
                <value>thrift://aai-adh.ru-central1.internal:9083</value>
        </property>
        <property>
                <name>hive.metastore.warehouse.dir</name>
                <value>/apps/hive/warehouse</value>
        </property>
        <property>
                <name>hive.server2.authentication</name>
                <value>NONE</value>
        </property>
        <property>
                <name>hive.server2.authentication.kerberos.keytab</name>
                <value>/etc/security/keytabs/hive.service.keytab</value>
        </property>
        <property>
                <name>hive.server2.authentication.kerberos.principal</name>
                <value>hive/_HOST@EXAMPLE.COM</value>
        </property>
        <property>
                <name>hive.server2.authentication.spnego.keytab</name>
                <value>/etc/security/keytabs/HTTP.service.keytab</value>
        </property>
        <property>
                <name>hive.server2.authentication.spnego.principal</name>
                <value>HTTP/_HOST@EXAMPLE.COM</value>
        </property>
        <property>
                <name>hive.server2.enable.doAs</name>
                <value>False</value>
        </property>
        <property>
                <name>hive.stats.fetch.column.stats</name>
                <value>False</value>
        </property>
        <property>
                <name>hive.support.concurrency</name>
                <value>False</value>
        </property>
        <property>
                <name>hive.tez.container.size</name>
                <value>1024</value>
        </property>
        <property>
                <name>hive.txn.manager</name>
                <value>org.apache.hadoop.hive.ql.lockmgr.DummyTxnManager</value>
        </property>
        <property>
                <name>javax.jdo.option.ConnectionDriverName</name>
                <value>com.mysql.jdbc.Driver</value>
        </property>
        <property>
                <name>javax.jdo.option.ConnectionPassword</name>
                <value>admin</value>
        </property>
        <property>
                <name>javax.jdo.option.ConnectionURL</name>
                <value>jdbc:mysql://aai-adh3.ru-central1.internal:3306/hive</value>
        </property>
        <property>
                <name>javax.jdo.option.ConnectionUserName</name>
                <value>APP</value>
        </property>
</configuration>
```

Hive и HBase

Евгения Кузина

Содержание

Назад к содержанию

Создание новой таблицы HBase в Hive

Работа с существующей таблицей HBase в Hive

Интеграция Hive/HBase позволяет операторам Hive QL получать доступ к таблицам HBase как для операций чтения (`SELECT`), так и для операций записи (`INSERT`). Вы можете создать новую таблицу HBase в Hive или получить доступ к существующей таблице.

Создание новой таблицы HBase в Hive

Чтобы создать новую таблицу HBase, которой можно управлять через Hive, используйте выражение `STORED BY` в инструкции `CREATE TABLE`. Оно позволяет указать обработчик хранилища (storage handler), необходимый для интеграции.

Пример:

```
$ sudo -u hive hive
```

```
CREATE TABLE hbase_table_test(key int, value1 string, value2 int, value3 int)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,col_family1:val1,col_family1:val2,col_family2:val3")
TBLPROPERTIES ("hbase.table.name" = "hive_test", "hbase.mapred.output.outputtable" = "hive_test");
```

Где:

- `STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'` — указывает обработчик хранилища, который позволяет Hive получать доступ к данным HBase.
- `WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,col_family1:val1,col_family1:val2,col_family2:val3")` — определяет соответствие между столбцами HBase и Hive. Параметр `hbase.columns.mapping` может содержать следующие записи: `:key`, `:timestamp` или `<column-family-name>:[<column-name>][#(binary|string)]`. Спецификация типа отделяется символом `#`. Если `hbase.columns.mapping` не включает спецификацию типа, используется значение из `hbase.table.default.storage.type`.
- `TBLPROPERTIES ("hbase.table.name" = "hive_test", "hbase.mapred.output.outputtable" = "hive_test")` — определяет свойства таблицы. Свойство `hbase.table.name` указывает имя таблицы в HBase. В этом примере имя таблицы в Hive — `hbase_table_test`, а в HBase — `hive_test`. Если `hbase.table.name` не определено, у таблицы будет одно и то же имя и в Hive, и в HBase. Свойство `hbase.mapred.output.outputtable` необходимо указать, если вы планируете вставлять данные в таблицу (используется классом hbase.mapreduce.TableOutputFormat).

Выполните команду `list`, чтобы проверить, есть ли в HBase информация о новой таблице:

```
$ hbase shell

$ list
```

Вывод команды `list` должен содержать таблицу `hive_test`.

В командной строке HBase используйте команду `put`, чтобы добавить данные в таблицу:

```
put 'hive_test', '1' , 'col_family1:val1', 'test'
put 'hive_test', '1' , 'col_family1:val2', '1'
put 'hive_test', '1' , 'col_family2:val3', '1'

put 'hive_test', '2' , 'col_family1:val1', 'test2'
put 'hive_test', '2' , 'col_family1:val2', '2'
put 'hive_test', '2' , 'col_family2:val3', '2'
```

Проверьте содержимое таблицы, выполнив команду `scan`:

```
scan 'hive_test'
```

Результат:

```
ROW                                                           COLUMN+CELL
1                                                            column=col_family1:val1, timestamp=1694606917613, value=test
1                                                            column=col_family1:val2, timestamp=1694606917647, value=1
1                                                            column=col_family2:val3, timestamp=1694606922996, value=1
2                                                            column=col_family1:val1, timestamp=1694607054746, value=test2
2                                                            column=col_family1:val2, timestamp=1694607054763, value=2
2                                                            column=col_family2:val3, timestamp=1694607056127, value=2
2 row(s)
Took 0.0175 seconds
```

HBase успешно добавила данные в таблицу, созданную в Hive.

Вставьте новую строку в ту же таблицу через командную строку Hive:

```
INSERT INTO hbase_table_test VALUES(3,'test3', 3, 3);
```

Проверьте данные таблицы в оболочке HBase:

```
scan 'hive_test'
```

Результат:

```
ROW                   COLUMN+CELL
 1                    column=col_family1:val1, timestamp=1698334191476, value=test
 1                    column=col_family1:val2, timestamp=1698334191504, value=1
 1                    column=col_family2:val3, timestamp=1698334191538, value=1
 2                    column=col_family1:val1, timestamp=1698334191591, value=test2
 2                    column=col_family1:val2, timestamp=1698334191616, value=2
 2                    column=col_family2:val3, timestamp=1698334193906, value=2
 3                    column=col_family1:val1, timestamp=1698334255870, value=test3
 3                    column=col_family1:val2, timestamp=1698334255870, value=3
 3                    column=col_family2:val3, timestamp=1698334255870, value=3
3 row(s)
Took 0.0133 seconds
```

Как показано в этом примере, Hive и HBase могут работать с одной и той же таблицей.

Работа с существующей таблицей HBase в Hive

Вы также можете работать с существующей таблицей HBase в Hive. В примере ниже используется таблица `people_ages` из статьи Массовая загрузка с использованием встроенных задач MapReduce.

Для существующей таблицы HBase необходимо создать внешнюю таблицу в командной строке Hive и указать другие свойства в соответствии со структурой таблицы:

```
CREATE EXTERNAL TABLE hbase_people_ages  (name string, age int)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,basic:age")
TBLPROPERTIES ("hbase.table.name" = "people_ages", "hbase.mapred.output.outputtable" = "people_ages");
```

Сделайте выборку данных, чтобы проверить, доступна ли таблица:

```
SELECT * FROM hbase_people_ages LIMIT 5;
```

Результат:

```
OK
Abbott Delia    62
Abbott Howard   24
Abbott Jack     29
Adams Clyde     29
Aguilar Myrtie  23
Time taken: 0.161 seconds, Fetched: 5 row(s)
```

Также можно объединить таблицы HBase с таблицами Hive с помощью операторов `JOIN` и `UNION`.

Создайте еще одну таблицу в Hive:

```
CREATE TABLE people (key int, name string, points int);

INSERT INTO people
    VALUES (1,'Phelps Rena', 35), (2,'Quinn Brian', 48), (1,'Reyes Danny', 52);
```

Выполните запрос с выражением `JOIN`:

```
SELECT hbase_people_ages.*, people.points FROM hbase_people_ages JOIN people ON (hbase_people_ages.name = people.name);
```

Результат:

```
OK
Phelps Rena     18      35
Quinn Brian     20      48
Reyes Danny     41      52
Time taken: 10.003 seconds, Fetched: 3 row(s)
```

Просмотр задач Hive в TEZ

Константин Алпашкин

Содержание

Назад к содержанию

- Статистика задач
  - DAG Details
  - DAG Counters
  - Graphical View
  - All Vertices
  - All Tasks
  - All Tasks Attempts
  - Vertex Swimlane

Все задачи (jobs), которые вы запускаете в Hive, доступны в Tez-интерфейсе. Ниже описаны шаги использования интерфейса Tez:

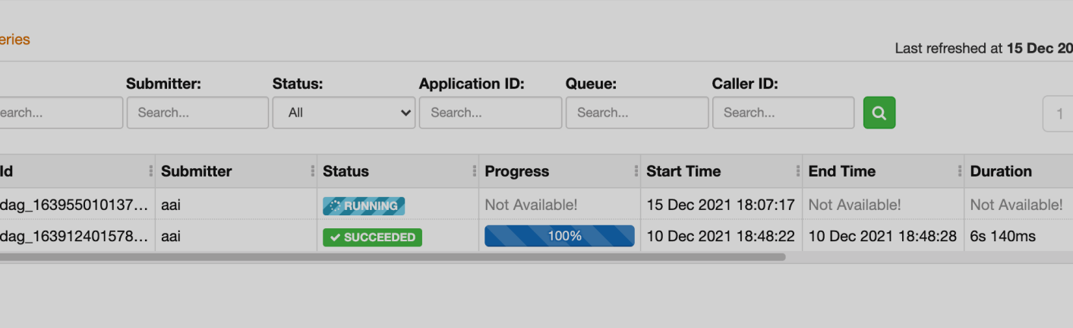
1. Откройте страницу интерфейса Tez. Ссылку на страницу вы можете найти в разделе Обзор работы TEZ.
2. Запустите Hive shell на хосте, где установлен компонент Hive HiveServer2. Информация об использовании оболочки Hive доступна на странице Работа с таблицами Hive.
3. Создайте таблицу, используя оболочку Hive:

```
CREATE TABLE customers
( customer_id int NOT NULL,
  customer_name char(50) NOT NULL,
  address char(50),
  city char(50),
  state char(25),
  zip_code char(10)
);
```

4. Вставьте данные в таблицу:

```
INSERT INTO TABLE customers VALUES
(1,'John Smith','Holliwood blvd','Los Angeles','CA','435267'),
(2,'Alice Brown','42 st','New York','NY','876234'),
(3,'Maria Jensen','Paris drv','Chicago','IL','293752');
```

5. Используйте интерфейс Tez, чтобы отследить процесс добавления данных:



Home / All DAGs
Version 0.9.2
All DAGs
Hive Queries
Last refreshed at 15 Dec 2021 18:07:22
Refresh
DAG Name:
ID:
Submitter:
Status:
Application ID:
Queue:
Caller ID:
10 Rows
Load Counters

Dag Name	Id	Submitter	Status	Progress	Start Time	End Time	Duration	Application Id
INSERT INTO TABL...	dag_163955010137...	aai	RUNNING	Not Available!	15 Dec 2021 18:07:17	Not Available!	Not Available!	application_1639
INSERT INTO TABL...	dag_163912401578...	aai	SUCCEEDED	100%	10 Dec 2021 18:48:22	10 Dec 2021 18:48:28	6s 140ms	application_1639

Добавление данных

6. Проверьте добавленные записи, используя следующий запрос:

```
SELECT *
FROM customers
WHERE state = 'CA'
ORDER BY customer_id ASC;
```

Вывод имеет следующий вид:

```
Query ID = aai_20211215195428_e60e1dc5-f6f3-4c5e-896a-6e56407845c1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1639550101374_0007)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 5,38 s
----------------------------------------------------------------------------------------------
OK
1       John Smith                                              Holliwood blvd                                              Los
Angeles                                         CA                              435267
4       Peter Stormik                                           International blv                                           San
Francisco                                       CA                              231987
5       Daniel Morrowetz                                        Bond rd                                                     Elk Grove
CA                              390817
Time taken: 6.447 seconds, Fetched: 3 row(s)
```

Задачи `SELECT` и `INSERT` отображаются в интерфейсе Tez.

7. Выполните вычисления из раздела Пример вычислений Hive, чтобы увидеть данные о результате вычислений в интерфейсе Tez.

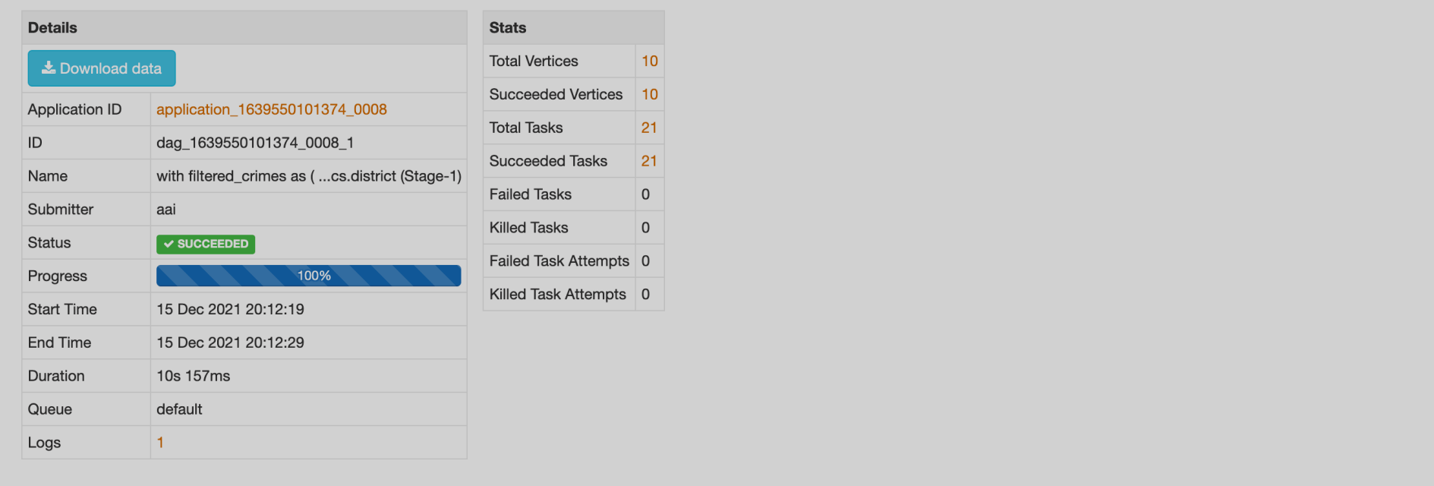
После выполнения задачи вся статистика доступна на странице задач.

Статистика задач

DAG Details

На этой вкладке вы можете увидеть подробную информацию о созданном ориентированном ациклическом графе (Directed Acyclic Graph, DAG). Вкладка ***Query Details*** содержит следующую информацию:

- ID Hive-запроса;
- пользователь, который выполнил запрос;
- время начала и окончания выполнения запроса;
- длительность выполнения запроса;
- таблицы, которые были задействованы в выполнении запроса;
- ID приложения;
- ID графа;
- ID сессии;
- ID потока;
- очередь, согласно которой был выполнен запрос.



Home / DAG [with filtered_crimes as (...cs.district (Stage-1)]
Version 0.9.2
DAG Details
DAG Counters
Graphical View
All Vertices
All Tasks
All Task Attempts
Vertex Swimlane
Auto Refresh
Last refreshed at 16 Dec 2021 07:26:14
Refresh
Download data

Details	
Application ID	application_1639550101374_0008
ID	dag_1639550101374_0008_1
Name	with filtered_crimes as (...cs.district (Stage-1)
Submitter	aai
Status	SUCCEEDED
Progress	100%
Start Time	15 Dec 2021 20:12:19
End Time	15 Dec 2021 20:12:29
Duration	10s 157ms
Queue	default
Logs	1

Stats	
Total Vertices	10
Succeeded Vertices	10
Total Tasks	21
Succeeded Tasks	21
Failed Tasks	0
Killed Tasks	0
Failed Task Attempts	0
Killed Task Attempts	0

Vertex Name	Status	Progress	Total Tasks	Succeeded Tasks	Running Tasks	Pending Tasks	Failed Task Atte...	Killed Task Atte..
Map 6	SUCCEEDED	100%	1	1	Not Available!	Not Available!	0	0
Reducer 3	SUCCEEDED	100%	3	3	Not Available!	Not Available!	0	0
Reducer 9	SUCCEEDED	100%	3	3	Not Available!	Not Available!	0	0

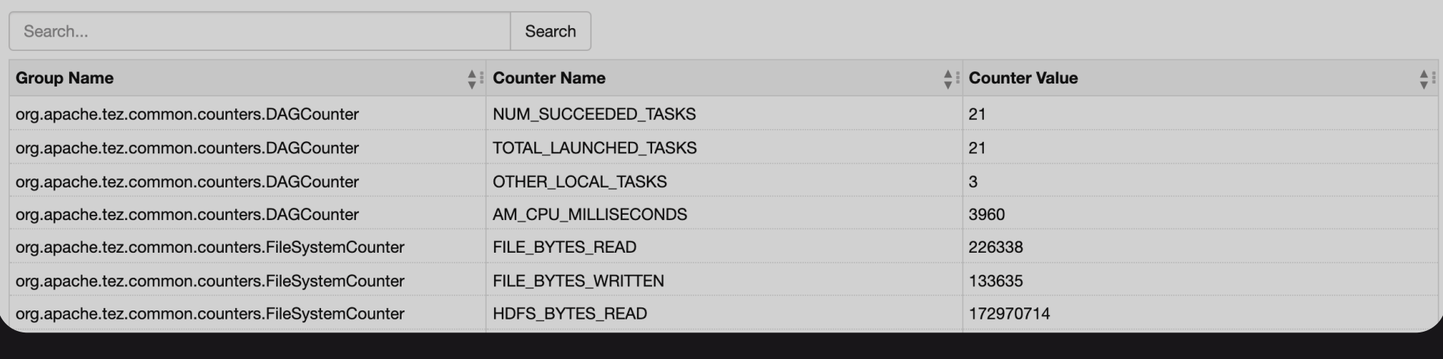
DAG Details

DAG Counters

DAG-счетчики (counters) позволяют отслеживать прогресс операций, выполняемых в контексте одного графа. Счетчики используются для сбора статистических данных с целью улучшения контроля качества или диагностирования ошибок.

DAG-счетчики содержат следующую информацию:

- имя группы;
- имя счетчика;
- значение счетчика.



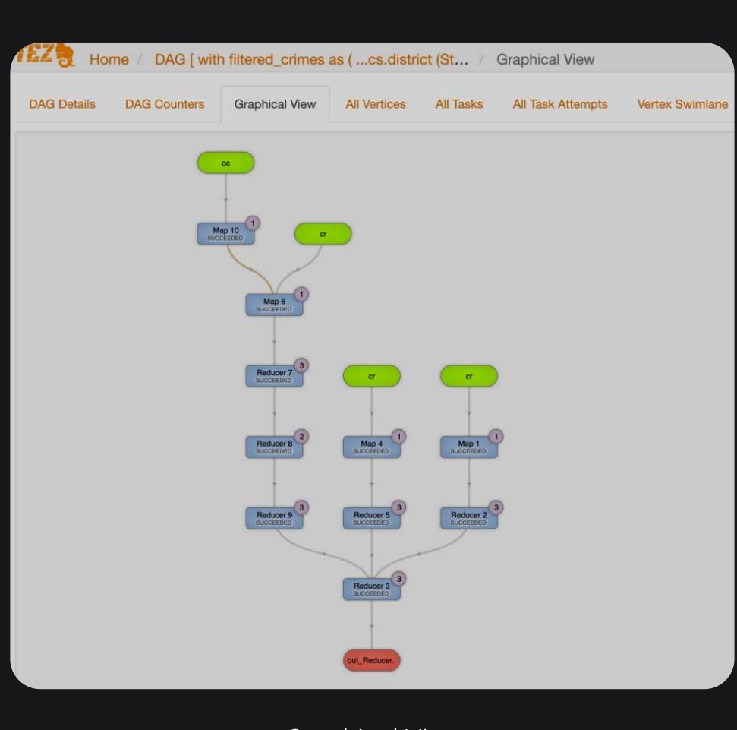
Home / DAG [with filtered_crimes as (...cs.district (... / DAG Counters
Version 0.9.2
DAG Details
DAG Counters
Graphical View
All Vertices
All Tasks
All Task Attempts
Vertex Swimlane
Auto Refresh
Last refreshed at 16 Dec 2021 07:26:14
Refresh
Search...
Search

Group Name	Counter Name	Counter Value
org.apache.tez.common.counters.DAGCounter	NUM_SUCCEEDED_TASKS	21
org.apache.tez.common.counters.DAGCounter	TOTAL_LAUNCHED_TASKS	21
org.apache.tez.common.counters.DAGCounter	OTHER_LOCAL_TASKS	3
org.apache.tez.common.counters.DAGCounter	AM_CPU_MILLISECONDS	3960
org.apache.tez.common.counters.FileSystemCounter	FILE_BYTES_READ	226338
org.apache.tez.common.counters.FileSystemCounter	FILE_BYTES_WRITTEN	133635
org.apache.tez.common.counters.FileSystemCounter	HDFS_BYTES_READ	172970714

DAG Counters

Graphical View

Вкладка ***Graphical View*** отображает граф, созданный Hive, в виде схемы:



Home / DAG [with filtered_crimes as (...cs.district (St... / Graphical View
DAG Details
DAG Counters
Graphical View
All Vertices
All Tasks
All Task Attempts
Vertex Swimlane
Map 10
Map 6
Reducer 7
Reducer 8
Map 4
Map 1
Reducer 9
Reducer 5
Reducer 2
Reducer 3
out_Reducer

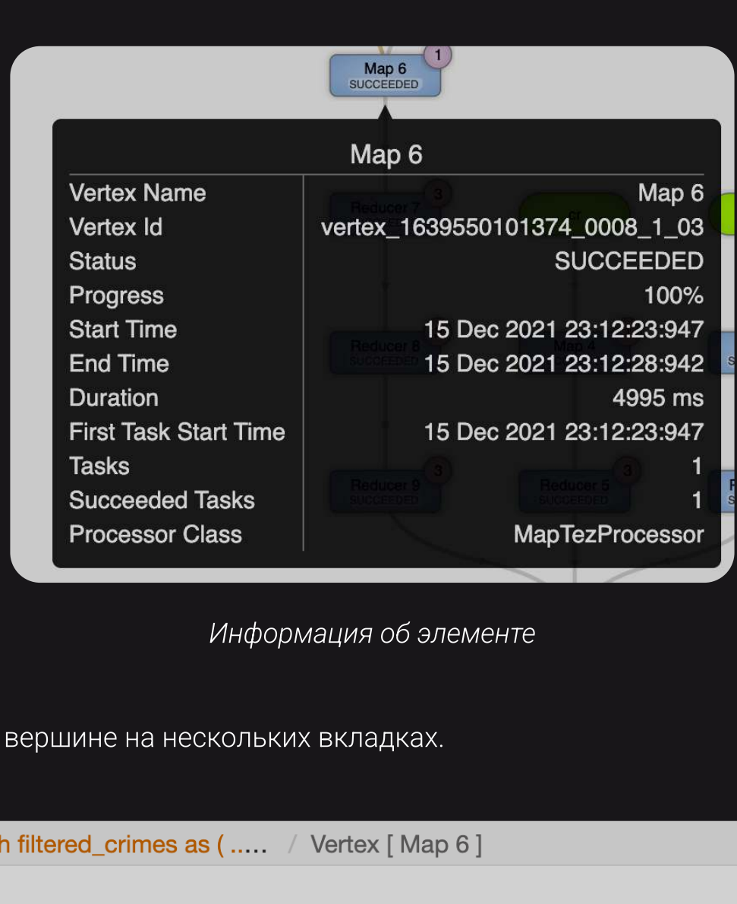
Graphical View

В данной схеме точки входа к вершинам `Map 10`, `Map 6`, `Map 4` и `Map 1` — это таблицы, отображаемые в виде зеленых прямоугольников. `Map 6` зависит от результатов, производимых функцией `Map 10`. Затем результат работы `Map 6` передается в виде входных данных на `Reducer 7`. После того, как `Reducer 7` отработал, результаты передаются на `Reducer 8`, а затем — на `Reducer 9`. `Reducer 3` является последней вершиной в графе.

После того, как `Reducer 3` успешно отработал, вывод запроса записывается в HDFS.

Есть несколько опций, чтобы изменить вид графа. Используя ***Toggle source/sink visibility***, вы можете скрыть входные и выходные узлы графа, так чтобы отображались лишь вершины с задачами. Чтобы изменить ориентацию, используйте кнопку ***Toggle orientation***.

При наведении мыши на элемент графа появляется детальная информация о вершинах, прогрессе и статусе, длительности и количестве задач, как показано на рисунке ниже.

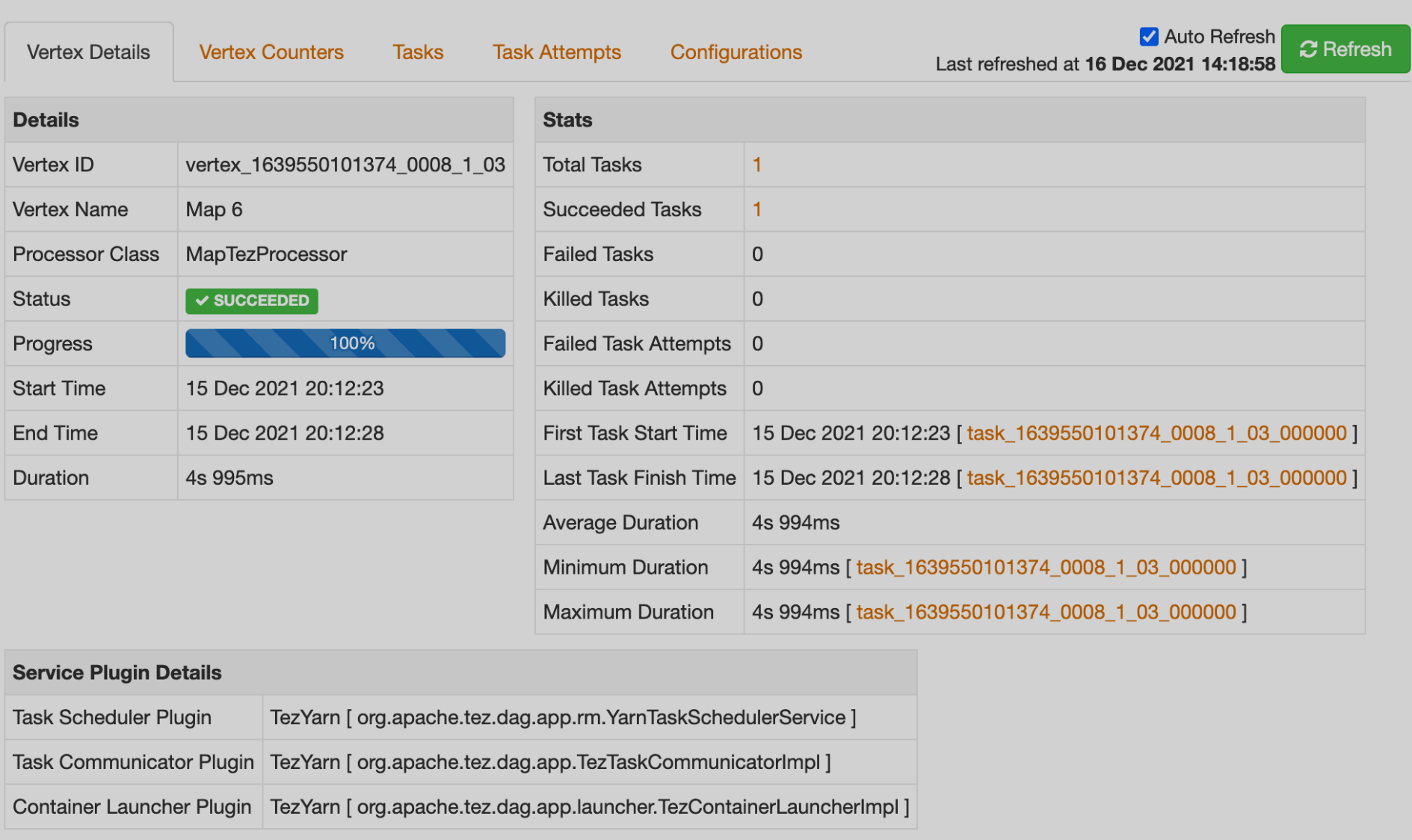


Map 6

Vertex Name	Map 6
Vertex Id	vertex_1639550101374_0008_1_03
Status	SUCCEEDED
Progress	100%
Start Time	15 Dec 2021 23:12:23:947
End Time	15 Dec 2021 23:12:28:942
Duration	4995 ms
First Task Start Time	15 Dec 2021 23:12:23:947
Tasks	1
Succeeded Tasks	1
Processor Class	MapTezProcessor

Информация об элементе

Кликнув на элемент, вы увидите подробную статистику о вершине на нескольких вкладках.



Home / DAG [with filtered_crimes as (..... / Vertex [Map 6]
Version 0.9.2
Vertex Details
Vertex Counters
Tasks
Task Attempts
Configurations
Auto Refresh
Last refreshed at 16 Dec 2021 14:18:58
Refresh

Details	
Vertex ID	vertex_1639550101374_0008_1_03
Vertex Name	Map 6
Processor Class	MapTezProcessor
Status	SUCCEEDED
Progress	100%
Start Time	15 Dec 2021 20:12:23
End Time	15 Dec 2021 20:12:28
Duration	4s 995ms

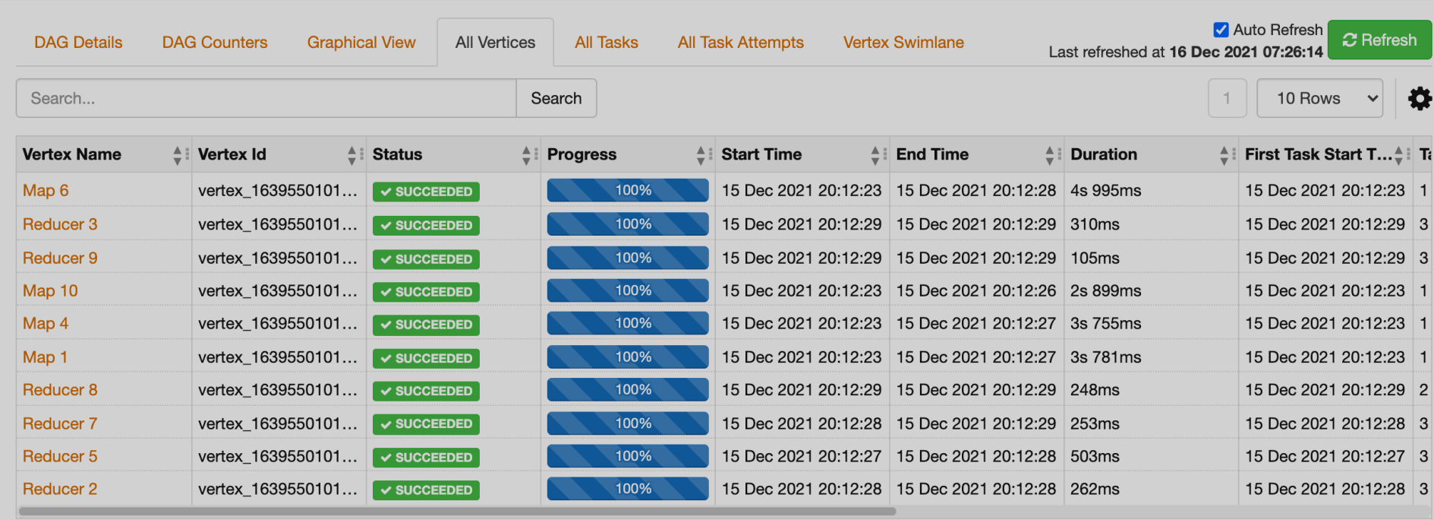
Stats	
Total Tasks	1
Succeeded Tasks	1
Failed Tasks	0
Killed Tasks	0
Failed Task Attempts	0
Killed Task Attempts	0
First Task Start Time	15 Dec 2021 20:12:23 [task_1639550101374_0008_1_03_000000]
Last Task Finish Time	15 Dec 2021 20:12:28 [task_1639550101374_0008_1_03_000000]
Average Duration	4s 994ms
Minimum Duration	4s 994ms [task_1639550101374_0008_1_03_000000]
Maximum Duration	4s 994ms [task_1639550101374_0008_1_03_000000]

Service Plugin Details	
Task Scheduler Plugin	TezYarn [org.apache.tez.dag.app.rm.YarnTaskSchedulerService]
Task Communicator Plugin	TezYarn [org.apache.tez.dag.app.TezTaskCommunicatorImpl]
Container Launcher Plugin	TezYarn [org.apache.tez.dag.app.launcher.TezContainerLauncherImpl]

Vertex details

All Vertices

На этой вкладке доступна вся информация о функциях Map и Reduce в виде таблицы. При клике на вершину, появится детальная информация о функциях Map и Reduce.



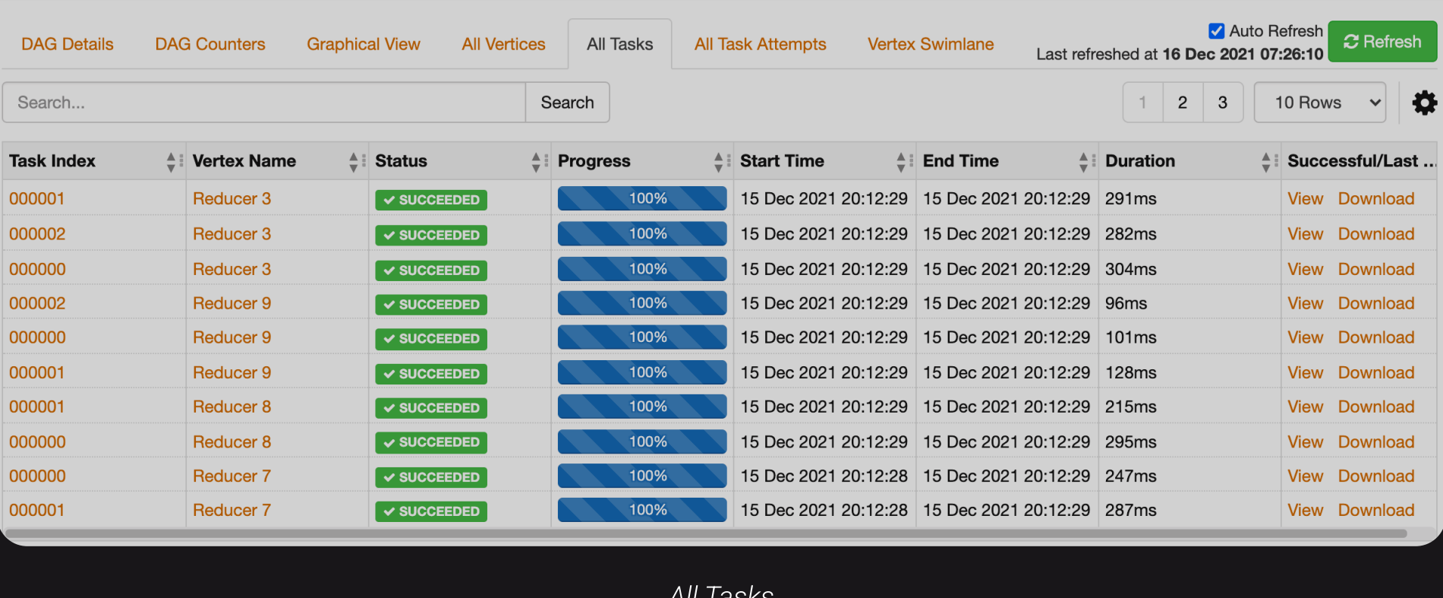
Home / DAG [with filtered_crimes as (...cs.district ... / All Vertices
Version 0.9.2
DAG Details
DAG Counters
Graphical View
All Vertices
All Tasks
All Task Attempts
Vertex Swimlane
Auto Refresh
Last refreshed at 16 Dec 2021 07:26:14
Refresh
Search...
Search
10 Rows

Vertex Name	Vertex Id	Status	Progress	Start Time	End Time	Duration	First Task Start T...
Map 6	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:23	15 Dec 2021 20:12:28	4s 995ms	15 Dec 2021 20:12:23
Reducer 3	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	310ms	15 Dec 2021 20:12:29
Reducer 9	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	105ms	15 Dec 2021 20:12:29
Map 10	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:23	15 Dec 2021 20:12:26	2s 899ms	15 Dec 2021 20:12:23
Map 4	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:23	15 Dec 2021 20:12:27	3s 755ms	15 Dec 2021 20:12:23
Map 1	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:23	15 Dec 2021 20:12:27	3s 781ms	15 Dec 2021 20:12:23
Reducer 8	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	248ms	15 Dec 2021 20:12:29
Reducer 7	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:28	15 Dec 2021 20:12:29	253ms	15 Dec 2021 20:12:28
Reducer 5	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:27	15 Dec 2021 20:12:28	503ms	15 Dec 2021 20:12:27
Reducer 2	vertex_1639550101...	SUCCEEDED	100%	15 Dec 2021 20:12:28	15 Dec 2021 20:12:28	262ms	15 Dec 2021 20:12:28

All Vertices

All Tasks

На этой вкладке представлена статистика о reducer-задачах. Используя ссылки, вы можете увидеть детальную информацию о задаче.



Home / DAG [with filtered_crimes as (...cs.distri... / All Tasks
Version 0.9.2
DAG Details
DAG Counters
Graphical View
All Vertices
All Tasks
All Task Attempts
Vertex Swimlane
Auto Refresh
Last refreshed at 16 Dec 2021 07:26:10
Refresh
Search...
Search
10 Rows

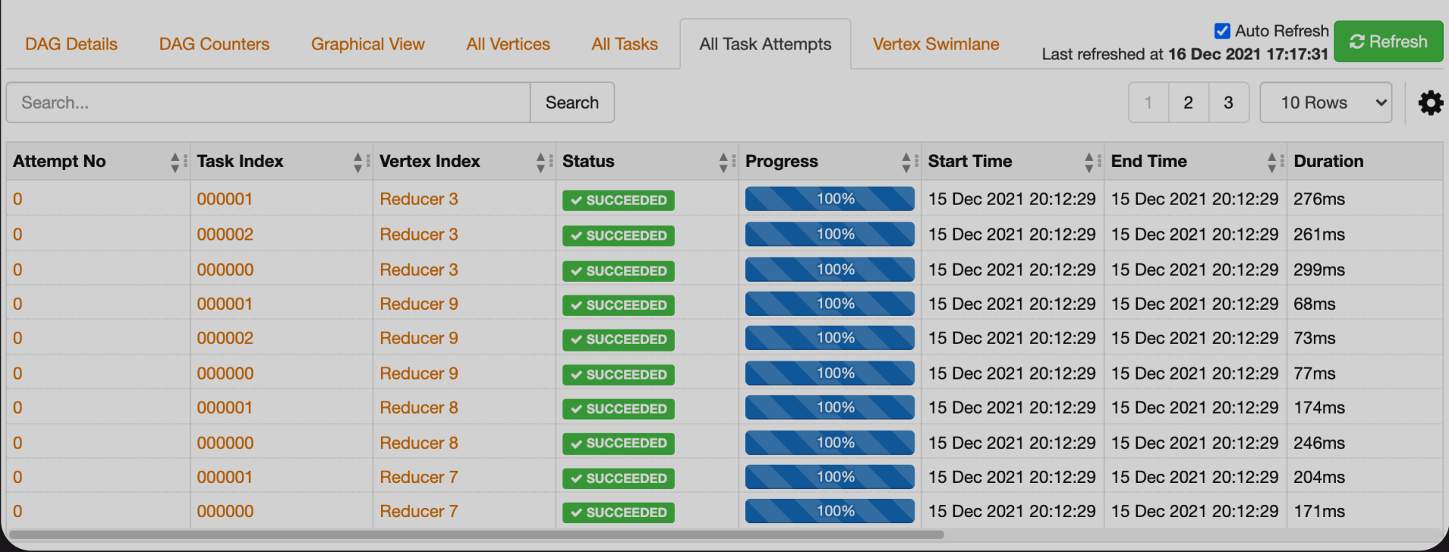
Task Index	Vertex Name	Status	Progress	Start Time	End Time	Duration	Successful/Last ...
000001	Reducer 3	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	291ms	View Download
000002	Reducer 3	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	282ms	View Download
000000	Reducer 3	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	304ms	View Download
000002	Reducer 9	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	96ms	View Download
000000	Reducer 9	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	101ms	View Download
000001	Reducer 9	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	128ms	View Download
000001	Reducer 8	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	215ms	View Download
000000	Reducer 8	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	295ms	View Download
000000	Reducer 7	SUCCEEDED	100%	15 Dec 2021 20:12:28	15 Dec 2021 20:12:29	247ms	View Download
000001	Reducer 7	SUCCEEDED	100%	15 Dec 2021 20:12:28	15 Dec 2021 20:12:29	287ms	View Download

All Tasks

All Tasks
image::how-to:tez/all-tasks_light.png[width=724, alt="Вкладка All Tasks"]

All Tasks Attempts

На этой вкладке доступна информация о всех попытках выполнения reducer-задач. Используя ссылки, вы можете увидеть детальную информацию о любой попытке, задаче или reducer.



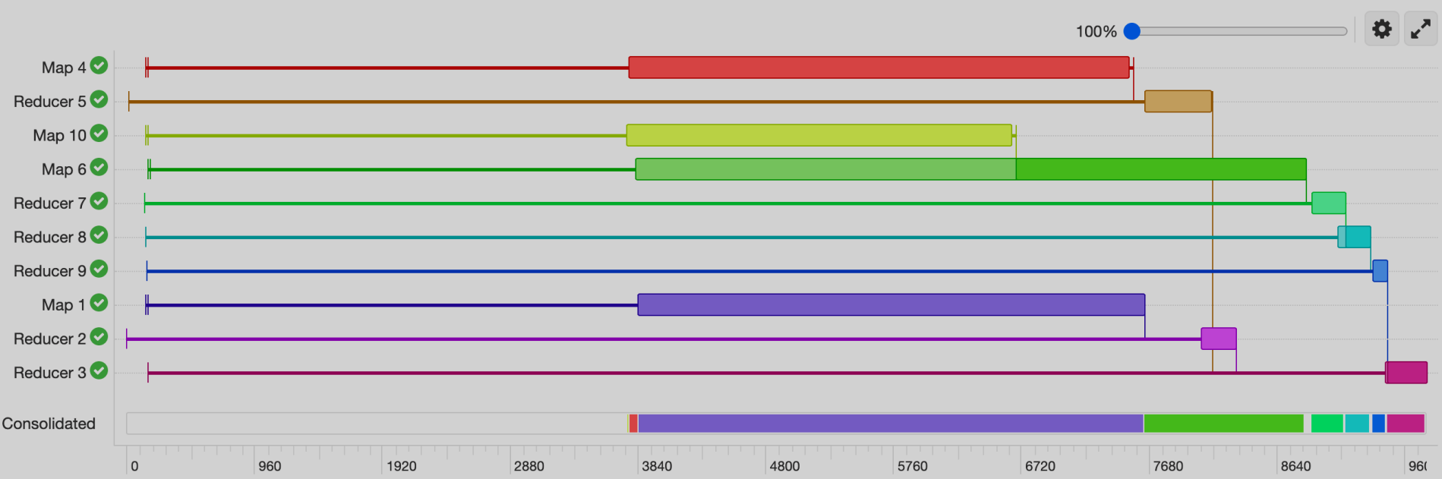
Home / DAG [with filtered_crimes as (...cs.distri... / All Task Attempts
Version 0.9.2
DAG Details
DAG Counters
Graphical View
All Vertices
All Tasks
All Task Attempts
Vertex Swimlane
Auto Refresh
Last refreshed at 16 Dec 2021 17:17:31
Refresh
Search...
Search
10 Rows

Attempt No	Task Index	Vertex Index	Status	Progress	Start Time	End Time	Duration
0	000001	Reducer 3	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	276ms
0	000002	Reducer 3	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	261ms
0	000000	Reducer 3	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	299ms
0	000001	Reducer 9	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	68ms
0	000002	Reducer 9	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	73ms
0	000000	Reducer 9	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	77ms
0	000001	Reducer 8	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	174ms
0	000000	Reducer 8	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	246ms
0	000001	Reducer 7	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	204ms
0	000000	Reducer 7	SUCCEEDED	100%	15 Dec 2021 20:12:29	15 Dec 2021 20:12:29	171ms

All Tasks Attempts

Vertex Swimlane

Эта вкладка отображает граф с учетом времени, где вершинами являются map/reduce-задачи.



Home / DAG [with filtered_crimes as (...cs.district (S... / Vertex Swimlane
Version 0.9.2
DAG Details
DAG Counters
Graphical View
All Vertices
All Tasks
All Task Attempts
Vertex Swimlane
Auto Refresh
Last refreshed at 16 Dec 2021 07:26:14
Refresh
100%
Map 4
Reducer 5
Map 10
Map 6
Reducer 7
Reducer 8
Reducer 9
Map 1
Reducer 2
Reducer 3
Consolidated
0
960
1920
2880
3840
4800
5760
6720
7680
8640
Milliseconds

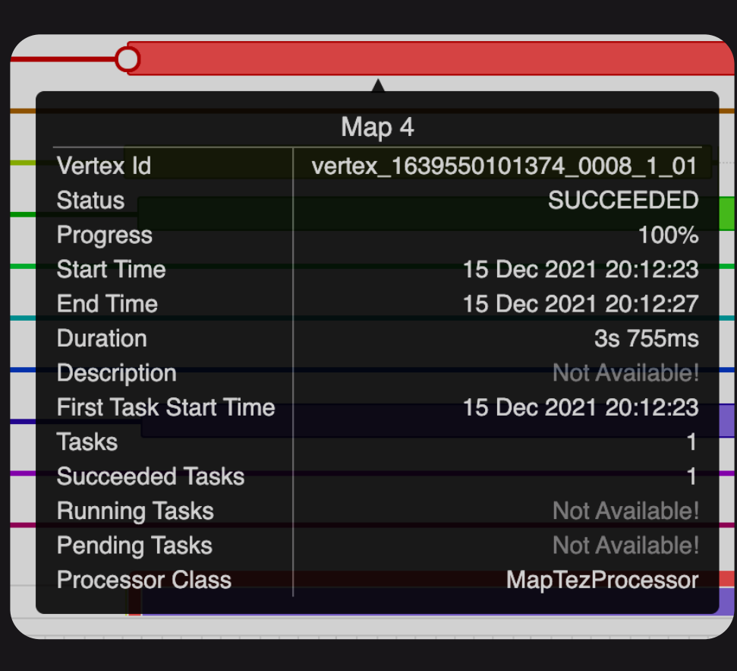
Vertex Swimlane

Каждая горизонтальная дорожка отображает время выполнения вершины. Вертикальные полосы отображают следующее:

- инициализация вершины;
- старт вершины;
- время запуска первой задачи;
- время окончания последней задачи;
- время завершения выполнения вершины.

При наведении мыши на вертикальную полосу виден текущий этап выполнения вершины и время.

Чтобы получить дополнительную информацию о вершине, наведите мышь на произвольное место на горизонтальной полосе.



Map 4

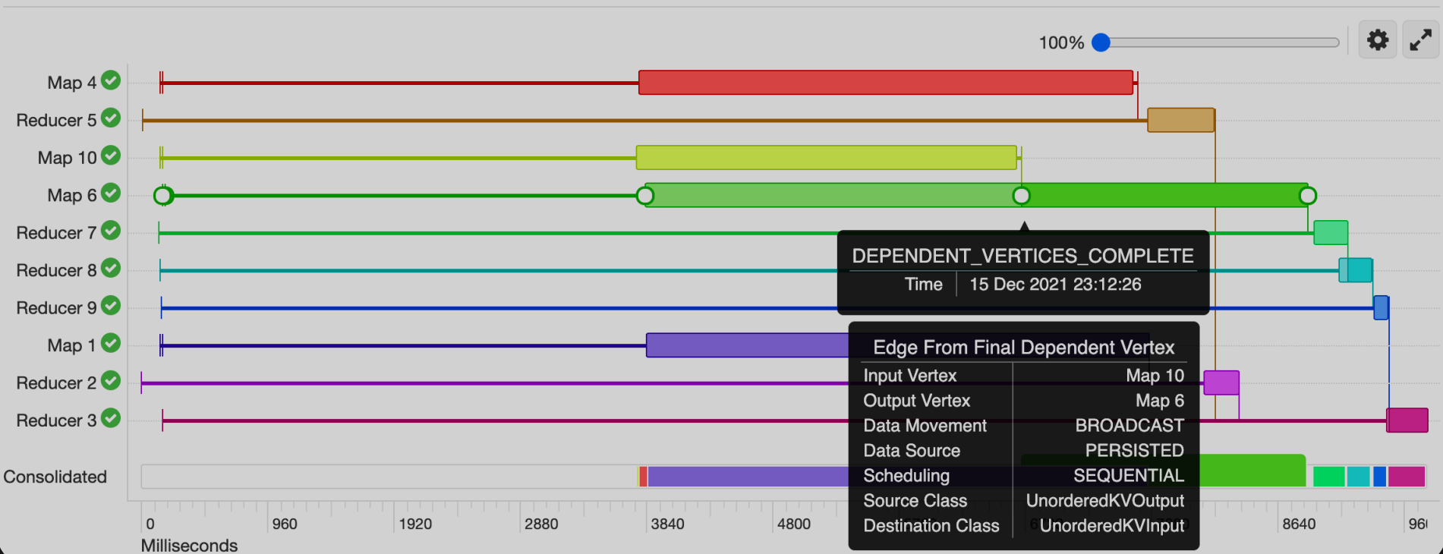
Vertex Id	vertex_1639550101374_0008_1_01
Status	SUCCEEDED
Progress	100%
Start Time	15 Dec 2021 20:12:23
End Time	15 Dec 2021 20:12:27
Duration	3s 755ms
Description	Not Available!
First Task Start Time	15 Dec 2021 20:12:23
Tasks	1
Succeeded Tasks	1
Running Tasks	Not Available!
Pending Tasks	Not Available!
Processor Class	MapTezProcessor

Информация о вершине

Следующая информация отражает затраченное на задачу время, и может быть полезна для отладки запросов:

- ***Vertex ID*** — уникальный идентификатор вершины.
- ***Status*** — статус запроса.
- ***Progress*** — прогресс выполнения запроса, соответствующего вершине; обычно полезно для запросов, выполняющихся долгое время.
- ***Start time*** — время запуска определенной вершины.
- ***End time*** — время окончания работы определенной вершины.
- ***Duration*** — время, затраченное вершиной на выполнение задачи (в миллисекундах).
- ***Description*** — описание вершины.
- ***First task start time*** — время начала выполнения первой задачи.
- ***Tasks*** — общее количество задач, выполненных в определенной вершине.
- ***Succeeded tasks*** — количество успешно выполненных задач вершины.
- ***Running tasks*** — активные задачи.
- ***Pending tasks*** — задачи, ожидающие очереди.
- ***Processor class*** — Hive-процессор для Tez, который формирует вершины в Tez и обрабатывает данные.

Вертикальные линии, соединяющие две вершины, означают зависимость между ними.



Home / DAG [with filtered_crimes as (...cs.distr... / Vertex Swimlane
Version 0.9.2
DAG Details
DAG Counters
Graphical View
All Vertices
All Tasks
All Task Attempts
Vertex Swimlane
Auto Refresh
Last refreshed at 16 Dec 2021 07:26:14
Refresh
100%
Map 4
Reducer 5
Map 10
Map 6
Reducer 7
Reducer 8
Reducer 9
Map 1
Reducer 2
Reducer 3
Consolidated
0
960
1920
2880
3840
4800
8640
Milliseconds

DEPENDENT_VERTICES_COMPLETE

Time	15 Dec 2021 23:12:26

Edge From Final Dependent Vertex

Input Vertex	Map 10
Output Vertex	Map 6
Data Movement	BROADCAST
Data Source	PERSISTED
Scheduling	SEQUENTIAL
Source Class	UnorderedKVOutput
Destination Class	UnorderedKVInput

Зависимости между вершинами

В данном примере, `Map 6` зависит от результата выполнения `Map 10`. `Map 6` закончит выполнение только после успешной отработки `Map 10`. Подобным образом `Reducer 8` ожидает окончания работы `Reducer 7` прежде чем закончить выполнение задачи.

Оптимизация производительности Hive

Константин Алпашкин

Содержание

Назад к содержанию

Партиционирование (Partitioning)

Бакетинг (Bucketing)

Векторизация

Оптимизация затрат (Cost-based optimization)

Использование TEZ

Формат данных ORC

В этой статье описаны методы оптимизации и лучшие практики для повышения производительности вашего Hive-кластера.

Партиционирование (Partitioning)

Партиционирование в Hive — это способ разбиения одной массивной таблицы на меньшие единицы (партиции) в зависимости от данных в столбцах. Hive создает отдельную партицию для каждого уникального значения в столбце. В файловой системе HDFS для каждой партиции создается директория, где хранятся данные каждой партиции.

Партиционирование может значительно ускорить работу Hive, избавляя от необходимости сканировать данные, не имеющие отношения к текущему запросу. Партиционирование наиболее эффективно, когда данные равномерно распределяются между партициями. При сильном "перекосе" (skew) данных в партициях у одних worker-узлов будет гораздо больше данных для обработки, чем у других, что приведет к неэффективному использованию ресурсов кластера. Также эффективность партиционирования во многом зависит от того, как будут использоваться столбцы разделения. Например, рекомендуется разделять данные на партиции по столбцам, которые регулярно участвуют в операциях `GROUP BY`.

Синтаксис для создания таблицы с партициями выглядит следующим образом:

```
CREATE TABLE transactions(
    txn_id int, acc_id int, txn_amount decimal(10,2)
 ) PARTITIONED BY (txn_date date);
```

Данная команда создает таблицу с партициями по столбцу `txn_date`. Вывод команды `DESCRIBE transactions`:

```
+--------------------------+----------------+----------+
|         col_name         |   data_type    | comment  |
+--------------------------+----------------+----------+
| txn_id                   | int            |          |
| acc_id                   | int            |          |
| txn_amount               | decimal(10,2)  |          |
| txn_date                 | date           |          |
|                          | NULL           | NULL     |
| # Partition Information  | NULL           | NULL     |
| # col_name               | data_type      | comment  |
| txn_date                 | date           |          |
+--------------------------+----------------+----------+
```

Для каждой уникальной даты в столбце `txn_date` Hive создаст новую партицию для хранения данных, относящихся к этой дате. Выполните следующую команду, чтобы увидеть HDFS-файлы, используемые Hive для хранения данных таблицы:

```
$ hdfs dfs -ls /apps/hive/warehouse/transactions
```

Вывод:

```
Found 3 items
drwxr-xr-x   - hive hadoop          0 2024-05-01 00:48 /apps/hive/warehouse/transactions/txn_date=2023-01-01
drwxr-xr-x   - hive hadoop          0 2024-05-01 00:49 /apps/hive/warehouse/transactions/txn_date=2023-01-02
drwxr-xr-x   - hive hadoop          0 2024-05-01 00:49 /apps/hive/warehouse/transactions/txn_date=2023-01-03
```

Hive создает директорию для каждой партиции и записывает соответствующие данные в эту директорию. Такое разбиение очень эффективно для дальнейших операций чтения, особенно на больших массивах данных. В последующих запросах Hive должен будет сканировать только определенные директории вместо сканирования всего набора данных.

> ПРИМЕЧАНИЕ
>
> Стоит внимательно выбирать столбец для партиционирования таблицы. Слишком большое количество партиций (а значит, и создаваемых HDFS-файлов) может избыточно нагружать NameNode.

Вы также можете разделить таблицу на партиции по нескольким столбцам одновременно для более детального разбиения. Пример создания таблицы с несколькими партициями показан ниже.

```
CREATE TABLE transactions_1 (
    id int, txn_amount decimal(10,2)
 ) PARTITIONED BY (txn_date date, acc_id int);
```

Получите информацию о партициях таблицы с помощью команды:

```
DESCRIBE transactions_1;
```

Вывод:

```
+--------------------------+----------------+----------+
|         col_name         |   data_type    | comment  |
+--------------------------+----------------+----------+
| id                       | int            |          |
| txn_amount               | decimal(10,2)  |          |
| txn_date                 | date           |          |
| acc_id                   | int            |          |
|                          | NULL           | NULL     |
| # Partition Information  | NULL           | NULL     |
| # col_name               | data_type      | comment  |
| txn_date                 | date           |          |
| acc_id                   | int            |          |
+--------------------------+----------------+----------+
```

При записи данных в такую таблицу Hive сохраняет данные в HDFS, как показано ниже.

```
$ hdfs dfs -ls -R /apps/hive/warehouse/transactions_1
```

Вывод:

```
drwxr-xr-x   - hive hadoop          0 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-01
drwxr-xr-x   - hive hadoop          0 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-01/acc_id=1
-rw-r--r--   3 hive hadoop          8 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-01/acc_id=1/000000_0
drwxr-xr-x   - hive hadoop          0 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-01/acc_id=2
-rw-r--r--   3 hive hadoop          9 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-01/acc_id=2/000000_0
drwxr-xr-x   - hive hadoop          0 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-02
drwxr-xr-x   - hive hadoop          0 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-02/acc_id=3
-rw-r--r--   3 hive hadoop          8 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-02/acc_id=3/000000_0
drwxr-xr-x   - hive hadoop          0 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-03
drwxr-xr-x   - hive hadoop          0 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-03/acc_id=1
-rw-r--r--   3 hive hadoop          8 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-03/acc_id=1/000000_0
drwxr-xr-x   - hive hadoop          0 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-03/acc_id=2
-rw-r--r--   3 hive hadoop          8 2024-05-15 17:45 /apps/hive/warehouse/transactions_1/txn_date=2024-01-03/acc_id=2/000000_0
```

Обратите внимание, что HDFS-директории *txn_date={date}* могут содержать поддиректории *acc_id=N*, где хранятся данные партиций по столбцу `acc_id`.

Бакетинг (Bucketing)

Использование бакетов (buckets) в Hive — это способ оптимизации, предполагающий разбиение одной большой таблицы на несколько файлов, которые легче сканировать. Основное различие между партициями и бакетами заключается в том, что Hive создает новые партиции на основе данных таблицы — по одной партиции на каждое уникальное значение в столбце. В случае с бакетами вы можете указать заранее определенное количество бакетов в момент создания таблицы.

В следующей таблице приведены основные различия между партициями и бакетами.

Партиционирование	Бакетинг
Для каждой партиции создается HDFS-директория	Для каждого бакета создается HDFS-файл
Разбиение таблицы на бакеты осуществляется по одному или по нескольким столбцам	Для разбиения таблицы на бакеты может использоваться только один столбец
Количество партиций определяется содержимым таблицы: одна партиция на каждое уникальное значение в столбце	Количество бакетов может быть указано в момент создания таблицы
Используется предложение `PARTITIONED BY`	Используется предложение `CLUSTERED BY`

Использование бакетов оказывает следующий эффект на Hive-кластер:

- Уменьшается общий размер хранимых данных.
- Снижается скорость вставки данных.
- Выборка по ключу сортировки выполняется быстрее.
- Объединение двух таблиц с помощью sort merge join может быть выполнено без предварительной сортировки.
- Hive-бакеты несовместимы с бакетами Spark из-за различий в алгоритмах хеширования.

Синтаксис для создания таблицы с бакетами имеет следующий вид:

```
CREATE TABLE employees(
    emp_id int,
    first_name string,
    last_name string,
    department_id int
)
CLUSTERED BY (department_id) SORTED BY (emp_id) INTO 5 BUCKETS;
```

Эта команда создает таблицу Hive, данные которой будут храниться в 5 HDFS-файлах (значение `5` в предложении `CLUSTERED BY`). При вставке данных в такую таблицу Hive равномерно распределяет данные между 5 файлами HDFS. В каждом бакете (файле) данные хранятся отсортированными по столбцу `emp_id` (предложение `SORTED BY`).

Как и в случае с партиционированием, распределение данных по отдельным бакетам сокращает общее время чтения. Однако количество бакетов должно быть разумным, так как слишком большое количество маленьких файлов может свести на нет все усилия по оптимизации.

Вы можете использовать бакеты вместе с партициями для расширенного разделения данных и повышения производительности. Например:

```
CREATE TABLE employees(
    emp_id int,
    first_name string,
    last_name string,
    office_num int
)
PARTITIONED BY (department_id int)
CLUSTERED BY (office_num) INTO 5 BUCKETS;
```

> ПРИМЕЧАНИЕ
>
> Столбец, используемый для партиционирования, указывается только в предложении `PARTITIONED BY`. Столбец, используемый для разбиения на бакеты, определяется и в спецификации столбцов, и в предложении `CLUSTERED BY`.

Векторизация

Векторное выполнение запросов — это встроенная функция оптимизации Hive, которая помогает снизить нагрузку на CPU за счет обработки блоков строк, а не одной строки за раз. При обработке блоков строк каждый столбец хранится в памяти как массив (вектор) примитивных типов данных. Арифметические операции и операции сравнения выполняются над примитивными значениями из массива. Такой подход сокращает количество проверок условий для каждого значения в столбце, позволяет более эффективно использовать процессор, активно задействуя кеширование, а также уменьшает общее время обработки данных.

Векторизация в Hive работает только для определенных типов данных и имеет ограниченную поддержку пользовательских функций. По умолчанию векторизация в Hive отключена. Для использования векторизованных запросов необходимо выполнить следующие условия:

- В качестве формата хранения должен использоваться ORC.
- Установлено конфигурационное свойство `hive.vectorized.execution.enabled = true`.

Для выполнения векторизованного запроса используйте следующие шаги:

1. Создайте Hive-таблицу с форматом данных ORC:

```
CREATE TABLE test_table_vectorized (
    id int, value string
) STORED AS ORC;
```

2. Активируйте функцию векторизации с помощью свойства:

```
SET hive.vectorized.execution.enabled=true;
```

3. Выполните команду `EXPLAIN EXTENDED` для функции агрегации:

```
EXPLAIN EXTENDED
    SELECT COUNT(*) FROM test_table_vectorized;
```

Вывод:

```
+----------------------------------------------------+
|                      Explain                       |
+----------------------------------------------------+
| STAGE DEPENDENCIES:                                |
|   Stage-1 is a root stage                          |
|   Stage-0 depends on stages: Stage-1               |
|                                                    |
| STAGE PLANS:                                       |
|   Stage: Stage-1                                   |
|     Tez                                            |
|       DagId: hive_20240504214716_0b61644b-d042-4333-8566-05a678208479:12 |
|       Edges:                                       |
|         Reducer 2 <- Map 1 (CUSTOM_SIMPLE_EDGE)    |
|       DagName: hive_20240504214716_0b61644b-d042-4333-8566-05a678208479:12 |
|       Vertices:                                    |
|         Map 1                                      |
|             Map Operator Tree:                     |
|                 TableScan                          |
|                   alias: test_table_vectorized     |
|                   Statistics: Num rows: 2 Data size: 182 Basic stats: COMPLETE Column stats: COMPLETE |
|                   GatherStats: false               |
|                   Select Operator                  |
|                     Statistics: Num rows: 2 Data size: 182 Basic stats: COMPLETE Column stats: COMPLETE |
|                     Group By Operator              |
|                       aggregations: count()        |
|                       mode: hash                   |
|                       outputColumnNames: _col0     |
|                       Statistics: Num rows: 1 Data size: 8 Basic stats: COMPLETE Column stats: COMPLETE |
|                       Reduce Output Operator       |
|                         null sort order:           |
|                         sort order:                |
|                         Statistics: Num rows: 1 Data size: 8 Basic stats: COMPLETE Column stats: COMPLETE |
|                         tag: -1                    |
|                         value expressions: _col0 (type: bigint) |
|                         auto parallelism: false    |
|             Execution mode: vectorized             |
|             ...
|         Reducer 2                                  |
|             Execution mode: vectorized             |
|             ...
|                                                    |
|   Stage: Stage-0                                   |
|     Fetch Operator                                 |
|       limit: -1                                    |
|       Processor Tree:                              |
|         ListSink                                   |
|                                                    |
+----------------------------------------------------+
```

Как можно увидеть в выводе, определенные этапы выполнения запроса содержат строку `Execution mode: vectorized`, что свидетельствует о том, что данный этап был выполнен в режиме векторизации.

Если установить значение `hive.vectorized.execution.enabled=false` (значение по умолчанию), Hive вернется к построчной обработке без использования векторов.

Оптимизация затрат (Cost-based optimization)

Механизм оптимизации затрат (Cost-Based Optimization, CBO) — это встроенная функция Hive, которая выполняет такие оптимизации, как переписывание запросов (query rewriting), переупорядочивание JOIN-операций для более эффективной работы, исключение JOIN-операций из AST и так далее.

В основе данного функционала лежит фреймворк Apache Calcite, а основным источником данных для выполнения оптимизаций является статистика таблиц. По умолчанию механизм оптимизации затрат отключен. Для активации CBO выполните следующие шаги.

1. Установите следующие конфигурационные свойства в ADCM (***Clusters → <clusterName> → Services → Hive → Primary configuration***):
    - `hive.cbo.enable=true`
    - `hive.compute.query.using.stats=true`
    - `hive.stats.fetch.column.stats=true`

2. Выполните сбор статистики для необходимой таблицы. Это обновит актуальную статистику таблицы в базе данных Hive Metastore.

```
ANALYZE TABLE transactions COMPUTE STATISTICS;
```

Использование TEZ

Apache TEZ — это платформа для создания высокопроизводительных приложений, которая используется в качестве движка исполнения для Hive. TEZ считается гораздо более гибким и мощным преемником фреймворка MapReduce.

TEZ является предпочтительным выбором для вычислительных задач Hive, поскольку он уменьшает количество обращений к диску и уменьшает общее время выполнения.

В ADH TEZ доступен в виде отдельного компонента (Hive Tez) сервиса Hive. Если в кластере ADH установлен компонент Hive Tez, он по умолчанию используется в качестве основного движка исполнения (параметр `hive.execution.engine`).

Формат данных ORC

Использование подходящего формата хранения данных имеет основополагающее значение для производительности Hive. Формат Optimized Row Columnar (ORC) — это формат данных, разработанный специально для экосистемы Hadoop. Будучи столбцовым форматом, ORC оптимизирован для операций на основе столбцов, например, фильтрация и агрегация. По сравнению с другими форматами данных, поддерживаемыми Hive (Parquet, Avro, TextFile и так далее), ORC опережает их по показателям компрессии и скорости, позволяя экономить петабайты данных.

По умолчанию Hive использует формат хранения `TextFile`. Чтобы создать таблицу, хранящую данные в формате ORC, используйте предложение `STORED AS ORC`, например:

```
CREATE TABLE test_tbl_orc (
  id int,
  data String
) STORED AS ORC;
```

Резервное копирование и восстановление Hive

Константин Алпашкин

Содержание

Назад к содержанию

Резервное копирование таблиц Hive
Восстановление таблиц Hive
Пример

В данной статье описаны процессы резервного копирования и восстановления Hive-таблиц. Создание резервной копии таблицы Hive включает следующие шаги:

- Создание копии данных из хранилища Hive (Hive warehouse) в HDFS.
- Создание дампа базы данных Hive Metastore.

Для восстановления Hive-таблицы из резервной копии необходимы обратные действия, а именно:

- Копирование файлов из резервной копии в HDFS.
- Импорт дамп-файла в базу данных Hive Metastore.

Подробности каждого из этапов описаны далее в статье. Также в статье приведен пример, в котором показаны все операции, необходимые для выполнения резервного копирования и восстановления таблиц Hive.

Резервное копирование таблиц Hive

Для создания резервной копии managed Hive-таблицы выполните следующее:

1. Остановите сервис Hive с помощью ADCM.
2. Создайте резервную копию HDFS-директории Hive warehouse.
   Hive хранит содержимое таблиц в HDFS-директории (устанавливается параметром `hive.metastore.warehouse.dir`, по умолчанию */apps/hive/warehouse*). Для создания копии HDFS-директории используйте снепшоты HDFS, distcp или любой другой способ для копирования HDFS-директории.
3. Создайте резервную копию базы данных Hive Metastore.
   Помимо "сырых" данных (содержимого таблиц), которые хранятся в HDFS, Hive Metastore использует реляционную базу данных для хранения метаданных о таблицах. Эти метаданные включают схему, информацию о партиционировании, бакетах, индексах, транзакциях, а также прочую системную информацию, необходимую для корректной работы Hive. Все метаданные хранятся в базе данных `hive` — именно эту базу необходимо экспортировать в дамп при создании резервной копии.

Восстановление таблиц Hive

Чтобы восстановить Hive-таблицу из резервной копии, необходимо выполнить следующее:

1. Остановите сервис Hive с помощью ADCM.
2. Скопируйте файлы резервной копии в соответствующую HDFS-директорию Hive warehouse.
3. Импортируйте SQL дамп-файл в базу данных Hive Metastore.

> **ПРИМЕЧАНИЕ**
>
> При импорте дамп-файла в Hive Metastore существующая база данных `hive` с метаданными полностью перезаписывается. Таким образом, после импорта дампа в Hive-кластере будут доступны лишь те Hive-таблицы, которые присутствовали в дампе.

Пример

Ниже приведен сценарий, описывающий все необходимые для выполнения резервного копирования и восстановления шаги. Пример предполагает полную перезапись метаданных Hive при восстановлении таблицы. Это означает, что после восстановления таблицы в кластере Hive останутся только таблицы из резервной копии.

1. Создайте тестовую Hive-таблицу и запишите в нее данные:

```
SELECT * FROM transactions;
+-----------------------+-----------------------+--------------------------+------------------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  |
+-----------------------+-----------------------+--------------------------+------------------------+
| 1                     | 1002                  | 10.00                    | 2023-01-01             |
| 8                     | 1003                  | 50.00                    | 2023-01-01             |
| 3                     | 1002                  | 30.00                    | 2023-01-02             |
| 4                     | 1001                  | 100.50                   | 2023-01-02             |
| 2                     | 1002                  | 20.00                    | 2023-01-03             |
| 6                     | 1001                  | 200.50                   | 2023-01-03             |
| 7                     | 1003                  | 50.00                    | 2023-01-03             |
| 5                     | 1001                  | 150.50                   | 2023-01-04             |
| 9                     | 1003                  | 75.00                    | 2023-01-04             |
+-----------------------+-----------------------+--------------------------+------------------------+
```

SQL для создания тестовой таблицы

Для создания и наполнения тестовой таблицы Hive выполните следующий SQL с помощью */bin/beeline*.

```
DROP TABLE IF EXISTS transactions;
CREATE TABLE transactions(txn_id int, acc_id int, txn_amount decimal(10,2), txn_date date);
INSERT INTO transactions VALUES
(1, 1002, 10.00, '2023-01-01'),
(2, 1002, 20.00, '2023-01-03'),
(3, 1002, 30.00, '2023-01-02'),
(4, 1001, 100.50, '2023-01-02'),
(5, 1001, 150.50, '2023-01-04'),
(6, 1001, 200.50, '2023-01-03'),
(7, 1003, 50.00, '2023-01-03'),
(8, 1003, 50.00, '2023-01-01'),
(9, 1003, 75.00, '2023-01-04');
```

2. С помощью ADCM остановите сервис Hive.
3. Создайте снепшот директории Hive warehouse в HDFS. В этом сценарии нужная директория — */apps/hive/warehouse/transactions* (при условии, что `hive.metastore.warehouse.dir` указывает на *apps/hive/warehouse/*). Для создания снепшота следуйте инструкциям ниже.

   Разрешите создание снепшотов для HDFS-директории */apps/hive/warehouse/transactions*:

```
$ sudo -u hdfs hdfs dfsadmin -allowSnapshot /apps/hive/warehouse/transactions
```

   Результат:

```
Allowing snapshot on /apps/hive/warehouse/transactions succeeded
```

   Создайте снепшот HDFS-директории:

```
$ sudo -u hdfs hdfs dfs -createSnapshot /apps/hive/warehouse/transactions snapshot_transactions
```

   Результат:

```
Created snapshot /apps/hive/warehouse/{test-table-name}/.snapshot/snapshot_transactions
```

   Скопируйте снепшот во временную локацию, поскольку директория */apps/hive/warehouse/transactions* будет удалена на дальнейших шагах. Например:

```
$ sudo -u hdfs hadoop fs -mkdir /user/tmp
$ sudo -u hdfs hadoop fs -chown hdfs:hdfs /user/tmp
$ sudo -u hdfs hdfs dfs -cp /apps/hive/warehouse/transactions/.snapshot/snapshot_transactions /user/tmp
```

   Удалите снепшот из */apps/hive/warehouse/transactions*. Этот шаг обязателен, иначе Hive не сможет удалить warehouse HDFS-директорию из-за наличия файлов снепшота внутри.

```
$ sudo -u hdfs hdfs dfs -deleteSnapshot /apps/hive/warehouse/transactions/ snapshot_transactions
```

4. Выполните дамп базы данных Hive Metastore. В этом сценарии Hive использует базу данных MariaDB, которая устанавливается как отдельный сервис ADH. Для других типов баз данных, например Postgres, используйте соответствующий синтаксис дампа. Для создания дампа базы данных MariaDB на хосте ADH с сервером MariaDB выполните команду с правами суперпользователя:

```
$ sudo mysqldump -uroot hive > hive_backup.sql
```

5. Запустите сервис Hive.
6. Удалите тестовую Hive-таблицу.

```
DROP TABLE transactions;
```

   Убедитесь, что HDFS директория */apps/hive/warehouse/transactions* также была удалена.

7. Восстановите тестовую таблицу Hive из резервной копии. Для этого выполните следующие шаги.

   Скопируйте файлы снепшота в директорию Hive warehouse в HDFS.

```
$ sudo -u hdfs hdfs dfs -cp /user/tmp/snapshot_transactions /apps/hive/warehouse/transactions
$ sudo -u hdfs hdfs dfs -ls /apps/hive/warehouse/transactions
```

   Затем импортируйте SQL дамп-файл в базу данных Hive Metastore. Для MariaDB используйте следующую команду:

```
$ sudo mysql hive < hive_backup.sql
```

8. Выполните рестарт сервиса Hive.
9. Запросите данные из восстановленной Hive-таблицы:

```
SELECT * FROM transactions;
```

   Вывод показывает, что тестовая таблица была успешно восстановлена.

```
+-----------------------+-----------------------+--------------------------+------------------------+
| transactions.txn_id  | transactions.acc_id  | transactions.txn_amount  | transactions.txn_date  |
+-----------------------+-----------------------+--------------------------+------------------------+
| 1                     | 1002                  | 10.00                    | 2023-01-01             |
| 2                     | 1002                  | 20.00                    | 2023-01-03             |
| 3                     | 1002                  | 30.00                    | 2023-01-02             |
| 4                     | 1001                  | 100.50                   | 2023-01-02             |
| 5                     | 1001                  | 150.50                   | 2023-01-04             |
| 6                     | 1001                  | 200.50                   | 2023-01-03             |
| 7                     | 1003                  | 50.00                    | 2023-01-03             |
| 8                     | 1003                  | 50.00                    | 2023-01-01             |
| 9                     | 1003                  | 75.00                    | 2023-01-04             |
+-----------------------+-----------------------+--------------------------+------------------------+
```

Режим высокой доступности в Hive

Константин Алпашкин

Содержание

Назад к содержанию

Строка JDBC-подключения в режиме HA

Режим высокой доступности (High Availability, HA) включен в Hive по умолчанию.

Если быть более точным, то в режиме HA работает компонент HiveServer2. При добавлении новых компонентов HiveServer2 в кластер ZooKeeper автоматически регистрирует их и обновляет JDBC-строку подключения.

Строка JDBC-подключения в режиме HA

В HA-режиме строка JDBC-подключения содержит ZooKeeper ensemble. Актуальная строка JDBC-подключения доступна на странице Hive ***Info*** в ADCM (***Clusters*** → ***Services*** → ***Hive*** → ***Info***).

Ниже показаны примеры строк JDBC-подключения в зависимости от используемого механизма безопасности.

HA без SSL/Kerberos

Режим высокой доступности, незащищенное соединение без SSL/Kerberos:

```
jdbc:hive2://<cluster_host_0>:2181,<cluster_host_1>:2181,
<cluster_host_N>:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/<cluster_id>/<namespace>
```

HA с SSL

Режим высокой доступности, защищенное соединение с SSL:

```
jdbc:hive2://<cluster_host_0>:2181,<cluster_host_1>:2181,
<cluster_host_N>:2181;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/<cluster_id>/<namespace>;ssl=true;ss
lTrustStore=/tmp/truststore.jks;trustStorePassword=bigdata
```

HA с SSL+Kerberos

Режим высокой доступности, защищенное соединение с SSL+Kerberos:

```
jdbc:hive2://<cluster_host_0>:2181,<cluster_host_1>:2181,
<cluster_host_N>:2181;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/<cluster_id>/<namespace>;ssl=true;ss
lTrustStore=/tmp/truststore.jks;trustStorePassword=bigdata;principal=hive/_HOST@EXAMPLE.COM
```

Где `EXAMPLE.COM` — ваш Kerberos realm, например `RU-CENTRAL1.INTERNAL`.

ПРИМЕЧАНИЕ

Вы также можете установить соединение с HiveServer2 напрямую, в non-HA режиме, используя порт Thrift (по умолчанию `10000`). В таком случае строка JDBC-подключения имеет вид `jdbc:hive2://<cluster_host>:10000/`.

Логирование в Hive

Константин Алпашкин

Содержание

Назад к содержанию

Расположение и формат лог-файлов
Типы логов Hive
  Логи HiveServer2
  Логи Metastore
  Логи Beeline
Настройка логирования в ADCM
  Типовые задачи
Просмотр логов в веб-интерфейсе

Под капотом Hive использует фреймворк Log4j 2 для ведения логов. Для конфигурации процессов логирования в Hive используйте настройки ***Custom log4j.properties*** в ADCM. Более подробная информация об этих настройках доступна в разделе Настройка логирования с помощью ADCM.

Расположение и формат лог-файлов

По умолчанию Hive хранит все логи в локальной файловой системе в корневой лог-директории */var/log/hive*. Корневую директорию для лог-файлов можно указать с помощью свойства Log4j `hive.log.dir`. Логи Hive хранятся в виде обычных текстовых файлов с расширениями *.log*/*.out*. Большинство событий в Hive-логах имеют следующую структуру:

```
[Timestamp] [Severity] [Thread that generates the event] [Event message]
```

Например:

```
2024-01-29T16:26:24,216  INFO [HiveServer2-Background-Pool: Thread-71] ql.Driver: Executing
command(queryId=hive_20240129162624_f4dfb3c4-3e27-4113-bbad-d524e54fcc26): drop table if exists
hive_main_through_beeline_check_75320
```

Типы логов Hive

Логи Hive генерируются отдельными компонентами, такими как HiveServer2, Metastore, Beeline. Каждый компонент имеет свою собственную настраиваемую конфигурацию Log4j, которая определяет правила логирования для этого компонента. Основные типы логов, генерируемые компонентами Hive, описаны ниже.

Логи HiveServer2

По умолчанию Hive записывает логи HiveServer2 в файлы с именем *hive-server2.log*. В этих логах отражена активность компонента HiveServer2, включая все ошибки и исключения, возникающие в ядре HiveServer2. Например:

```
2024-01-30T13:42:04,048  INFO [1bd2c26d-2da1-4bea-aa8d-53e24b21fa9e HiveServer2-Handler-Pool: Thread-60] ql.Context: New scratch
dir is hdfs://adh/tmp/hive/hive/1bd2c26d-2da1-4bea-aa8d-53e24b21fa9e/hive_2024-01-30_13-42-02_298_4565019303774254377-2
2024-01-30T13:42:04,097 ERROR [1bd2c26d-2da1-4bea-aa8d-53e24b21fa9e HiveServer2-Handler-Pool: Thread-60] parse.CalcitePlanner:
CBO failed, skipping CBO.
org.apache.hadoop.hive.ql.parse.SemanticException: Line 1:12 Cannot insert into target table because column number/types are
different 'transactions': Table insclause-0 has 4 columns, but query has 1 columns.
        at org.apache.hadoop.hive.ql.parse.SemanticAnalyzer.genConversionSelectOperator(SemanticAnalyzer.java:8031) ~[hive-exec-3.
1.3.jar:3.1.3]
        at org.apache.hadoop.hive.ql.parse.SemanticAnalyzer.genFileSinkPlan(SemanticAnalyzer.java:7576) ~[hive-exec-3.1.3.jar:3.1.
3]
        at org.apache.hadoop.hive.ql.parse.SemanticAnalyzer.genPostGroupByBodyPlan(SemanticAnalyzer.java:10635) ~[hive-exec-3.1.3.
jar:3.1.3]
        at org.apache.hadoop.hive.ql.parse.SemanticAnalyzer.genBodyPlan(SemanticAnalyzer.java:10507) ~[hive-exec-3.1.3.jar:3.1.3]
        at org.apache.hadoop.hive.ql.parse.SemanticAnalyzer.genPlan(SemanticAnalyzer.java:11420) ~[hive-exec-3.1.3.jar:3.1.3]
...
```

В файле *hive-server2.out* содержится перенаправленный вывод HiveServer2; данный файл очищается при каждом запуске HiveServer2.

Логи Metastore

В файлах *hive-metastore.log* хранится информация, связанная с активностью компонента Hive Metastore. Например:

```
2024-01-30T14:02:50,819  INFO [main] server.HiveServer2: Starting HiveServer2
2024-01-30T14:02:52,582  INFO [main] service.AbstractService: Service:OperationManager is inited.
2024-01-30T14:02:52,582  INFO [main] service.AbstractService: Service:SessionManager is inited.
2024-01-30T14:02:52,583  INFO [main] service.AbstractService: Service:CLIService is inited.
2024-01-30T14:02:52,583  INFO [main] service.AbstractService: Service:ThriftBinaryCLIService is inited.
...
```

Кроме того, логи Metastore содержат информацию об аудите Hive. Например:

```
$ grep 'audit' /var/log/hive/hive-metastore.log
```

Вывод:

```
2024-02-01T00:00:48,327  INFO [pool-6-thread-4] HiveMetaStore.audit: ugi=hive   ip=10.92.41.182 cmd=source:10.92.41.182
get_config_value: name=metastore.batch.retrieve.max defaultValue=50
...
```

Логи Beeline

Лог-файлы *hive-beeline.log* отражают информацию о работе Beeline shell.

Настройка логирования в ADCM

С использованием Log4j настройка процессов логирования в Hive сводится к модификации конфигурационных параметров в файлах **-log4j.properties*. Эти файлы доступны в различных локациях на хостах с соответствующими компонентами Hive, и Hive считывает их при старте. Однако, чтобы не редактировать эти файлы вручную, ADCM предоставляет удобные настройки для изменения конфигураций Log4j на всех хостах кластера ADH. Для этого:

1. В ADCM перейдите на страницу ***Clusters*** → ***<your_cluster_name>*** → ***Services*** → ***Hive*** → ***Primary Configuration*** и включите опцию ***Show advanced***.
2. В разделе настроек ***Custom log4j.properties*** откройте конфигурацию ***hive-log4j.properties***, которая соответствует нужному Hive-компоненту и отредактируйте свойства Log4j. Параметры Log4j, указанные таким способом, расширяют/переопределяют значения в файлах **-log4j.properties*, которые Hive считывает при старте. Подробная информация о конфигурационных свойствах Log4j доступна в документации Log4j 2.
3. Сохраните конфигурацию сервиса Hive.
4. Выполните рестарт сервиса Hive.

> **ПРИМЕЧАНИЕ**
>
> В данной статье любое упоминание действия "обновите свойство Log4j" подразумевает изменение конфигурационных параметров Log4j через ADCM UI, включая обязательный рестарт Hive.

Ниже приведен стандартный файл *hive-log4j.properties*, используемый компонентом HiveServer2. Основные конфигурационные параметры подсвечены комментариями, чтобы помочь вам быстрее освоить настройку Log4j. Подробная информация о конфигурационных свойствах Log4j доступна в документации Log4j 2.

```
status = INFO
name = HiveLog4j2
packages = org.apache.hadoop.hive.ql.log

# list of properties
property.hive.log.level = INFO ①
property.hive.root.logger = DRFA ②
property.hive.log.dir = /var/log/hive ③
property.hive.log.file = hive-server2.log ④
property.hive.perflogger.log.level = INFO

# list of all appenders
appenders = console, DRFA ⑤

# console appender
appender.console.type = Console
appender.console.name = console
appender.console.target = SYSTEM_ERR
appender.console.layout.type = PatternLayout
appender.console.layout.pattern = %d{ISO8601} %5p [%t] %c{2}: %m%n

# daily rolling file appender
appender.DRFA.type = RollingRandomAccessFile ⑥
appender.DRFA.name = DRFA
appender.DRFA.fileName = ${sys:hive.log.dir}/${sys:hive.log.file}
# Use %pid in the filePattern to append <process-id>@<host-name> to the filename if you want separate log files for different CLI
session
appender.DRFA.filePattern = ${sys:hive.log.dir}/${sys:hive.log.file}.%d{yyyy-MM-dd} ⑦
appender.DRFA.layout.type = PatternLayout
appender.DRFA.layout.pattern = %d{ISO8601} %5p [%t] %c{2}: %m%n ⑧
appender.DRFA.policies.type = Policies
appender.DRFA.policies.time.type = TimeBasedTriggeringPolicy ⑨
appender.DRFA.policies.time.interval = 1 ⑩
appender.DRFA.policies.time.modulate = true

# list of all loggers
loggers = NIOServerCnxn, ClientCnxnSocketNIO, DataNucleus, Datastore, JPOX, PerfLogger, AmazonAws, ApacheHttp ⑪

logger.NIOServerCnxn.name = org.apache.zookeeper.server.NIOServerCnxn
logger.NIOServerCnxn.level = WARN

logger.ClientCnxnSocketNIO.name = org.apache.zookeeper.ClientCnxnSocketNIO
logger.ClientCnxnSocketNIO.level = WARN

logger.DataNucleus.name = DataNucleus
logger.DataNucleus.level = ERROR

logger.Datastore.name = Datastore
logger.Datastore.level = ERROR

logger.JPOX.name = JPOX
logger.JPOX.level = ERROR

logger.AmazonAws.name=com.amazonaws
logger.AmazonAws.level = INFO

logger.ApacheHttp.name=org.apache.http
logger.ApacheHttp.level = INFO

logger.PerfLogger.name = org.apache.hadoop.hive.ql.log.PerfLogger
logger.PerfLogger.level = ${sys:hive.perflogger.log.level}

# root logger
rootLogger.level = ${sys:hive.log.level} ⑫
rootLogger.appenderRefs = root
rootLogger.appenderRef.root.ref = ${sys:hive.root.logger}
```

① Установка базового уровня логирования. Возможные уровни: `ALL`, `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, `FATAL`, `OFF`. События с меньшим уровнем не попадают в лог. Например, при использовании `DEBUG` генерируются более детальные и объемные логи, а при уровне `ERROR` Hive будет записывать в лог только ошибки и более критичные события.

② Установка рутового логера (root logger). Логер `DRFA` записывает события в файл, который периодически ротируется.

③ Корневая директория для хранения лог-файлов. Все логи, сгенерированные с помощью данной конфигурации Log4j, будут храниться в указанной директории.

④ Имя лог-файла.

⑤ Список Log4j-аппендеров (appenders), которые отвечают за отправку событий в файл, в JMS-очередь, и так далее. В данной конфигурации используются 2 аппендера, которые направляют события в файл и в консоль.

⑥ Использование аппендера RollingRandomAccessFile, который записывает события в файл.

⑦ Шаблон, используемый для переименования файла при ротации.

⑧ Шаблон для преобразования объекта события в строку.

⑨ Использование политики TimeBasedTriggeringPolicy, которая определяет интервал ротации лог-файлов.

⑩ Стандартный интервал ротации лог-файлов (1 сутки).

⑪ Список всех активных логеров. Каждый логер генерирует события, относящиеся к определенному функционалу.

⑫ Объявление рутового логера (root logger). В Log4j логеры имеют иерархическую структуру, а свойства рутового логера наследуются другими нерутовыми логерами (за исключением явного переопределения свойств в логере-наследнике).

Типовые задачи

Ниже показаны решения типовых задач, которые могут возникнуть при настройке логирования Hive с помощью Log4j.

Изменение стандартного уровня логирования

Чтобы изменить уровень логирования (по умолчанию `INFO`), используйте свойство `property.hive.log.level`. Вы можете указать рутовый уровень, который будет наследоваться другими логерами Log4j, либо установить нужный уровень для конкретного логера, например `logger.PerfLogger.level = <OTHER_LEVEL>`.

Изменение местоположения логов по умолчанию

По умолчанию журналы Hive хранятся в директории */var/log/hive/*. Для изменения директории используйте свойство `property.hive.log.dir`.

Включение/отключение логеров

В Log4j каждый `logger` отвечает за генерацию событий, связанных с определенным функционалом. Чтобы исключить из журналов ненужные события (например, события `PerfLogger`), удалите соответствующий логер из списка `loggers = …` или установите для этого логера более высокий уровень логирования.

Отображение ID запроса/сессии в логах HiveServer2

Вы можете включить отображение ID запросов/сессий Hive в логах HiveServer2. Для этого измените параметр `appender.DRFA.layout.pattern` в файле конфигурации ***HiveServer2 hive-log4j.properties*** как показано в примере ниже.

```
appender.DRFA.layout.pattern = %d{ISO8601} %5p [%t] %c{2}: QueryID: %X{queryId}; SessionID: %X{sessionId} %m%n
```

После этого ID запросов/сессий Hive будут доступны в логах. Например:

```
2024-02-01T17:17:01,175  INFO [5818afd9-10b9-4d59-9efc-33fb4b5d2c2c HiveServer2-Handler-Pool: Thread-49] exec.SelectOperator:
QueryID: hive_20240201171700_bdc1be52-6070-41f9-bb5d-d9b2c33dd24b; SessionID: 5818afd9-10b9-4d59-9efc-33fb4b5d2c2c SELECT
struct<txn_id:int,acc_id:int,txn_amount:decimal(10,2),txn_date:date>
2024-02-01T17:17:01,175  INFO [5818afd9-10b9-4d59-9efc-33fb4b5d2c2c HiveServer2-Handler-Pool: Thread-49] exec.
ListSinkOperator: QueryID: hive_20240201171700_bdc1be52-6070-41f9-bb5d-d9b2c33dd24b; SessionID: 5818afd9-10b9-4d59-9efc-
33fb4b5d2c2c Initializing operator LIST_SINK[3]
...
```

Изменение политики ротации логов

По умолчанию логи Hive ротируются ежедневно. Чтобы изменить политику ротации и, например, добавить ротацию в зависимости от размера файла, используйте SizeBasedTriggeringPolicy как показано ниже. Следующие параметры Log4j позволяют выполнять ротацию лог-файла при достижении определенного размера (10 МБ).

```
appender.DRFA.policies.size.type = SizeBasedTriggeringPolicy
appender.DRFA.policies.size.size = 10MB
# '%i' is mandatory if SizeBasedTriggeringPolicy is used with TimeBasedTriggeringPolicy:
appender.DRFA.filePattern = ${sys:hive.log.dir}/${sys:hive.log.file}.%d{yyyy-MM-dd}-%i
```

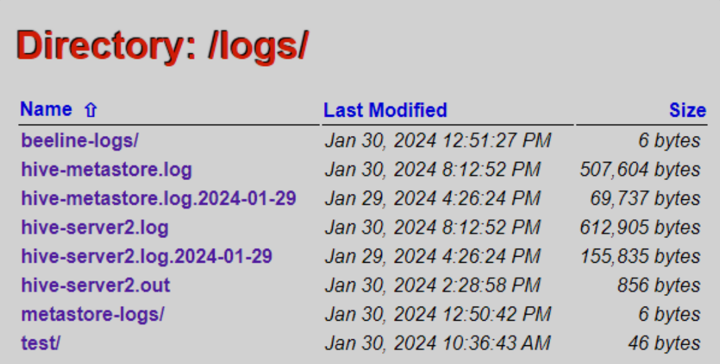
Автоматическая архивация логов

Сразу после выполнения ротации вы можете сжать "старый" лог-файл для экономии места. Для этого необходимо указать желаемый формат архива в свойстве `filePattern`, как показано в примере ниже. Если значение `filePattern` заканчивается символами ".gz", ".zip", ".bz2", ".deflate", ".pack200", ".zst" или ".xz", Log4j автоматически выполнит архивацию, используя соответствующий алгоритм сжатия.

```
appender.DRFA.filePattern = ${sys:hive.log.dir}/${sys:hive.log.file}.%d{yyyy-MM-dd}-%i.log.gz
```

Просмотр логов в веб-интерфейсе

Веб-интерфейс HiveServer2 позволяет просматривать текстовые логи на странице Local logs. На этой странице отображается содержимое корневой лог-директории (устанавливается свойством `hive.log.dir`), где вы можете перемещаться между директориями и просматривать логи в виде обычного текста.



Directory: /logs/

Name ⇧	Last Modified	Size
beeline-logs/	Jan 30, 2024 12:51:27 PM	6 bytes
hive-metastore.log	Jan 30, 2024 8:12:52 PM	507,604 bytes
hive-metastore.log.2024-01-29	Jan 29, 2024 4:26:24 PM	69,737 bytes
hive-server2.log	Jan 30, 2024 8:12:52 PM	612,905 bytes
hive-server2.log.2024-01-29	Jan 29, 2024 4:26:24 PM	155,835 bytes
hive-server2.out	Jan 30, 2024 2:28:58 PM	856 bytes
metastore-logs/	Jan 30, 2024 12:50:42 PM	6 bytes
test/	Jan 30, 2024 10:36:43 AM	46 bytes
test1/	Jan 30, 2024 10:36:43 AM	47 bytes

Страница Local logs

ACID-транзакции в Hive

Константин Алпашкин

Содержание

Назад к содержанию

- Дельта-файлы и сжатие
- Активация транзакций ACID
- Использование транзакционных таблиц
  - Создание таблицы
  - Преобразование в ACID-таблицу
  - Вставка и обновление данных
- Специальные команды транзакций
  - COMPACT
  - SHOW TRANSACTIONS
  - SHOW LOCKS
- Ограничения

Hive поддерживает транзакции ACID, которые обладают следующими свойствами:

- **Атомарность (atomicity)**. Гарантирует, что транзакция либо выполняется полностью, либо не выполняются вообще. Другими словами, транзакции неделимы, и если одна часть транзакции не выполняется, вся транзакция откатывается, оставляя систему в исходном состоянии.
- **Консистентность (consistency)**. Обеспечивает целостность базы данных как до, так и после выполнения транзакции. Транзакции обеспечивают целостность данных и наложенных на них констрейнтов.
- **Изоляция (isolation)**. Гарантирует, что параллельное выполнение нескольких транзакций не будет мешать друг другу.
- **Надежность (durability)**. В момент коммита транзакции ее изменения становятся постоянными и не теряются даже в случае системного сбоя.

В совокупности эти свойства обеспечивают надежность, целостность и устойчивость транзакций Hive. Hive поддерживает транзакции на уровне строк, что позволяет приложениям (например, стриминговый сервис на базе Kafka) постоянно записывать данные в таблицы Hive, в то время как другие приложения могут параллельно считывать свежие данные, не мешая друг другу.

Дельта-файлы и сжатие

При вставке или обновлении данных в транзакционной таблице Hive создает набор дельта-файлов (delta files). В каждом файле хранятся изменения, внесенные в таблицу (партицию) в рамках одной транзакции. Когда Hive-клиенты запрашивают данные, Hive анализирует дельта-файлы, чтобы учесть все обновления из отдельных транзакций при построении результирующего набора. Таким образом, возвращается наиболее актуальное состояние таблицы — в этом заключается основная идея транзакций Hive.

Чтобы избежать накопления слишком большого количества дельта-файлов (что неизбежно приводит к снижению производительности), Hive выполняет регулярную очистку, называемую сжатием (compaction). Процесс сжатия автоматически выполняется в фоновом режиме через регулярные промежутки времени, не мешая одновременному чтению/записи.

Существует два типа сжатий:

- **Минорное (minor)**. Объединяет несколько дельта-файлов в один.
- **Мажорное (major)**. Переписывает один или несколько дельта-файлов и базовый файл в новый базовый файл. Этот процесс гораздо более ресурсоемкий, чем минорное сжатие.

Помимо автоматических запусков процесса сжатия вы также можете запустить процесс вручную с помощью команды COMPACT.

Активация транзакций ACID

По умолчанию транзакции Hive ACID отключены. Для работы с транзакционными таблицами включите опцию ***ACID Transactions*** в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Hive*** → ***Primary configuration***), а затем перезапустите сервис Hive. При включении этой опции ADCM автоматически обновляет все конфигурационные параметры Hive, необходимые для активации механизма транзакций, включая менеджер транзакций (`hive.txn.manager`).

Использование транзакционных таблиц

Создание таблицы

Для создания транзакционной таблицы с полной поддержкой ACID необходимо следующее:

- Транзакции Hive ACID включены в ADCM.
- Создаваемая таблица разделена на бакеты.
- В качестве формата хранения используется ORC. В настоящее время это единственный формат, поддерживаемый для транзакционных таблиц.
- При создании таблицы устанавливается свойство `TBLPROPERTIES ("transactional"="true")`.

Пример создания транзакционной таблицы:

```
CREATE TABLE `transactions`
    (`txn_id` int,
    `acc_id` int,
    `txn_amount` decimal(10,2),
    `txn_date` date)
    CLUSTERED BY (acc_id) INTO 3 BUCKETS ①
    STORED AS ORC ②
    TBLPROPERTIES ("transactional"="true"); ③
```

① Разделение новой таблицы на бакеты.

② Использование ORC в качестве формата хранения.

③ Установка свойства для создания транзакционной таблицы.

Используя `DESCRIBE FORMATTED <table_name>`, можно проверить, что новая таблица была создана как транзакционная. Ниже приведена строка вывода, подтверждающая это.

```
...
| transactional                                      | true                                               |
...
```

Если вы наблюдаете следующую ошибку при создании транзакционной таблицы, убедитесь, что опция ***ACID Transactions*** равна `true` в ADCM, а сервис Hive был перезапущен.

```
This command is not allowed on an ACID table <table_name> with a non-ACID transaction manager
```

Преобразование в ACID-таблицу

Вы можете преобразовать обычную нетранзакционную таблицу Hive в таблицу со свойствами ACID. Главное требование — исходная таблица Hive должна быть типа managed, а в качестве формата хранения должен использоваться ORC. Например:

```
ALTER TABLE <non-acid_table> SET TBLPROPERTIES ("transactional"="true");
```

Чтобы наделить таблицу типа external свойствами ACID, для начала ее необходимо сделать managed-таблицей. Например:

```
ALTER TABLE <external_table_name> SET TBLPROPERTIES("EXTERNAL"="false")
```

ПРИМЕЧАНИЕ

Если таблица Hive создана как ACID-таблица, ее обратная конвертация в не-ACID таблицу невозможна.

Вставка и обновление данных

Команды записи или обновления данных для транзакционной таблицы не отличаются от команд для обычных таблиц. Синтаксис команд `INSERT`/`UPDATE`/`DELETE` — стандартный, например:

```
INSERT INTO transactions VALUES
(1, 1002, 10.00, '2023-01-01'),
(2, 1001, 100.50, '2023-01-02'),
(3, 1003, 50.00, '2023-01-03');
```

Главной особенностью при записи в транзакционную таблицу является то, что Hive создает дельта-файлы в директории HDFS warehouse. Чтобы просмотреть дельта-файлы, созданные в результате выполнения приведенного выше запроса `INSERT`, выполните команду:

```
$ hdfs dfs -ls /apps/hive/warehouse/transactions/delta_0000001_0000001_0000
```

РЕКОМЕНДАЦИЯ

Числа в имени директории *delta_0000001_0000001_0000* означают ID транзакции.

Вывод:

```
Found 3 items
-rw-r--r--   3 hive hadoop          1 2024-03-28 17:21
/apps/hive/warehouse/transactions/delta_0000001_0000001_0000/_orc_acid_version ①
-rw-r--r--   3 hive hadoop        906 2024-03-28 17:21 /apps/hive/warehouse/transactions/delta_0000001_0000001_0000/bucket_00000
②
-rw-r--r--   3 hive hadoop        959 2024-03-28 17:21 /apps/hive/warehouse/transactions/delta_0000001_0000001_0000/bucket_00002
②
```

① Файл содержит версию ORC.

② Файлы хранят данные, разделенные по бакетам.

Специальные команды транзакций

Ниже показаны несколько команд, специфичных для работы с транзакциями Hive.

COMPACT

С помощью следующей команды вы можете вручную запустить операцию минорного/мажорного сжатия:

```
ALTER TABLE <table_name> COMPACT 'major|minor';
```

Выполнение этой команды помещает операцию сжатия в очередь и выводит идентификатор операции сжатия.

```
...
Compaction enqueued with id {id}
```

Чтобы увидеть список всех операций сжатия, выполните команду:

```
SHOW COMPACTIONS;
```

Пример вывода:

```
+---------------+-----------+----------------+------------+--------+------------+-----------+-------------+---------------+------
--------+
| compactionid  |  dbname   |    tabname     |  partname  |  type  |   state    | workerid  |  starttime  |   duration    |
hadoopjobid  |
+---------------+-----------+----------------+------------+--------+------------+-----------+-------------+---------------+------
--------+
| CompactionId  | Database  | Table          | Partition  | Type   | State      | Worker    | Start Time  | Duration(ms)  |
HadoopJobId  |
| 1             | default   | transactions   |  ---       | MINOR  | initiated  |  ---      |  ---        |  ---          | None
|
+---------------+-----------+----------------+------------+--------+------------+-----------+-------------+---------------+------
--------+
```

SHOW TRANSACTIONS

Данная команда выводит список всех транзакций в системе, например:

```
+-----------------+--------------------+----------------+----------------------+-------+--------------------------------+
|      txnid      |       state        |  startedtime   |  lastheartbeattime   | user  |              host              |
+-----------------+--------------------+----------------+----------------------+-------+--------------------------------+
| Transaction ID  | Transaction State  | Started Time   | Last Heartbeat Time  | User  | Hostname                       |
| 10              | ABORTED            | 1711646536000  | 1711646536000        | hive  | ka-adh-2.ru-central1.internal  |
| 29              | OPEN               | 1711970603000  | 1711970603000        | hive  | ka-adh-2.ru-central1.internal  |
+-----------------+--------------------+----------------+----------------------+-------+--------------------------------+
```

SHOW LOCKS

С помощью этой команды вы можете увидеть все блокировки (locks) для таблицы или партиции. Пример вывода показан ниже.

```
+----------+------------------+----------------+------------+-------------+---------------+--------------+-----------------+-----
------------+----------------+-------+--------------------------------+----------------------------------------------------+
|  lockid  |     database     |     table      | partition  | lock_state  |  blocked_by   |  lock_type   | transaction_id  |
last_heartbeat  |  acquired_at   | user  |            hostname            |                     agent_info                     |
+----------+------------------+----------------+------------+-------------+---------------+--------------+-----------------+-----
------------+----------------+-------+--------------------------------+----------------------------------------------------+
| Lock ID  | Database         | Table          | Partition  | State       | Blocked By    | Type         | Transaction ID  | Last
Heartbeat  | Acquired At    | User  | Hostname                       | Agent Info                                         |
| 41.1     | _dummy_database  | _dummy_table   | NULL       | ACQUIRED    |               | SHARED_READ  | 50              | 0
| 1711992560000  | hive  | ka-adh-2.ru-central1.internal  | hive_20240401172919_fbb1738c-5736-458f-956e-d89ea7d3a727 |
| 41.2     | default          | transactions2  | NULL       | ACQUIRED    |               | SHARED_READ  | 50              | 0
| 1711992560000  | hive  | ka-adh-2.ru-central1.internal  | hive_20240401172919_fbb1738c-5736-458f-956e-d89ea7d3a727 |
+----------+------------------+----------------+------------+-------------+---------------+--------------+-----------------+-----
------------+----------------+-------+--------------------------------+----------------------------------------------------+
```

Ограничения

Существует несколько ограничений для использования транзакций Hive:

- В отличие от традиционных транзакций SQL, операции `BEGIN`, `COMMIT` и `ROLLBACK` не поддерживаются. Все транзакции Hive работают в режиме auto-commit.
- ORC — единственный поддерживаемый формат хранения.
- Чтобы использовать транзакции ACID, таблица Hive должна быть разбита на бакеты.
- Внешние (external) таблицы не могут быть транзакционными.
- Чтение/запись в ACID-таблицу из не ACID-сессии запрещены. Другими словами, для работы с ACID-таблицами должен использоваться менеджер транзакций Hive `org.apache.hadoop.hive.ql.lockmgr.DbTxnManager`.
- Менеджеры блокировок In-memory и ZooKeeper не совместимы с транзакциями.
- Использование Oracle в качестве базы данных для Metastore, а также свойства `datanucleus.connectionPoolingType=BONECP` может приводить к периодическим ошибкам "No such lock ..." и "No such transaction ...". В этом случае рекомендуется установить значение `datanucleus.connectionPoolingType=DBCP`.
- Операции `LOAD DATA` не поддерживаются для транзакционных таблиц.

Работа с таблицами Iceberg в Hive

Константин Алпашкин

Содержание

Назад к содержанию

Команды DDL
- Создание таблицы Iceberg
- CREATE TABLE AS SELECT
- CREATE TABLE LIKE
- Создание партиционированной таблицы

Команды DML
- SELECT
- INSERT

Кастомные каталоги Hive

Ограничения

Apache Iceberg — это открытый высокопроизводительный формат для больших аналитических таблиц. В сервисе ADH Hive данный формат может быть использован для повышения производительности таблиц Hive.

> **ПРИМЕЧАНИЕ**
>
> Hive поддерживает Iceberg-таблицы начиная с версии ADH 3.2.4.3.

Использование Iceberg-таблиц в Hive доступно "из коробки". Поддержка формата Iceberg основана на механизме Hive StorageHandlers. Это значит, что для создания Iceberg-таблицы в Hive необходимо указать обработчик `org.apache.iceberg.mr.hive.HiveIcebergStorageHandler` в предложении `STORED BY`. Поддержка таблиц Iceberg включена по умолчанию и может быть отключена с помощью свойства `iceberg.engine.hive.enabled` (***Clusters → <clusterName> → Services → Hive → Primary Configuration → hive-site.xml***). Также параметр `engine.hive.enabled` можно указать программно в качестве свойства таблицы в момент ее создания. Пример показан ниже:

```
Catalog catalog = ... ;
    Map<String, String> tableProperties=Maps.newHashMap();
    tableProperties.put(TableProperties.ENGINE_HIVE_ENABLED,"true");
    catalog.createTable(tableId,schema,spec,tableProperties);
```

> **ПРИМЕЧАНИЕ**
>
> Указание свойства для конкретной таблицы имеет приоритет над глобальной конфигурацией ADH.

На момент релиза ADH 3.2.4.3 поддержка Iceberg-таблиц в Hive включает в себя следующие возможности:

- Создание таблицы.
- Удаление таблицы.
- Выборка данных.
- Вставка данных.

Команды DDL

Создание таблицы Iceberg

Чтобы создать Iceberg-таблицу, используйте предложение `STORED BY 'org.apache.iceberg.mr.hive.HiveIcebergStorageHandler'` в стандартной Hive-команде `CREATE TABLE`.

Пример:

```
CREATE TABLE transactions (txn_id int, acc_id int, txn_amount decimal(10,2), txn_date date)
STORED BY 'org.apache.iceberg.mr.hive.HiveIcebergStorageHandler';
```

Выполните команду `DESCRIBE FORMATTED`, чтобы получить информацию о только что созданной таблице:

```
DESCRIBE FORMATTED transactions;
```

Пример вывода

```
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
|           col_name            |                     data_type                      |                       comment
|
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
| # col_name                    | data_type                                          | comment
|
| id                            | int                                                | from deserializer
|
| value                         | string                                             | from deserializer
|
|                               | NULL                                               | NULL
|
| # Detailed Table Information  | NULL                                               | NULL
|
| Database:                     | default                                            | NULL
|
| OwnerType:                    | USER                                               | NULL
|
| Owner:                        | hive                                               | NULL
|
| CreateTime:                   | Wed Jun 05 14:09:36 UTC 2024                       | NULL
|
| LastAccessTime:               | UNKNOWN                                            | NULL
|
| Retention:                    | 0                                                  | NULL
|
| Location:                     | hdfs://adh/apps/hive/warehouse/test_iceberg_table  | NULL
|
| Table Type:                   | MANAGED_TABLE                                      | NULL
|
| Table Parameters:             | NULL                                               | NULL
|
|                               | bucketing_version                                  | 2
|
|                               | current-schema                                     | {\"type\":\"struct\",\"schema-
id\":0,\"fields\":[{\"id\":1,\"name\":\"id\",\"required\":false,\"type\":\"int\"},
{\"id\":2,\"name\":\"value\",\"required\":false,\"type\":\"string\"}]} |
|                               | engine.hive.enabled                               | true
|
|                               | external.table.purge                              | TRUE
|
|                               | metadata_location                                  |
hdfs://adh/apps/hive/warehouse/test_iceberg_table/metadata/00000-fa65f392-3b34-4e51-915b-8e2aae261098.metadata.json |
|                               | numFiles                                           | 0
|
|                               | numRows                                            | 0
|
|                               | rawDataSize                                        | 0
|
|                               | snapshot-count                                     | 0
|
|                               | storage_handler                                    | org.apache.iceberg.mr.hive.
HiveIcebergStorageHandler |
|                               | table_type                                         | ICEBERG
|
|                               | totalSize                                          | 0
|
|                               | transient_lastDdlTime                              | 1717596576
|
|                               | uuid                                               | f9da6135-095e-491f-919e-f2194253b8f8
|
|                               | write.parquet.compression-codec                   | zstd
|
|                               | NULL                                               | NULL
|
| # Storage Information         | NULL                                               | NULL
|
| SerDe Library:                | org.apache.iceberg.mr.hive.HiveIcebergSerDe       | NULL
|
| InputFormat:                  | org.apache.iceberg.mr.hive.HiveIcebergInputFormat  | NULL
|
| OutputFormat:                 | org.apache.iceberg.mr.hive.HiveIcebergOutputFormat | NULL
|
| Compressed:                   | No                                                 | NULL
|
| Num Buckets:                  | 0                                                  | NULL
|
| Bucket Columns:               | []                                                 | NULL
|
| Sort Columns:                 | []                                                 | NULL
|
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
```

Поле `table_type` в выводе указывает на то, что таблица была создана именно как Iceberg-таблица.

CREATE TABLE AS SELECT

Вы также можете использовать синтаксис Hive `CREATE TABLE AS <tbl_name>` (CTAS) для создания таблицы Iceberg на основе уже имеющейся таблицы Hive. В этом случае основное отличие заключается в том, что таблица Iceberg создается в самом начале выполнения запроса, в то время как данные вставляются в таблицу уже после завершения запроса. Таким образом, в течение некоторого времени таблица может не содержать никаких данных.

```
CREATE TABLE transactions_ctas
STORED BY 'org.apache.iceberg.mr.hive.HiveIcebergStorageHandler' AS
SELECT * FROM transactions;
```

Пример вывода команды DESCRIBE FORMATTED

```
DESCRIBE FORMATTED transactions_ctas;
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
|           col_name            |                     data_type                      |                       comment
|
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
| # col_name                    | data_type                                          | comment
|
| id                            | int                                                | from deserializer
|
|                               | NULL                                               | NULL
|
| # Detailed Table Information  | NULL                                               | NULL
|
| Database:                     | default                                            | NULL
|
| OwnerType:                    | USER                                               | NULL
|
| Owner:                        | hive                                               | NULL
|
| CreateTime:                   | Mon Jun 03 20:59:25 UTC 2024                       | NULL
|
| LastAccessTime:               | UNKNOWN                                            | NULL
|
| Retention:                    | 0                                                  | NULL
|
| Location:                     | hdfs://adh/apps/hive/warehouse/test_iceberg_table1 | NULL
|
| Table Type:                   | MANAGED_TABLE                                      | NULL
|
| Table Parameters:             | NULL                                               | NULL
|
|                               | current-schema                                     | {\"type\":\"struct\",\"schema-
id\":0,\"fields\":[{\"id\":1,\"name\":\"id\",\"required\":false,\"type\":\"int\"}]} |
|                               | engine.hive.enabled                               | true
|
|                               | external.table.purge                              | TRUE
|
|                               | metadata_location                                  |
hdfs://adh/apps/hive/warehouse/test_iceberg_table1/metadata/00000-df221cec-ab76-4de9-95cd-ef7605c8e79f.metadata.json |
|                               | numFiles                                           | 0
|
|                               | numRows                                            | 0
|
|                               | rawDataSize                                        | 0
|
|                               | snapshot-count                                     | 0
|
|                               | storage_handler                                    | org.apache.iceberg.mr.hive.
HiveIcebergStorageHandler |
|                               | table_type                                         | ICEBERG
|
|                               | totalSize                                          | 0
|
|                               | transient_lastDdlTime                              | 1717448365
|
|                               | uuid                                               | eb1933a0-43d9-4dd2-9426-cf7ace5063c8
|
|                               | write.parquet.compression-codec                   | zstd
|
|                               | NULL                                               | NULL
|
| # Storage Information         | NULL                                               | NULL
|
| SerDe Library:                | org.apache.iceberg.mr.hive.HiveIcebergSerDe       | NULL
|
| InputFormat:                  | org.apache.iceberg.mr.hive.HiveIcebergInputFormat  | NULL
|
| OutputFormat:                 | org.apache.iceberg.mr.hive.HiveIcebergOutputFormat | NULL
|
| Compressed:                   | No                                                 | NULL
|
| Num Buckets:                  | 0                                                  | NULL
|
| Bucket Columns:               | []                                                 | NULL
|
| Sort Columns:                 | []                                                 | NULL
|
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
```

CREATE TABLE LIKE

Вы также можете использовать синтаксис `CREATE TABLE LIKE <tbl_name>` для создания пустой таблицы Iceberg, используя структуру существующей таблицы Hive.

Пример:

```
CREATE TABLE transactions_ctlt LIKE transactions
STORED BY 'org.apache.iceberg.mr.hive.HiveIcebergStorageHandler';
```

Пример вывода команды DESCRIBE FORMATTED

```
DESCRIBE FORMATTED transactions_ctlt;
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
|           col_name            |                     data_type                      |                       comment
|
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
| # col_name                    | data_type                                          | comment
|
| txn_id                        | int                                                | from deserializer
|
| txn_amount                    | decimal(10,2)                                      | from deserializer
|
| txn_date                      | date                                               | from deserializer
|
| acc_id                        | int                                                | from deserializer
|
|                               | NULL                                               | NULL
|
| # Detailed Table Information  | NULL                                               | NULL
|
| Database:                     | default                                            | NULL
|
| OwnerType:                    | USER                                               | NULL
|
| Owner:                        | hive                                               | NULL
|
| CreateTime:                   | Thu Jun 06 13:57:31 UTC 2024                       | NULL
|
| LastAccessTime:               | UNKNOWN                                            | NULL
|
| Retention:                    | 0                                                  | NULL
|
| Location:                     | hdfs://adh/apps/hive/warehouse/transactions        | NULL
|
| Table Type:                   | MANAGED_TABLE                                      | NULL
|
| Table Parameters:             | NULL                                               | NULL
|
|                               | bucketing_version                                  | 2
|
|                               | current-schema                                     | {\"type\":\"struct\",\"schema-
id\":0,\"fields\":[{\"id\":1,\"name\":\"txn_id\",\"required\":false,\"type\":\"int\"},
{\"id\":2,\"name\":\"txn_amount\",\"required\":false,\"type\":\"decimal(10, 2)\"},
{\"id\":3,\"name\":\"txn_date\",\"required\":false,\"type\":\"date\"},
{\"id\":4,\"name\":\"acc_id\",\"required\":false,\"type\":\"int\"}]} |
|                               | default-partition-spec                             | {\"spec-id\":0,\"fields\":
[{\"name\":\"acc_id\",\"transform\":\"identity\",\"source-id\":4,\"field-id\":1000}]} |
|                               | engine.hive.enabled                               | true
|
|                               | external.table.purge                              | TRUE
|
|                               | metadata_location                                  |
hdfs://adh/apps/hive/warehouse/transactions/metadata/00000-b6dde9b3-d1cc-4f92-828e-5dab83b1e5e8.metadata.json |
|                               | snapshot-count                                     | 0
|
|                               | storage_handler                                    | org.apache.iceberg.mr.hive.
HiveIcebergStorageHandler |
|                               | table_type                                         | ICEBERG
|
|                               | transient_lastDdlTime                              | 1717682251
|
|                               | uuid                                               | c6205d6d-ddca-4b04-a780-8b8e1048db0b
|
|                               | write.parquet.compression-codec                   | zstd
|
|                               | NULL                                               | NULL
|
| # Storage Information         | NULL                                               | NULL
|
| SerDe Library:                | org.apache.iceberg.mr.hive.HiveIcebergSerDe       | NULL
|
| InputFormat:                  | org.apache.iceberg.mr.hive.HiveIcebergInputFormat  | NULL
|
| OutputFormat:                 | org.apache.iceberg.mr.hive.HiveIcebergOutputFormat | NULL
|
| Compressed:                   | No                                                 | NULL
|
| Num Buckets:                  | 0                                                  | NULL
|
| Bucket Columns:               | []                                                 | NULL
|
| Sort Columns:                 | []                                                 | NULL
|
+-------------------------------+----------------------------------------------------+----------------------------------------
------------
```

Создание партиционированной таблицы

Чтобы создать партиционированную таблицу, используйте синтаксис, как показано в следующем примере:

```
CREATE TABLE transactions_partitioned (
    txn_id int, txn_amount decimal(10,2), txn_date date)
PARTITIONED BY (acc_id int)
STORED BY 'org.apache.iceberg.mr.hive.HiveIcebergStorageHandler';
```

В выводе команды `DESCRIBE FORMATTED` найдите свойство `default-partition-spec`. В этом поле хранится спецификация партиций в формате Iceberg.

Пример вывода команды DESCRIBE FORMATTED

```
DESCRIBE FORMATTED transactions_partitioned;
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
|           col_name            |                     data_type                      |                       comment
|
+-------------------------------+----------------------------------------------------+----------------------------------------
------------+
| # col_name                    | data_type                                          | comment
|
| txn_id                        | int                                                | from deserializer
|
| txn_amount                    | decimal(10,2)                                      | from deserializer
|
| txn_date                      | date                                               | from deserializer
|
| acc_id                        | int                                                | from deserializer
|
|                               | NULL                                               | NULL
|
| # Detailed Table Information  | NULL                                               | NULL
|
| Database:                     | default                                            | NULL
|
| OwnerType:                    | USER                                               | NULL
|
| Owner:                        | hive                                               | NULL
|
| CreateTime:                   | Thu Jun 06 13:57:31 UTC 2024                       | NULL
|
| LastAccessTime:               | UNKNOWN                                            | NULL
|
| Retention:                    | 0                                                  | NULL
|
| Location:                     | hdfs://adh/apps/hive/warehouse/transactions        | NULL
|
| Table Type:                   | MANAGED_TABLE                                      | NULL
|
| Table Parameters:             | NULL                                               | NULL
|
|                               | bucketing_version                                  | 2
|
|                               | current-schema                                     | {\"type\":\"struct\",\"schema-
id\":0,\"fields\":[{\"id\":1,\"name\":\"txn_id\",\"required\":false,\"type\":\"int\"},
{\"id\":2,\"name\":\"txn_amount\",\"required\":false,\"type\":\"decimal(10, 2)\"},
{\"id\":3,\"name\":\"txn_date\",\"required\":false,\"type\":\"date\"},
{\"id\":4,\"name\":\"acc_id\",\"required\":false,\"type\":\"int\"}]} |
|                               | default-partition-spec                             | {\"spec-id\":0,\"fields\":
[{\"name\":\"acc_id\",\"transform\":\"identity\",\"source-id\":4,\"field-id\":1000}]} |
|                               | engine.hive.enabled                               | true
|
|                               | external.table.purge                              | TRUE
|
|                               | metadata_location                                  |
hdfs://adh/apps/hive/warehouse/transactions/metadata/00000-b6dde9b3-d1cc-4f92-828e-5dab83b1e5e8.metadata.json |
|                               | snapshot-count                                     | 0
|
|                               | storage_handler                                    | org.apache.iceberg.mr.hive.
HiveIcebergStorageHandler |
|                               | table_type                                         | ICEBERG
|
|                               | transient_lastDdlTime                              | 1717682251
|
|                               | uuid                                               | c6205d6d-ddca-4b04-a780-8b8e1048db0b
|
|                               | write.parquet.compression-codec                   | zstd
|
|                               | NULL                                               | NULL
|
| # Storage Information         | NULL                                               | NULL
|
| SerDe Library:                | org.apache.iceberg.mr.hive.HiveIcebergSerDe       | NULL
|
| InputFormat:                  | org.apache.iceberg.mr.hive.HiveIcebergInputFormat  | NULL
|
| OutputFormat:                 | org.apache.iceberg.mr.hive.HiveIcebergOutputFormat | NULL
|
| Compressed:                   | No                                                 | NULL
|
| Num Buckets:                  | 0                                                  | NULL
|
| Bucket Columns:               | []                                                 | NULL
|
| Sort Columns:                 | []                                                 | NULL
|
+-------------------------------+----------------------------------------------------+----------------------------------------
------------
```

Команды DML

На момент релиза ADH 3.2.4.3 для Iceberg-таблиц в Hive доступны DML-операции `SELECT` и `INSERT`.

> **ПРИМЕЧАНИЕ**
>
> Для корректной работы DML-команд для таблиц Iceberg в Hive должен использоваться движок MapReduce.

Чтобы сменить движок Hive, используйте свойство `hive.execution.engine` в ADCM (***Clusters → <clusterName> → Services → Hive → Primary Configuration → hive-site.xml***). После изменения свойства необходим рестарт Hive.

SELECT

Синтаксис команды `SELECT` для работы с Iceberg-таблицами идентичен обычной команде `SELECT` в Hive. Однако под капотом Hive выполняет специфичный набор действий для получения всех преимуществ формата Iceberg. Отличия присутствуют как на этапе компиляции запроса, так и на этапе его выполнения.

Ниже приведено краткое описание алгоритма выборки данных из Iceberg-таблиц:

1. Hive обращается к каталогу Iceberg для получения **файла метаданных** (metadata file) таблицы. Данный файл описывает структуру таблицы Iceberg. Каталог Iceberg возвращает указатель на файл метаданных.
2. Hive анализирует файл метаданных и извлекает запись **manifest list**. В этой записи содержится информация о **файлах манифеста (manifest files)**, которые вместе составляют единый снепшот, определяют партиции, их границы и так далее. Если таблица разбита на партиции, Hive также считывает спецификацию партиционирования (partition spec). Это позволяет выполнить определенные шаги по оптимизации уже на данном этапе.
3. Далее Hive анализирует каждый отдельный файл манифеста из списка. В каждом таком файле содержится информация о файлах данных, где физически хранятся данные таблицы.
4. Наконец, Hive сканирует сами файлы данных. Если таблица разбита на партиции, Hive обрабатывает только те файлы данных, которые соответствуют предложению `WHERE` или другим фильтрам. Основное преимущество в производительности формата Iceberg заключается именно в этой фазе — к этому моменту движок располагает достаточной информацией для сканирования только тех файлов с данными, которые релевантны указанному запросу `SELECT`. Такое поведение возможно благодаря тому, что данные отслеживаются на уровне файлов (а не на уровне директорий, как это реализовано в традиционной модели Hive).

INSERT

Iceberg-таблицы в Hive поддерживают классическую команду `INSERT INTO`.

Пример:

```
INSERT INTO transactions VALUES
(1, 1002, 10.00, '2023-01-01'),
(2, 1002, 20.00, '2023-01-03'),
(3, 1002, 30.00, '2023-01-02'),
(4, 1001, 100.50, '2023-01-02'),
(5, 1001, 150.50, '2023-01-04'),
(6, 1001, 200.50, '2023-01-03'),
(7, 1003, 50.00, '2023-01-03'),
(8, 1003, 50.00, '2023-01-01'),
(9, 1003, 75.00, '2023-01-04');
```

Также поддерживается многотабличная вставка данных, однако такие вставки не являются атомарными. Коммиты транзакций выполняются для каждой таблицы. В процессе вставки клиентские приложения могут "видеть" частичные обновления содержимого таблицы, а ошибки могут приводить к частичной записи данных. Изменения в рамках одной таблицы являются атомарной операцией.

В следующем примере показано наполнение двух Iceberg-таблиц данными из имеющейся таблицы Hive с использованием одной операции многотабличной вставки.

```
FROM transactions
   INSERT INTO transactions11 SELECT txn_id, txn_date
   INSERT INTO transactions22 SELECT txn_id, acc_id;
```

Кастомные каталоги Hive

В контексте Hive существует только один глобальный каталог данных — `HiveCatalog`, согласно которому данные хранятся в */apps/hive/warehouse/* директории в HDFS. Однако при использовании Iceberg-таблиц можно добавить несколько различных каталогов данных, например, `Hive`, `Hadoop` или их кастомные реализации.

Таблицы Iceberg можно создавать без привязки к определенному каталогу, указывая лишь директорию HDFS для хранения данных. Такие кросс-каталоговые таблицы могут использоваться в качестве временных или промежуточных таблиц, например, для операций JOIN.

В следующем примере создается дополнительный каталог Hive и указывается кастомная локация для хранения файлов, принадлежащих этому каталогу.

```
SET iceberg.catalog.my_test_hive.type=hive;
SET iceberg.catalog.my_test_hive.warehouse=hdfs:/adh/apps/hive/custom_catalog/warehouse;
```

Ограничения

Существуют некоторые ограничения при работе с таблицами Iceberg в Hive. Данные ограничения связаны с версией Hive 3.x, используемой в ADH. В ближайших релизах ADH с переходом на версию Hive 4.x поддержка Iceberg таблиц будет улучшена, а перечисленные ниже ограничения сняты.

- На момент релиза ADH 3.2.4.3 доступны следующие операции DML/DDL:
  - `CREATE TABLE`
  - `DROP TABLE`
  - `SELECT`
  - `INSERT`
- На момент релиза ADH 3.2.4.3 для создания Iceberg-таблиц в Hive используется предложение `STORED BY 'org.apache.iceberg.mr.hive.HiveIcebergStorageHandler'`, а не `STORED BY ICEBERG`.
- Начиная с версии Apache Iceberg 0.11.0 при использовании движка Tez для Hive необходимо также отключить векторизацию (`hive.vectorized.execution.enabled=false`).

Управление сервисом Hive через ADCM

Константин Алпашкин

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Hive. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Hive, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Hive
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Manage Ranger plugin	Включает или отключает плагин Ranger для Hive

Действия с компонентами Hive

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Обзор HUE UI

Владимир Адаменков

Содержание

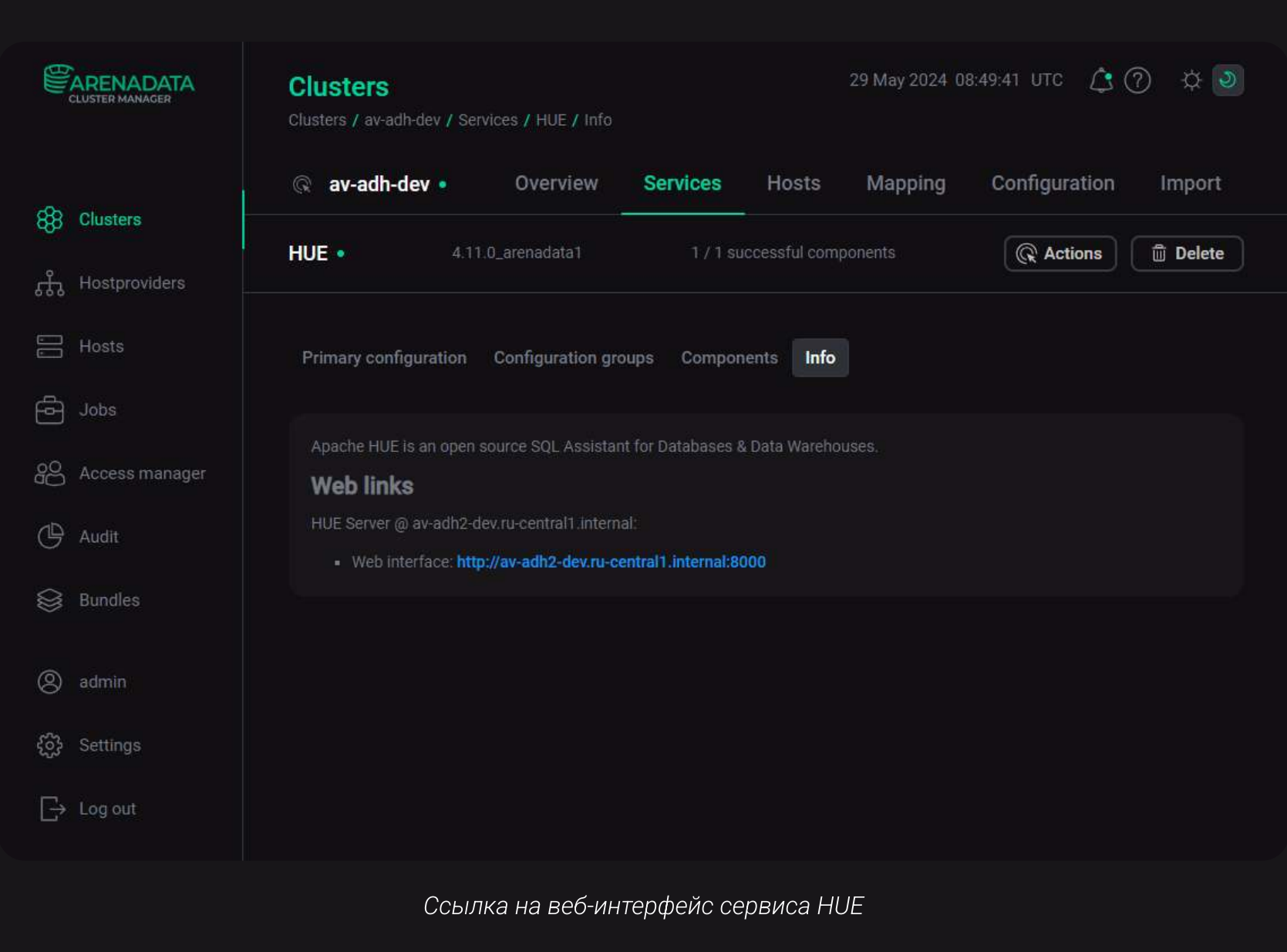
Назад к содержанию

- Editor
- Documents
- Files
- Ozone
- Tables
  - Вкладка Overview
  - Вкладка Sample
  - Вкладка Details
- Jobs
- Importer
  - Импорт удаленного файла
  - Импорт внешней БД
  - Ручной импорт данных

Сервис HUE имеет веб-интерфейс, позволяющий удобно взаимодействовать с различными базами данных и хранилищами данных. В дистрибутиве ADH он содержит преднастроенные интерпретаторы SQL для следующих решений:

- Hive
- Impala
- PySpark
- Trino
- Kyuubi (с преднастроенным Spark3)
- Flink 2

Ссылку на веб-интерфейс HUE можно найти на вкладке ***Info*** сервиса HUE в веб-интерфейсе ADCM.



ARENADATA CLUSTER MANAGER
Clusters
Clusters / av-adh-dev / Services / HUE / Info
29 May 2024 08:49:41 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
av-adh-dev
Overview
Services
Hosts
Mapping
Configuration
Import
HUE
4.11.0_arenadata1
1 / 1 successful components
Actions
Delete
Primary configuration
Configuration groups
Components
Info

Apache HUE is an open source SQL Assistant for Databases & Data Warehouses.

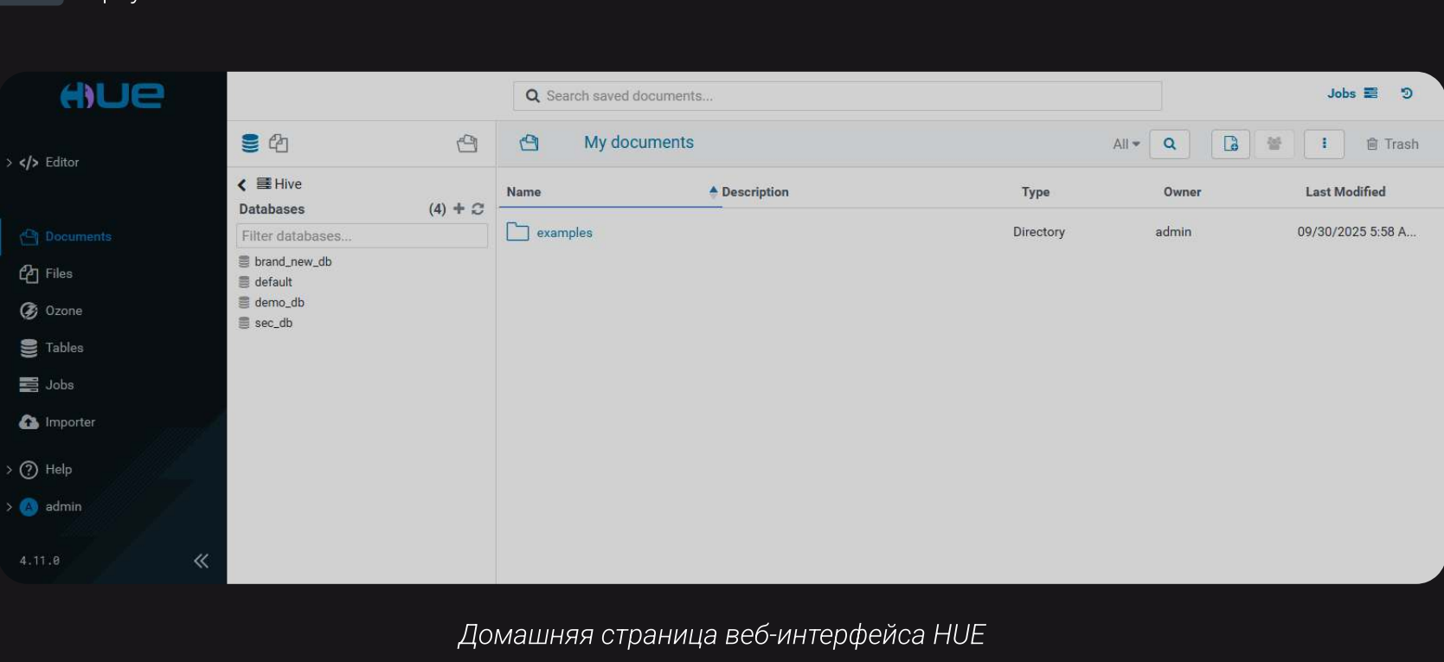
Web links

HUE Server @ av-adh2-dev.ru-central1.internal:

- Web interface: http://av-adh2-dev.ru-central1.internal:8000

Ссылка на веб-интерфейс сервиса HUE

Веб-сервер HUE работает на `8000` порту.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
admin
4.11.0
Search saved documents...
Jobs
Hive
Databases
(4)
Filter databases...
brand_new_db
default
demo_db
sec_db
My documents
All
Trash

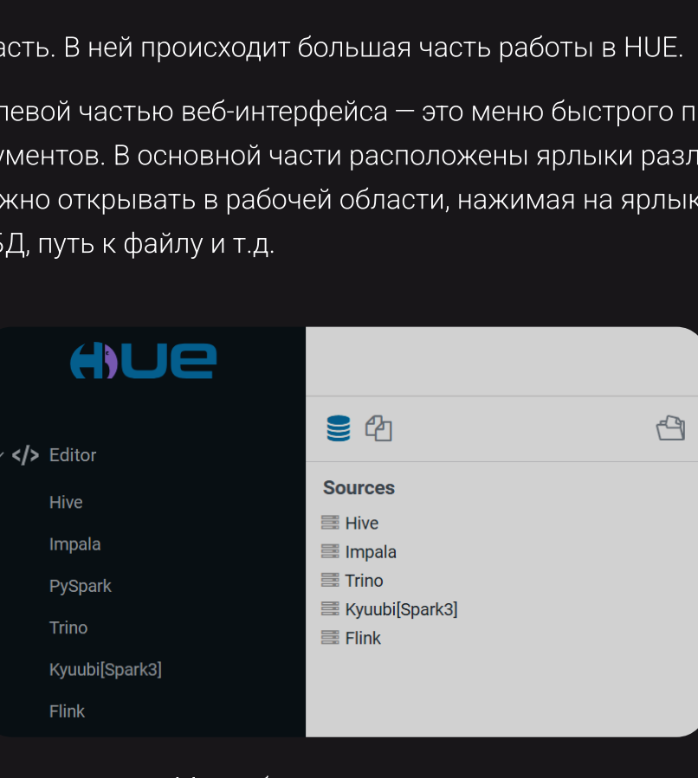
Name	Description	Type	Owner	Last Modified
examples		Directory	admin	09/30/2025 5:58 A...

Домашняя страница веб-интерфейса HUE

Верхняя прямоугольная область домашней страницы содержит строку поиска документов, кнопки браузера и предпросмотра заданий, и кнопку истории заданий.

Центральную часть веб-интерфейса занимает рабочая область. В ней происходит большая часть работы в HUE.

Сворачиваемое меню между рабочей областью и крайней левой частью веб-интерфейса — это меню быстрого просмотра. В его верхней части расположен переключатель между списками баз данных, файлов и документов. В основной части расположены ярлыки различных данных: элементов БД, файлов, документов — в зависимости от положения переключателя. Эти данные можно открывать в рабочей области, нажимая на ярлыки. Также можно переносить (используя "drag and drop") ярлыки в рабочую область, чтобы вставить имя элемента БД, путь к файлу и т.д.



HUE
Editor
Hive
Impala
PySpark
Trino
Kyuubi[Spark3]
Flink
Sources
Hive
Impala
Trino
Kyuubi[Spark3]
Flink

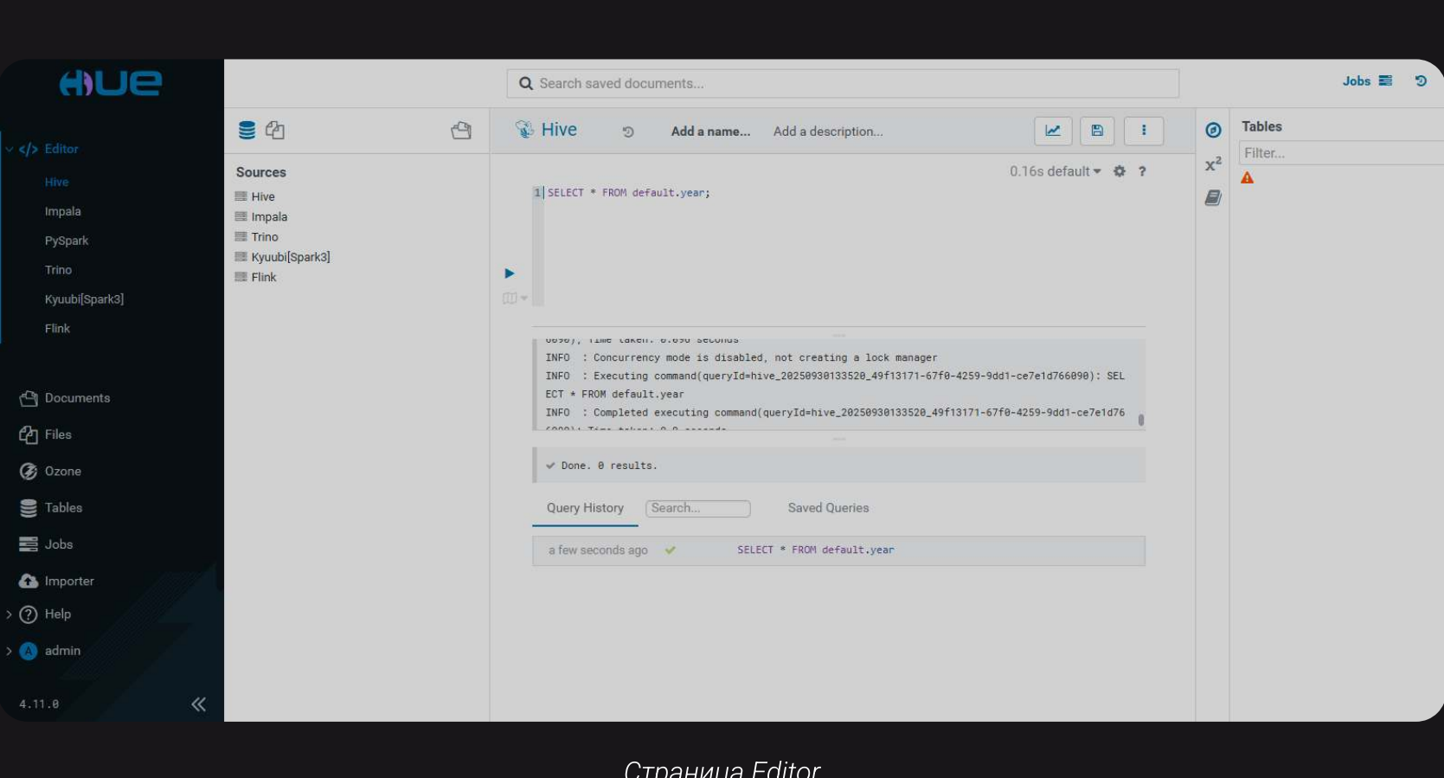
Меню быстрого просмотра

Сворачиваемое меню в крайней левой части содержит следующие страницы:

- Editor
- Documents
- Files
- Ozone
- Tables
- Jobs
- Importer

Editor

Страница ***Editor*** выглядит следующим образом.



HUE
Editor
Hive
Impala
PySpark
Trino
Kyuubi[Spark3]
Flink
Documents
Files
Ozone
Tables
Jobs
Importer
Help
admin
4.11.0
Sources
Search saved documents...
Hive
Add a name...
Add a description...
Jobs
Tables
Filter...
Query History
Saved Queries

```
SELECT * FROM default.year;
```

```
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20250930133520_49f13171-67f0-4259-9dd1-ce7e1d766090): SEL
ECT * FROM default.year
INFO  : Completed executing command(queryId=hive_20250930133520_49f13171-67f0-4259-9dd1-ce7e1d76
```

✓ Done. 0 results.

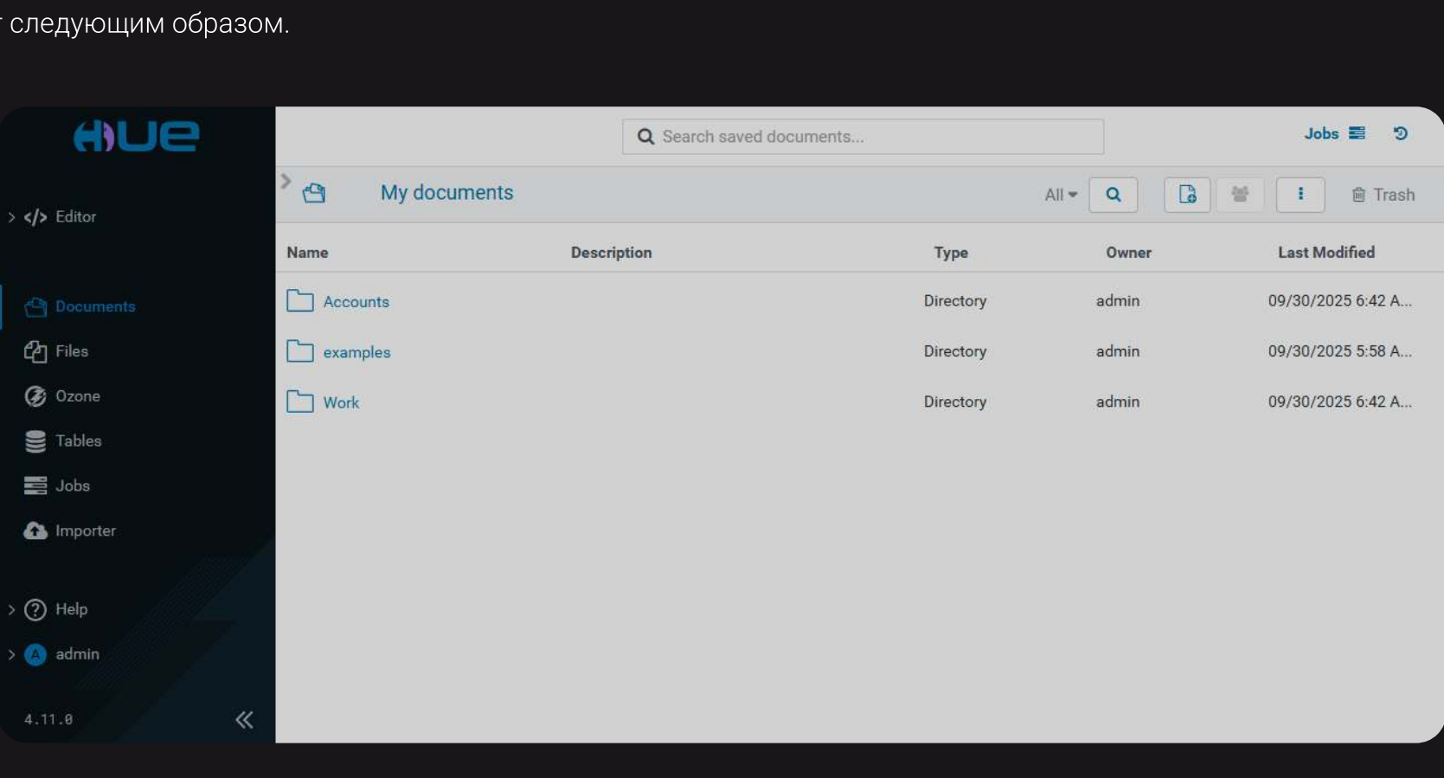
a few seconds ago ✓ SELECT * FROM default.year

Страница Editor

На странице ***Editor*** расположена рабочая область, в которой можно составлять и выполнять SQL-запросы, в том числе с ключевым словом `EXPLAIN`. В качестве источников данных используются ***impala***, ***hive*** и ***kyuubi***. Также можно использовать режим блокнота (***notebook***) для создания и хранения SQL-сниппетов.

Documents

Страница ***Documents*** выглядит следующим образом.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
admin
4.11.0
Search saved documents...
Jobs
My documents
All
Trash

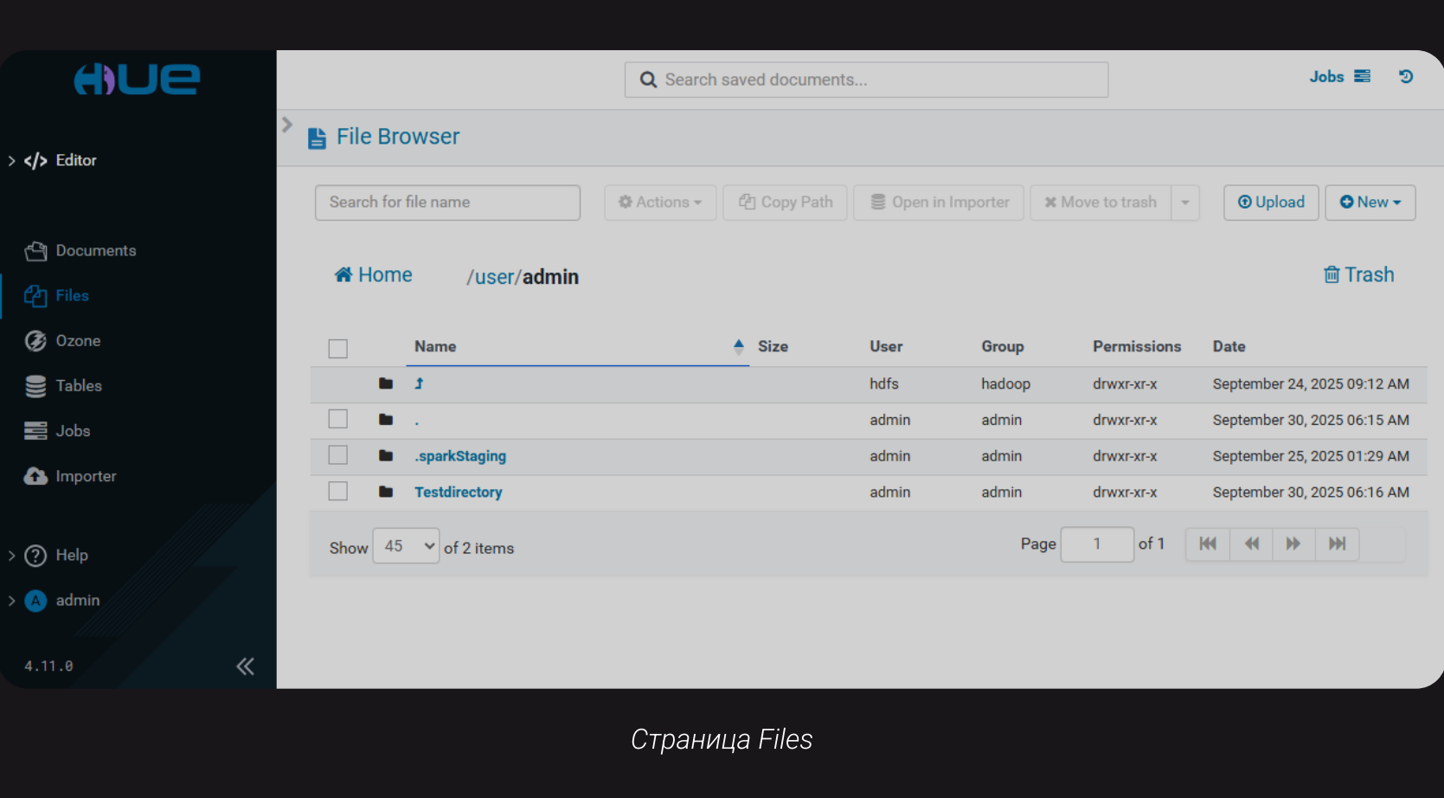
Name	Description	Type	Owner	Last Modified
Accounts		Directory	admin	09/30/2025 6:42 A...
examples		Directory	admin	09/30/2025 5:58 A...
Work		Directory	admin	09/30/2025 6:42 A...

Страница Documents

На странице ***Documents*** содержатся ваши документы и каталоги. Здесь можно создавать документы, делиться и управлять ими, а также просматривать корзину, используя соответствующие кнопки.

Files

Страница ***Files*** выглядит следующим образом.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
admin
4.11.0
Search saved documents...
Jobs
File Browser
Search for file name
Actions
Copy Path
Open in Importer
Move to trash
Upload
New
Home
/user/admin
Trash
Show 45 of 2 items
Page 1 of 1

Name	Size	User	Group	Permissions	Date
↑		hdfs	hadoop	drwxr-xr-x	September 24, 2025 09:12 AM
.		admin	admin	drwxr-xr-x	September 30, 2025 06:15 AM
.sparkStaging		admin	admin	drwxr-xr-x	September 25, 2025 01:29 AM
Testdirectory		admin	admin	drwxr-xr-x	September 30, 2025 06:16 AM

Страница Files

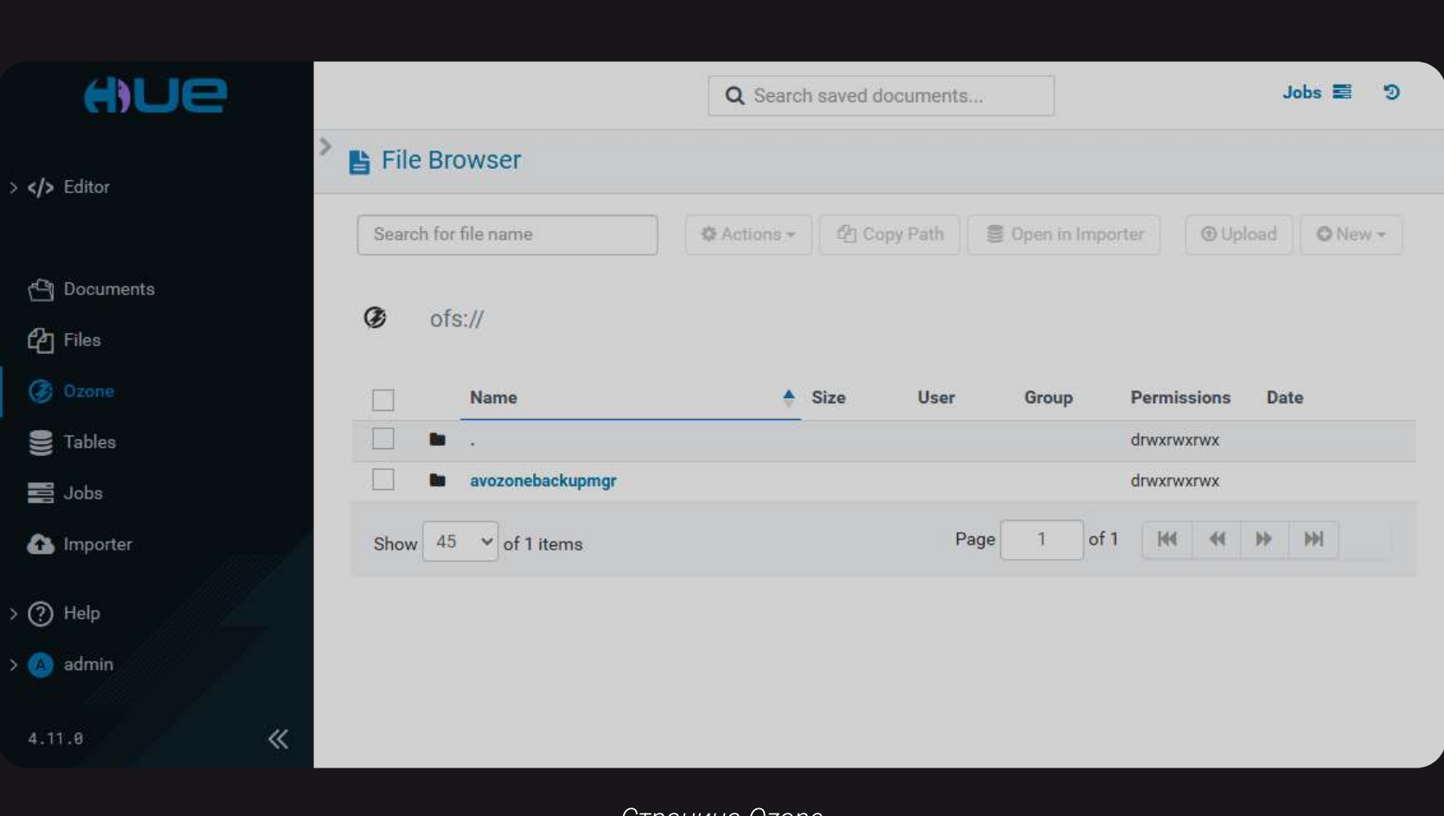
На странице ***Files*** расположен браузер файловой системы Hadoop. Здесь можно искать файлы и каталоги, а также управлять ими. Доступны следующие действия:

- переименовать;
- переместить;
- копировать;
- загрузить из Hadoop;
- изменить разрешения;
- просмотреть сведения;
- изменить фактор репликации;
- переместить в корзину;
- удалить навсегда.

Кроме того, на этой странице можно создать или загрузить в Hadoop новые файлы, а также скопировать путь к файлу или открыть его на странице ***Importer***.

Ozone

Страница ***Ozone*** выглядит следующим образом.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
admin
4.11.0
Search saved documents...
Jobs
File Browser
Search for file name
Actions
Copy Path
Open in Importer
Upload
New
ofs://
Show 45 of 1 items
Page 1 of 1

Name	Size	User	Group	Permissions	Date
.				drwxrwxrwx	
avozonebackupmgr				drwxrwxrwx	

Страница Ozone

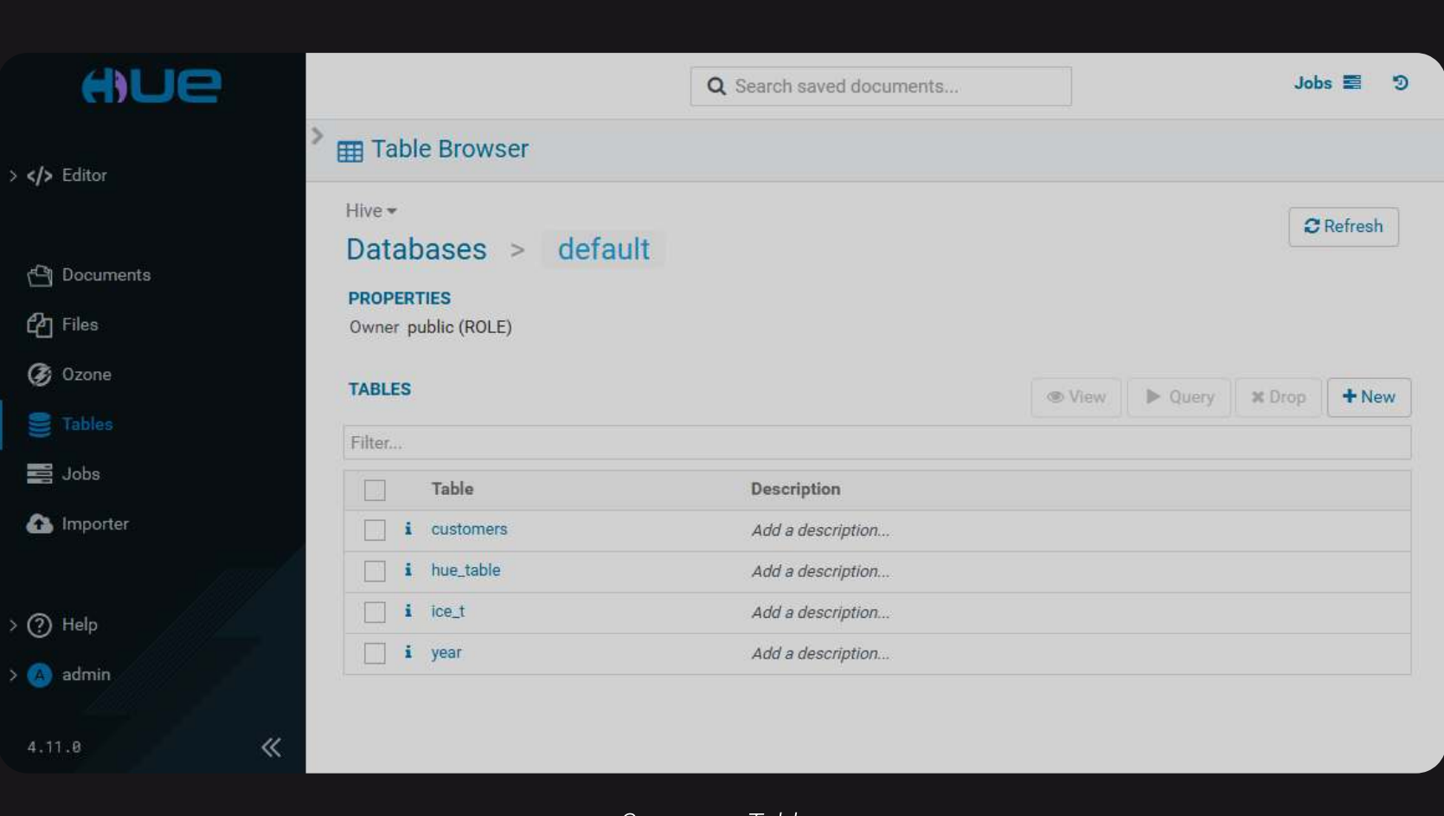
На странице ***Ozone*** расположен браузер файловой системы OFS (Ozone file system). Здесь можно искать файлы и каталоги, а также управлять ими. Доступны следующие действия:

- переименовать;
- переместить;
- копировать;
- загрузить из OFS;
- изменить разрешения;
- просмотреть сведения;
- удалить навсегда.

Кроме того, на этой странице можно создать или загрузить в OFS новые файлы, а также скопировать путь к файлу или открыть его на странице ***Importer***.

Tables

Страница ***Tables*** выглядит следующим образом.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
admin
4.11.0
Search saved documents...
Jobs
Table Browser
Hive
Databases > default
Refresh
PROPERTIES
Owner public (ROLE)
TABLES
View
Query
Drop
New
Filter...

Table	Description
customers	Add a description...
hue_table	Add a description...
ice_t	Add a description...
year	Add a description...

Страница Tables

На странице ***Tables*** можно выбрать источник БД и просматривать таблицы доступных БД. При выборе таблицы становятся доступными следующие действия:

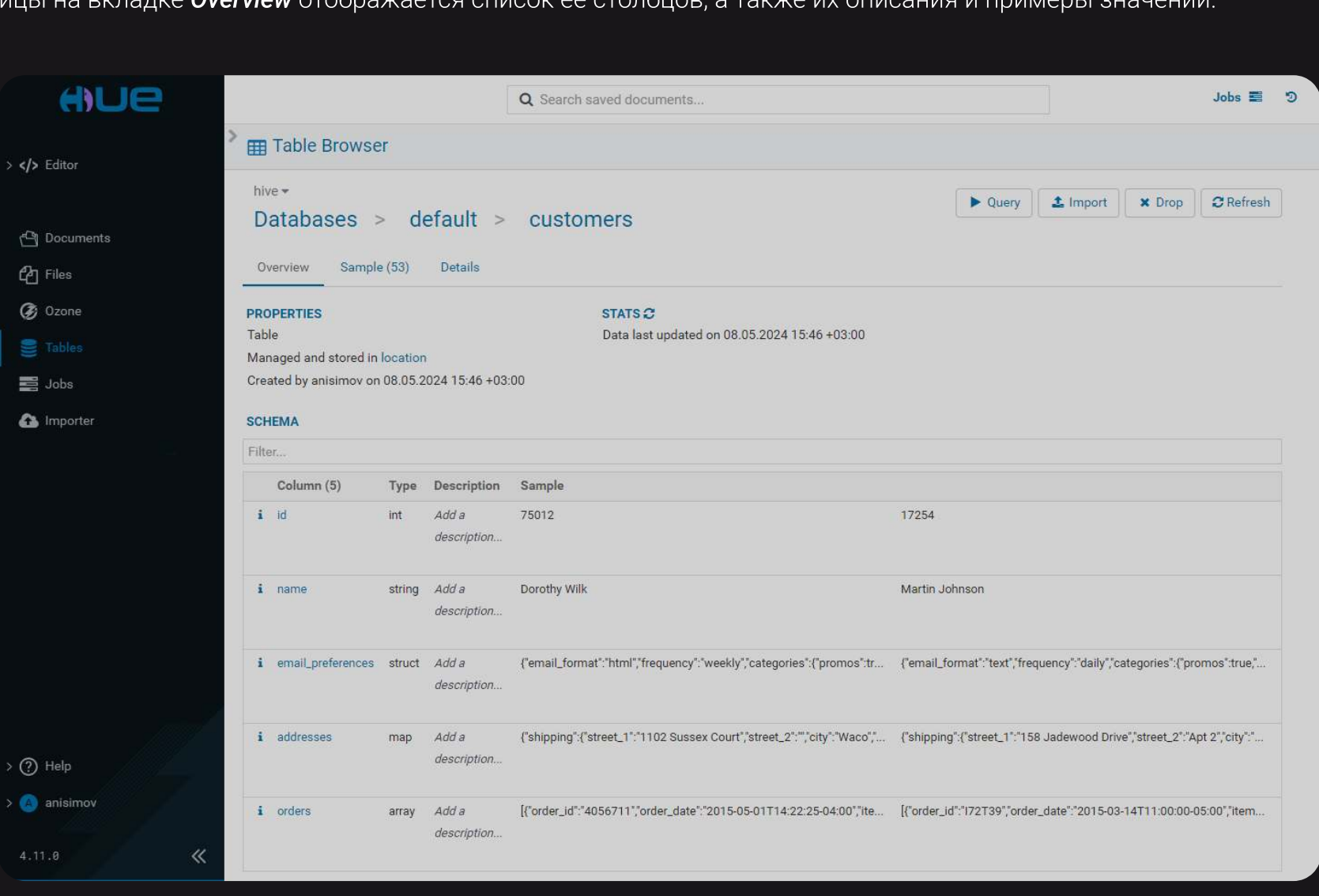
- ***Query*** — выполнить следующий запрос в выбранной таблице:

```
SELECT * FROM <selected_db>.<selected_table> LIMIT 100;
```

- ***Import*** — импортировать данные из файловой системы Hadoop. Во время этого действия данные перемещаются из своего расположения в место размещения выбранной таблицы. С помощью этого действия также можно перезаписать имеющиеся данные.
- ***Drop*** — удалить выбранную таблицу.
- ***Refresh*** — обновить отображаемые данные для выбранной таблицы.

Вкладка Overview

При выборе той или иной таблицы на вкладке ***Overview*** отображается список её столбцов, а также их описания и примеры значений.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
anisimov
4.11.0
Search saved documents...
Jobs
Table Browser
hive
Databases > default > customers
Query
Import
Drop
Refresh
Overview
Sample (53)
Details
PROPERTIES
STATS
SCHEMA
Filter...

Table

Managed and stored in location

Created by anisimov on 08.05.2024 15:46 +03:00

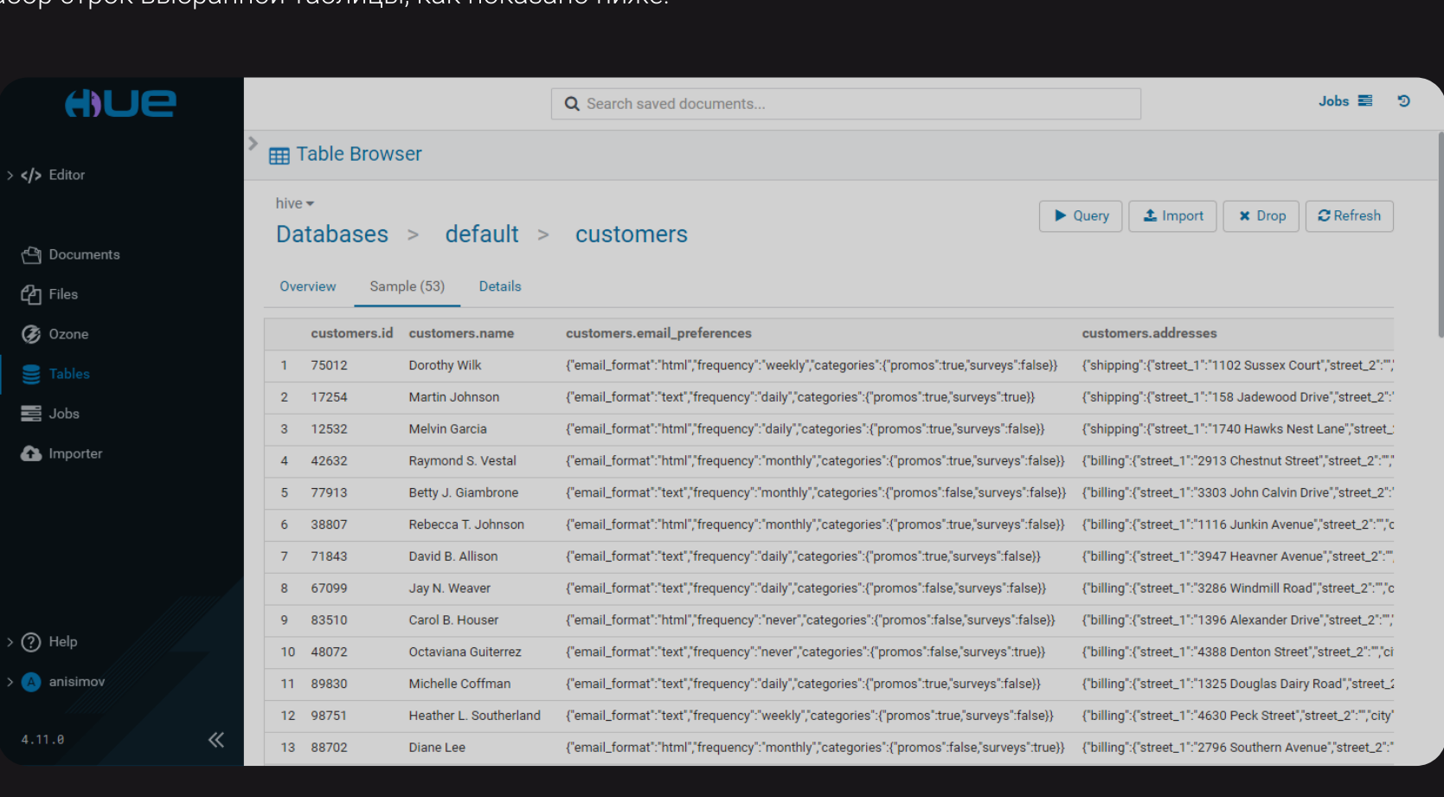
Data last updated on 08.05.2024 15:46 +03:00

Column (5)	Type	Description	Sample	
id	int	Add a description...	75012	17254
name	string	Add a description...	Dorothy Wilk	Martin Johnson
email_preferences	struct	Add a description...	{"email_format":"html","frequency":"weekly","categories":{"promos":tr...	{"email_format":"text","frequency":"daily","categories":{"promos":true,"...
addresses	map	Add a description...	{"shipping":{"street_1":"1102 Sussex Court","street_2":"","city":"Waco",...	{"shipping":{"street_1":"158 Jadewood Drive","street_2":"Apt 2","city":"...
orders	array	Add a description...	[{"order_id":"4056711","order_date":"2015-05-01T14:22:25-04:00","ite...	[{"order_id":"172T39","order_date":"2015-03-14T11:00:00-05:00","item...

Вкладка Overview для таблицы

Вкладка Sample

Вкладка ***Sample*** отображает набор строк выбранной таблицы, как показано ниже.



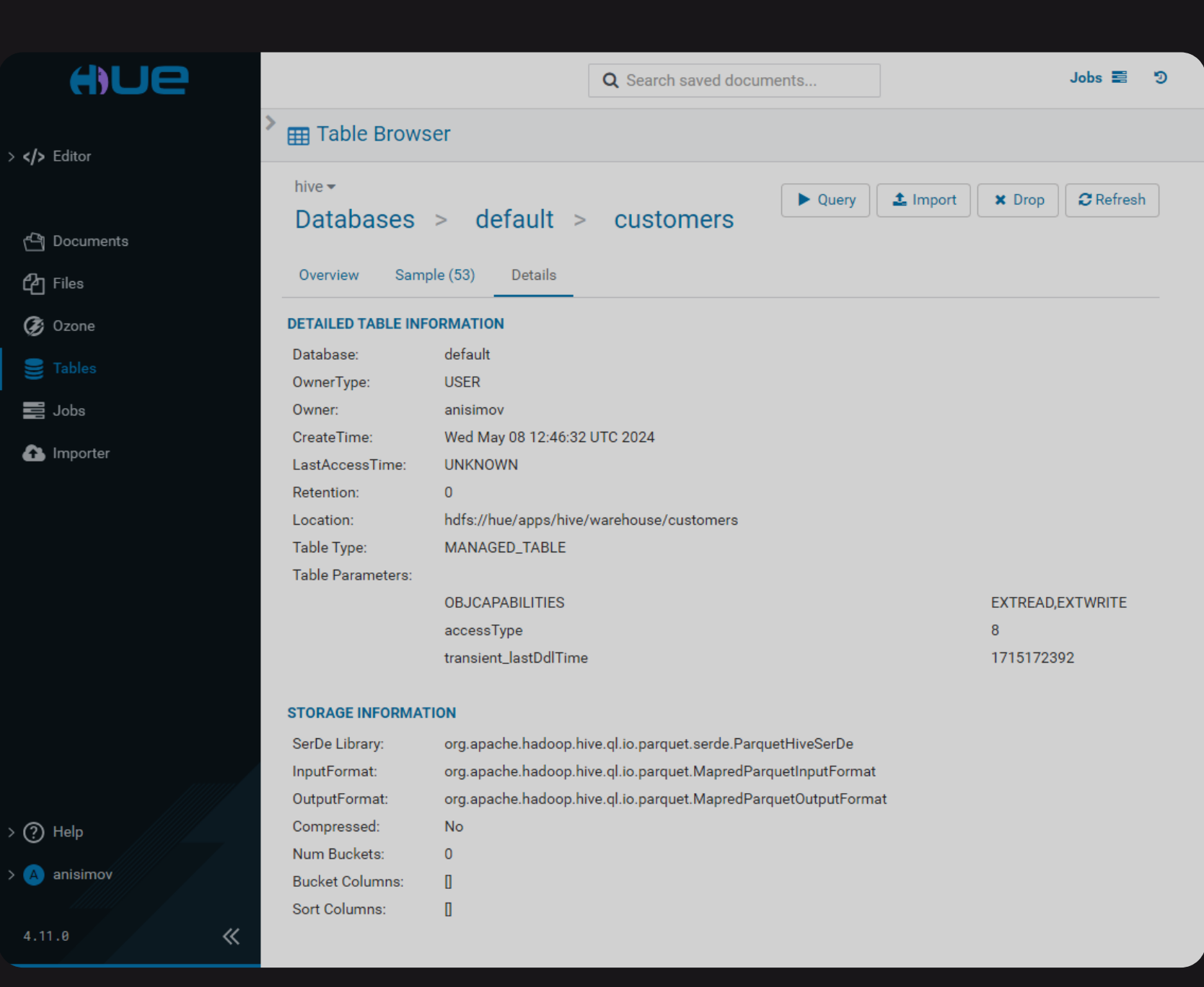
HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
anisimov
4.11.0
Search saved documents...
Jobs
Table Browser
hive
Databases > default > customers
Query
Import
Drop
Refresh
Overview
Sample (53)
Details

	customers.id	customers.name
1	75012	Dorothy Wilk
2	17254	Martin Johnson
3	12532	Melvin Garcia
4	42632	Raymond S. Vestal
5	77913	Betty J. Giambrone
6	38807	Rebecca T. Johnson
7	71843	David B. Allison
8	67099	Jay N. Weaver
9	83510	Carol B. Houser
10	48072	Octaviana Guiterrez
11	89830	Michelle Coffman
12	98751	Heather L. Southerland
13	88702	Diane Lee

Вкладка Sample для таблицы

Вкладка Details

Вкладка ***Details*** отображает различные сведения о выбранной таблице, как показано ниже.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
anisimov
4.11.0
Search saved documents...
Jobs
Table Browser
hive
Databases > default > customers
Query
Import
Drop
Refresh
Overview
Sample (53)
Details

DETAILED TABLE INFORMATION

Database: default
OwnerType: USER
Owner: anisimov
CreateTime: Wed May 08 12:46:32 UTC 2024
LastAccessTime: UNKNOWN
Retention: 0
Location: hdfs://hue/apps/hive/warehouse/customers
Table Type: MANAGED_TABLE
Table Parameters:
OBJCAPABILITIES EXTREAD,EXTWRITE
accessType 8
transient_lastDdlTime 1715172392

STORAGE INFORMATION

SerDe Library: org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe
InputFormat: org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat
OutputFormat: org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat
Compressed: No
Num Buckets: 0
Bucket Columns: []
Sort Columns: []

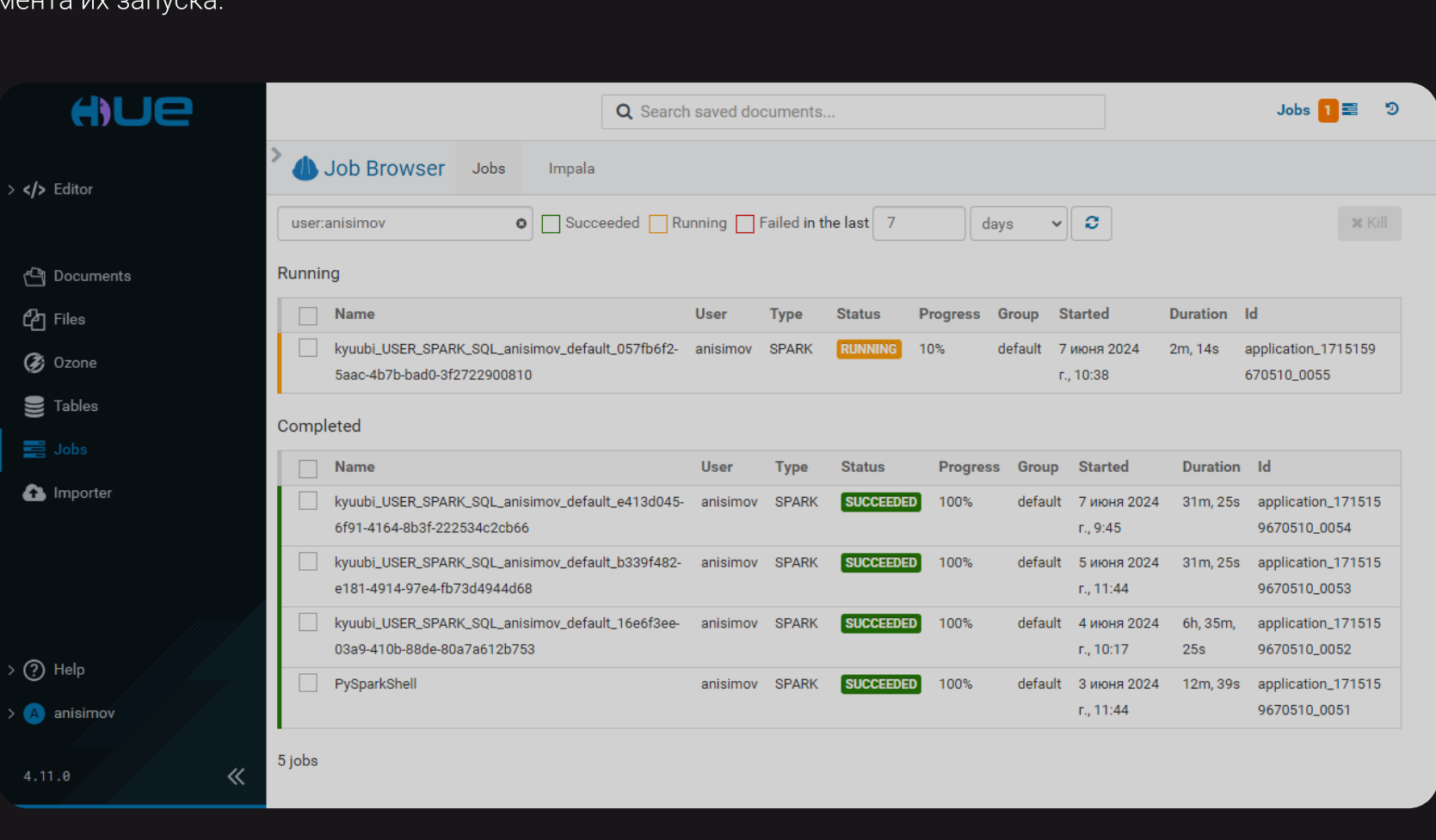
Вкладка Details для таблицы

Jobs

Страница ***Jobs*** отображает список всех выполняющихся и завершенных задач. Списки задач YARN и историю запросов Impala можно посмотреть во вкладках ***Jobs*** и ***Impala*** соответственно.

> **ПРИМЕЧАНИЕ**
>
> Эти вкладки появляются только если все компоненты соответствующих сервисов присутствуют в кластере ADH.

Список задач можно фильтровать при помощи опций ***Succeeded*** (завершено успешно), ***Running*** (выполняется) или ***Failed*** (завершено с ошибкой). Также доступен фильтр списка задач по времени с момента их запуска.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
anisimov
4.11.0
Search saved documents...
Jobs
Job Browser
Jobs
Impala
Kill

user:anisimov ☐ Succeeded ☐ Running ☐ Failed in the last 7 days

Running

Name	User	Type	Status	Progress	Group	Started	Duration	Id
kyuubi_USER_SPARK_SQL_anisimov_default_057fb6f2-5aac-4b7b-bad0-3f2722900810	anisimov	SPARK	RUNNING	10%	default	7 июня 2024 г., 10:38	2m, 14s	application_1715159670510_0055

Completed

Name	User	Type	Status	Progress	Group	Started	Duration	Id
kyuubi_USER_SPARK_SQL_anisimov_default_e413d045-6f91-4164-8b3f-222534c2cb66	anisimov	SPARK	SUCCEEDED	100%	default	7 июня 2024 г., 9:45	31m, 25s	application_1715159670510_0054
kyuubi_USER_SPARK_SQL_anisimov_default_b339f482-e181-4914-97e4-fb73d4944d68	anisimov	SPARK	SUCCEEDED	100%	default	5 июня 2024 г., 11:44	31m, 25s	application_1715159670510_0053
kyuubi_USER_SPARK_SQL_anisimov_default_16e6f3ee-03a9-410b-88de-80a7a612b753	anisimov	SPARK	SUCCEEDED	100%	default	4 июня 2024 г., 10:17	6h, 35m, 25s	application_1715159670510_0052
PySparkShell	anisimov	SPARK	SUCCEEDED	100%	default	3 июня 2024 г., 11:44	12m, 39s	application_1715159670510_0051

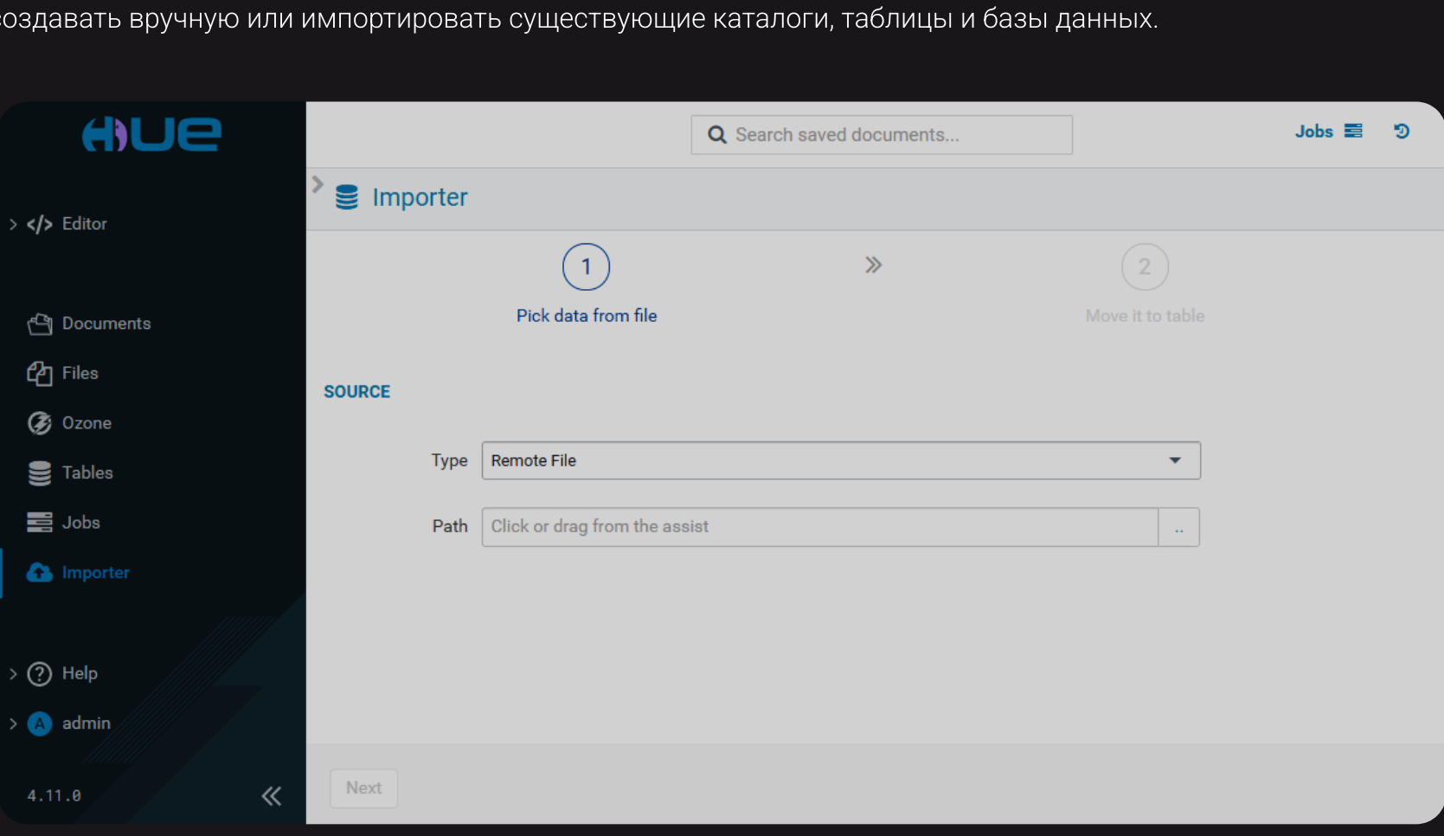
5 jobs

Страница Jobs

Чтобы приостановить, возобновить или остановить задачу, выделите её флажком и нажмите ***Suspend***, ***Resume*** или ***Kill*** соответственно. Эти действия можно выполнять над любым количеством задач.

Importer

Страница ***Importer*** позволяет создавать вручную или импортировать существующие каталоги, таблицы и базы данных.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
admin
4.11.0
Search saved documents...
Jobs
Importer
1
Pick data from file
2
Move it to table
SOURCE
Next

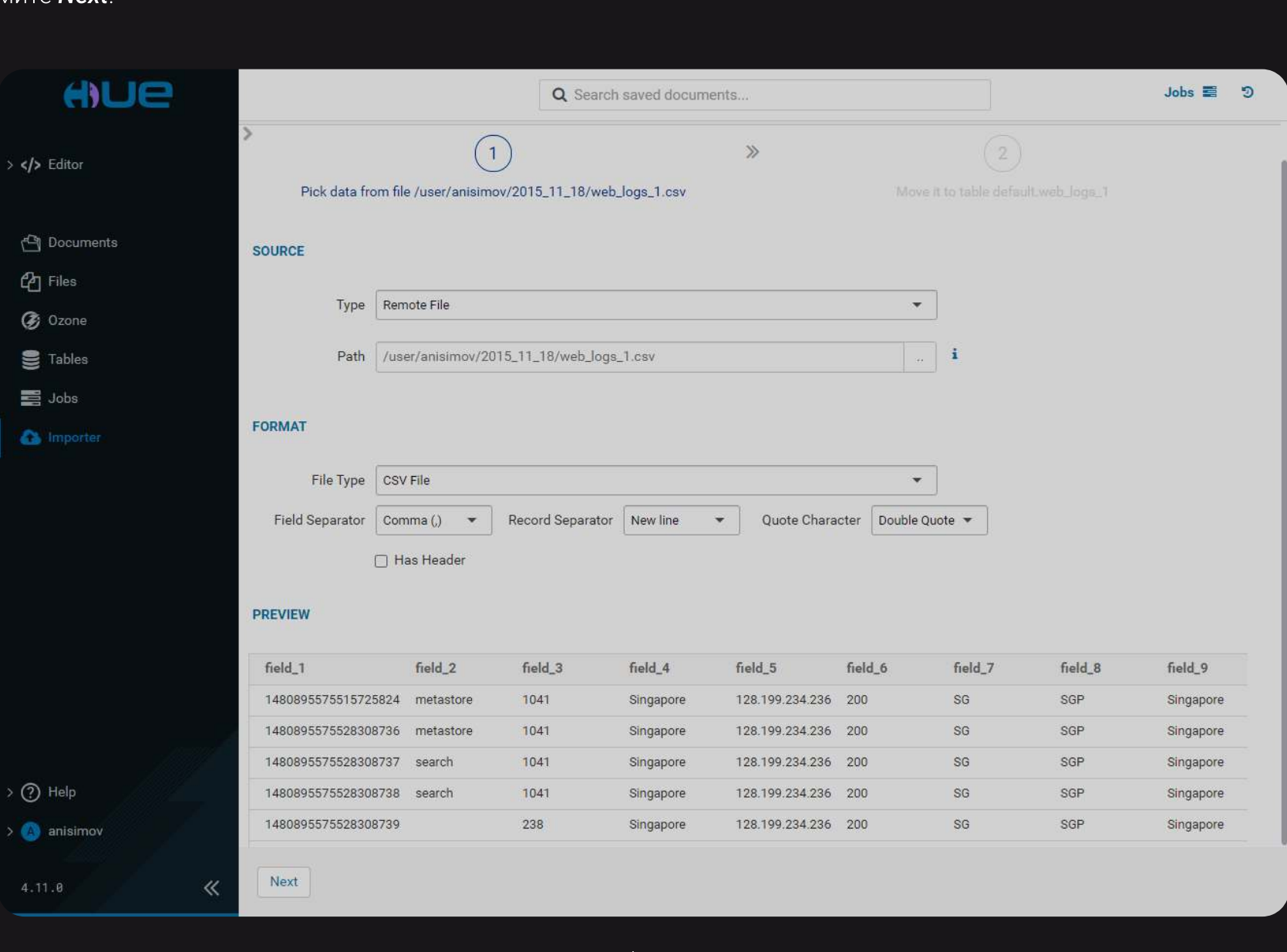
Type: Remote File

Path: Click or drag from the assist

Страница Importer

Импорт удаленного файла

Если вы хотите загрузить таблицу, хранящуюся в файле, выберите в поле ***Type*** значение `Remote File` и задайте путь к этому файлу. Поправьте значения в разделе ***Format***, если необходимо. Нажмите ***Next***.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
anisimov
4.11.0
Search saved documents...
Jobs
1
Pick data from file /user/anisimov/2015_11_18/web_logs_1.csv
2
Move it to table default.web_logs_1
SOURCE
FORMAT
PREVIEW
Next

Type: Remote File

Path: /user/anisimov/2015_11_18/web_logs_1.csv

File Type: CSV File

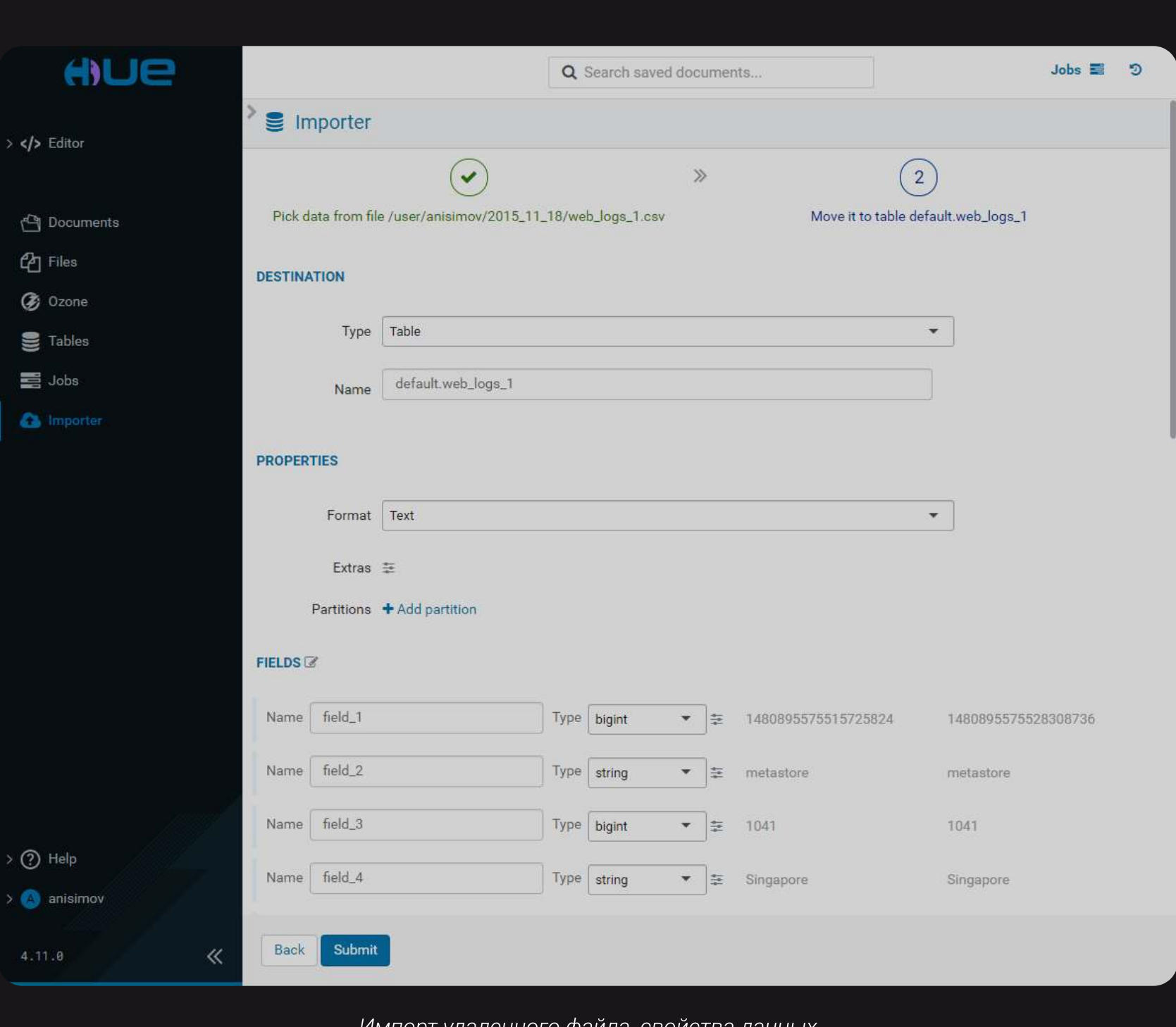
Field Separator: Comma (,) Record Separator: New line Quote Character: Double Quote

☐ Has Header

field_1	field_2	field_3	field_4	field_5	field_6	field_7	field_8	field_9
1480895575515725824	metastore	1041	Singapore	128.199.234.236	200	SG	SGP	Singapore
1480895575528308736	metastore	1041	Singapore	128.199.234.236	200	SG	SGP	Singapore
1480895575528308737	search	1041	Singapore	128.199.234.236	200	SG	SGP	Singapore
1480895575528308738	search	1041	Singapore	128.199.234.236	200	SG	SGP	Singapore
1480895575528308739		238	Singapore	128.199.234.236	200	SG	SGP	Singapore

Импорт удаленного файла, входные данные

На следующей странице задайте тип данных (таблица или индекс поиска), свойства данных и сведения о полях, и нажмите ***Submit***.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
anisimov
4.11.0
Search saved documents...
Jobs
Importer
Pick data from file /user/anisimov/2015_11_18/web_logs_1.csv
2
Move it to table default.web_logs_1
DESTINATION
PROPERTIES
Add partition
FIELDS
Back
Submit

Type: Table

Name: default.web_logs_1

Format: Text

Extras

Partitions

Name	Type		
field_1	bigint	1480895575515725824	1480895575528308736
field_2	string	metastore	metastore
field_3	bigint	1041	1041
field_4	string	Singapore	Singapore

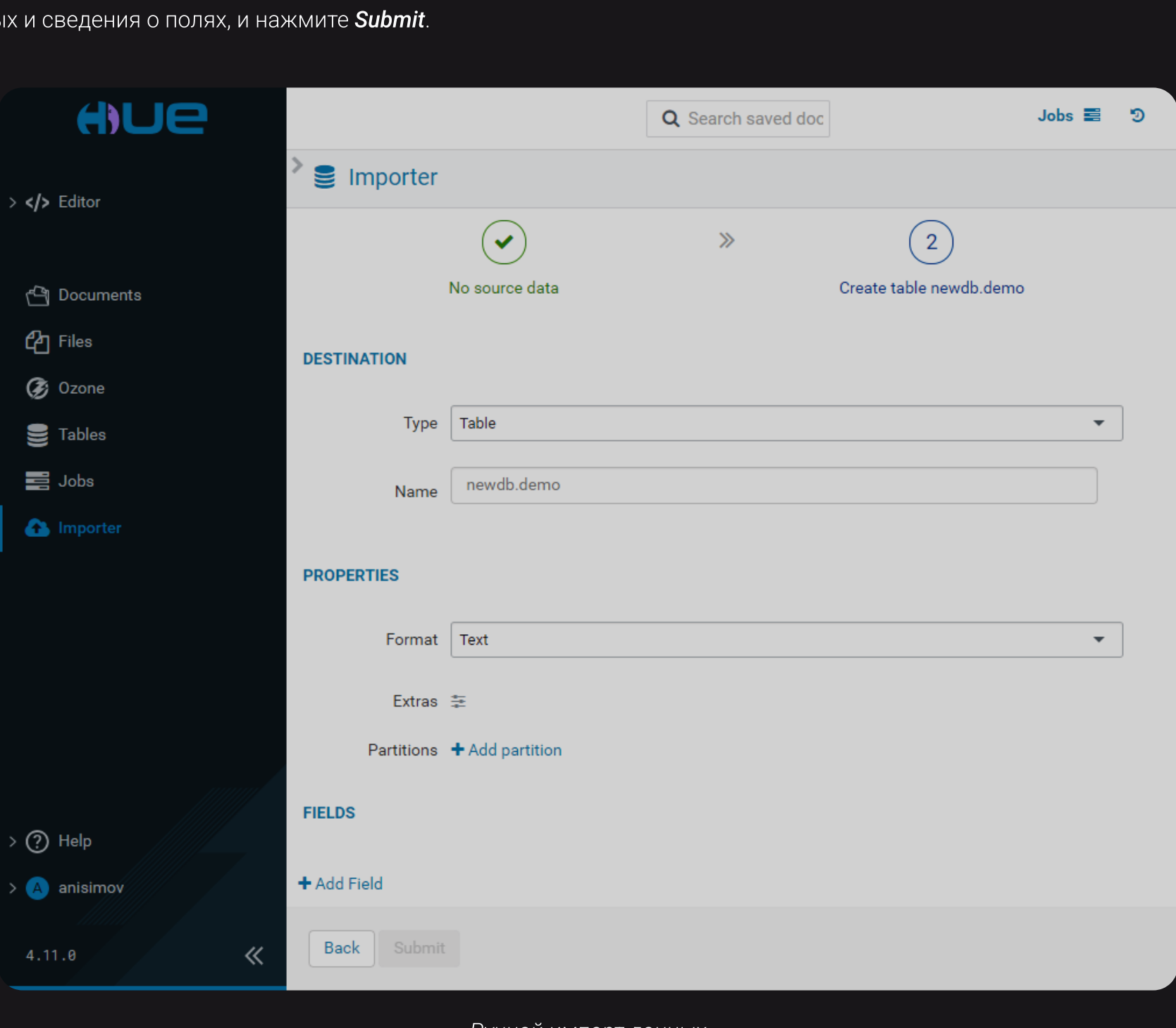
Импорт удаленного файла, свойства данных

Импорт внешней БД

Если вы хотите загрузить внешнюю БД, выберите в поле ***Type*** значение `External Databse` и задайте параметры подключения. Нажмите ***Next***. На следующей странице укажите параметры импорта и нажмите ***Submit***.

Ручной импорт данных

Если вы хотите загрузить данные вручную, выберите в поле ***Type*** значение `Manually` и нажмите ***Next***. На следующей странице задайте тип данных (`table`, `database` или `folder`), свойства данных и сведения о полях, и нажмите ***Submit***.



HUE
Editor
Documents
Files
Ozone
Tables
Jobs
Importer
Help
anisimov
4.11.0
Search saved doc
Jobs
Importer
No source data
2
Create table newdb.demo
DESTINATION
PROPERTIES
Add partition
FIELDS
Add Field
Back
Submit

Type: Table

Name: newdb.demo

Format: Text

Extras

Partitions

Ручной импорт данных

Добавление и использование интерпретатора в HUE

Владимир Адаменков

Содержание

Назад к содержанию

Используемые в HUE коннекторы

Соединение с ADB (Greenplum)

Используемые в HUE коннекторы

HUE в составе ADH имеет преднастроенные коннекторы для источников данных Hive, Impala и Kyuubi[Spark3]. С помощью секции ***Custom hue.ini***, доступной на конфигурационной странице сервиса HUE в интерфейсе ADCM, можно добавить множество других коннекторов — полный список приведен в документации HUE.

ПРИМЕЧАНИЕ

Преднастроенные коннекторы появляются в веб-интерфейсе HUE, если в кластере ADH присутствуют компоненты соответствующих сервисов:

- Hive — сервис Hive со всеми компонентами;
- Impala — сервис Impala со всеми компонентами;
- PySpark — компонент Spark3 Livy Server на том же хосте, что и HUE Server;
- Trino — сервис Trino со всеми компонентами, также требует SSL при использовании Kerberos-аутентификации;
- Kyuubi[Spark3] — компоненты Kyuubi Server и Spark3 Client на том же хосте, что и HUE Server;
- Flink — компонент Flink2 SQL-Gateway на любом хосте.

После добавления необходимых сервисов и/или компонентов нужно перезапустить сервис HUE, чтобы изменения вступили в силу.

ВАЖНО

Если Kerberos-аутентификация включена в кластере, интерпретатор Trino требует использования SSL.

Чтобы настроить коннектор, войдите в веб-интерфейс ADCM, выберите кластер ADH и перейдите в раздел ***Services → HUE → Components → HUE Server***. Выберите требуемую секцию интерпретатора и откорректируйте параметры по необходимости. Описание параметров приведено на странице Конфигурационные параметры.

Соединение с ADB (Greenplum)

Ниже приведен пример, как добавить и настроить коннектор для БД ADB, которая основана на Greenplum.

Чтобы соединить HUE с экземпляром БД ADB, убедитесь, что она ожидает соединения на определенном сетевом интерфейсе от определенной учетной записи. Войдите в веб-интерфейс ADCM и выберите ваш кластер ADB, затем перейдите в раздел ***Services → ADB → Primary configuration***. Откройте параметр ***Custom pg_hba section*** и добавьте в него строку следующего вида:

```
host <dbname> <user> <IP-address>/<mask> trust
```

где:

- `<dbname>` — имя БД, к которой будет выполняться подключение.
- `<user>` — имя пользователя БД.
- `<IP-address>/<mask>` — IP-адрес и маска сети хоста ADH, на котором установлен HUE Server.

Пример:

```
host adb admin 10.10.10.21/0 trust
```

Сохраните конфигурацию, затем нажмите ***Actions*** и выберите ***Reconfigure***.

После этого нужно добавить параметры соединения в конфигурацию HUE. Выберите ваш кластер ADH в веб-интерфейсе ADCM и перейдите в раздел ***Services → HUE → Components → HUE Server***. В появившейся вкладке ***Primary configuration*** нажмите ***Show advanced***. Откройте секцию ***Custom hue.ini*** и добавьте три параметра, нажимая ***Add property***:

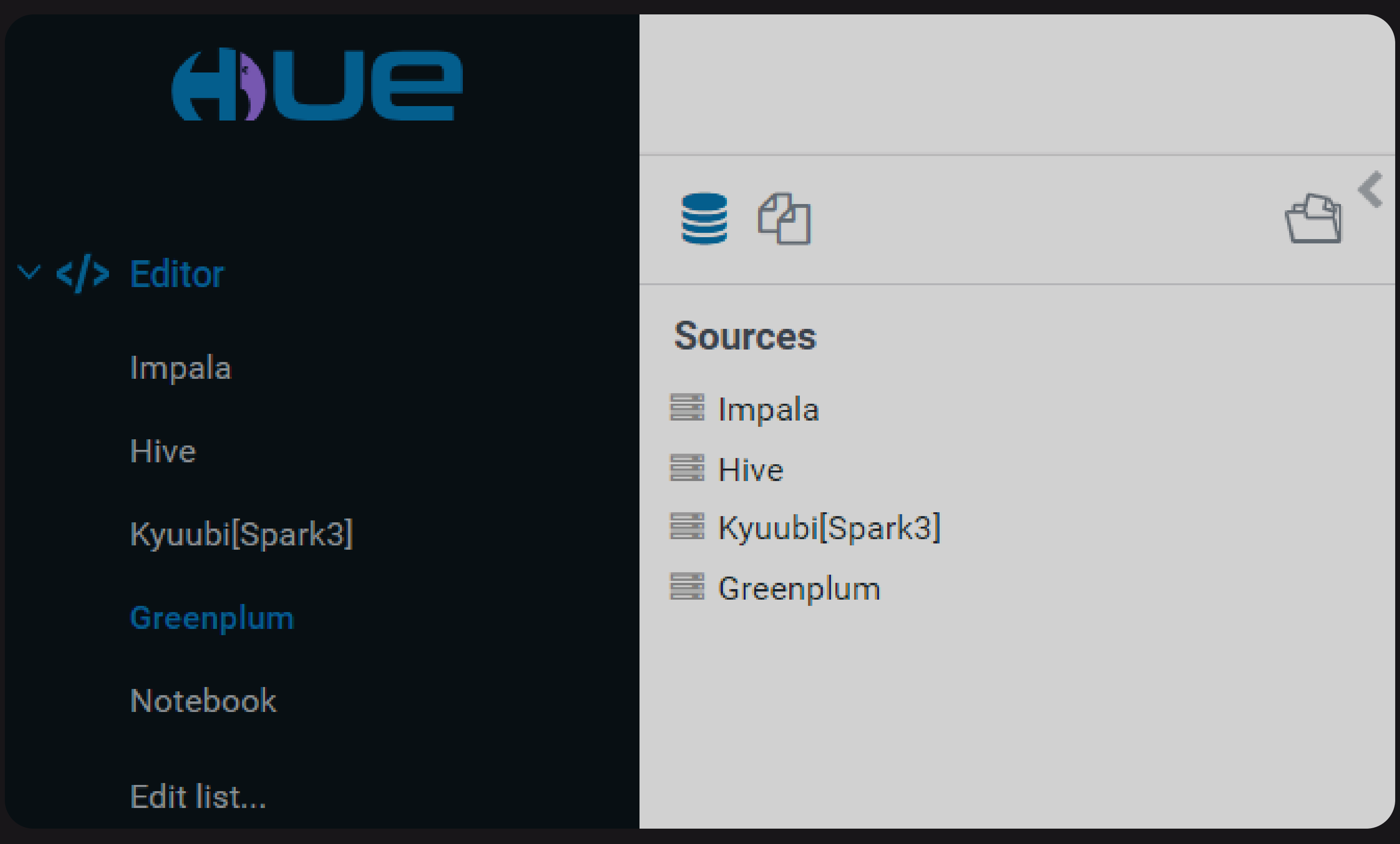
- `notebook.interpreters.greenplum.name` = `Greenplum`
- `notebook.interpreters.greenplum.interface` = `sqlalchemy`
- `notebook.interpreters.greenplum.options` = `'{"url": "postgresql+psycopg2://<user>:<password>@<dbaddress>:<port>/<dbname>"}'`

где:

- `<password>` — пароль пользователя БД.
- `<dbaddress>` — IP-адрес хоста, на котором БД ожидает соединения (ADB master).
- `<port>` — порт, на котором БД ожидает соединения.

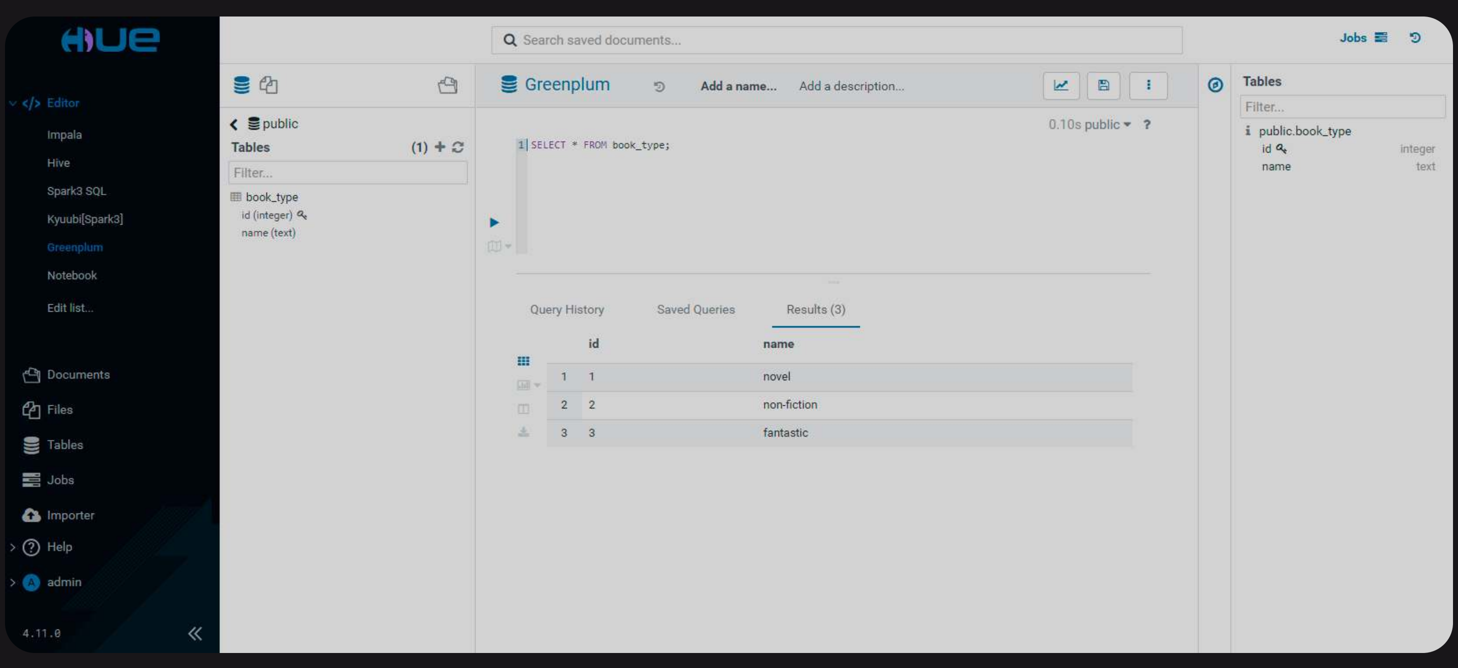
Стандартный порт Greenplum в ADB — 5432. Пример для параметра `options`: `'{"url": "postgresql+psycopg2://admin:Secure_pass35@10.10.10.10:5432/adb"}'`.

Сохраните конфигурацию, затем в меню ***Actions*** выберите ***Restart HUE Server***. Дождитесь окончания перезапуска и перейдите в веб-интерфейс HUE. Источник данных ***Greenplum*** должен появиться в меню быстрого просмотра и в списке ***Editor***.



Источник данных Greenplum

Можно выполнять запросы для выбранной таблицы. Введите запрос в рабочей области и нажмите ***Execute*** (или `Ctrl+Enter`).

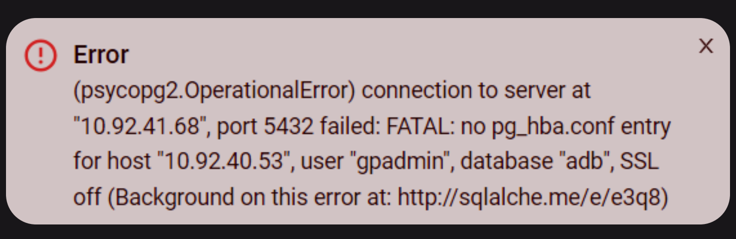


```
SELECT * FROM book_type;
```

	id	name
1	1	novel
2	2	non-fiction
3	3	fantastic

Пример запроса

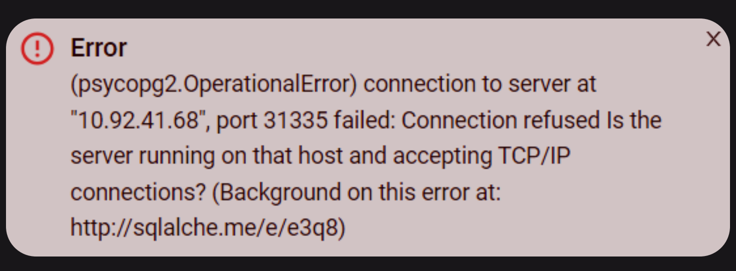
Если параметр ADB ***Custom pg_hba section*** был задан неправильно или в конфигурации HUE Server были заданы неверные учетные данные, соединение не будет установлено и появится сообщение об ошибке.



```
Error
(psycopg2.OperationalError) connection to server at
"10.92.41.68", port 5432 failed: FATAL: no pg_hba.conf entry
for host "10.92.40.53", user "gpadmin", database "adb", SSL
off (Background on this error at: http://sqlalche.me/e/e3q8)
```

Ошибка настройки pg_hba.conf

Если в конфигурации HUE Server был задан неверный порт, соединение не будет установлено и появится сообщение об ошибке.



```
Error
(psycopg2.OperationalError) connection to server at
"10.92.41.68", port 31335 failed: Connection refused Is the
server running on that host and accepting TCP/IP
connections? (Background on this error at:
http://sqlalche.me/e/e3q8)
```

Ошибка отказа в соединении

Работа с таблицами Iceberg в HUE

Владимир Адаменков

Содержание

Назад к содержанию

Обзор

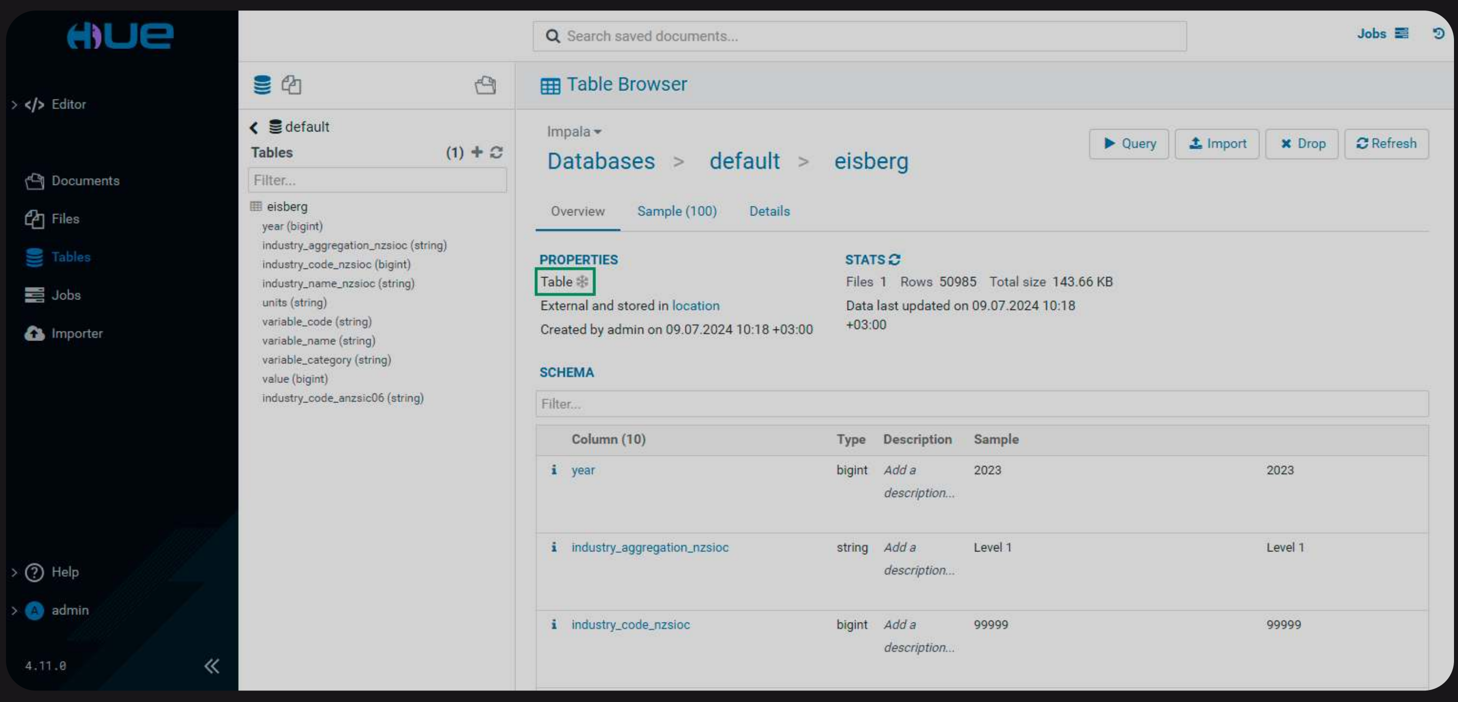
Добавление таблицы Iceberg

Обзор

Apache Iceberg — это открытый высокопроизводительный формат для больших аналитических таблиц. HUE умеет распознавать таблицы Iceberg и конвертировать таблицы из удаленных файлов в формат Iceberg.

> **ПРИМЕЧАНИЕ**
>
> Hive поддерживает Iceberg-таблицы начиная с версии ADH 3.2.4.3 и выше.

Таблицы Iceberg отмечены иконкой ❄ во вкладке ***Overview*** под заголовком ***Properties*** в браузере таблиц HUE.


HUE
Search saved documents...
Jobs
Editor
Documents
Files
Tables
Jobs
Importer
Help
admin
4.11.0
Table Browser
Query
Import
Drop
Refresh
Overview
Sample (100)
Details
Filter...

default

Tables (1)

eisberg
- year (bigint)
- industry_aggregation_nzsioc (string)
- industry_code_nzsioc (bigint)
- industry_name_nzsioc (string)
- units (string)
- variable_code (string)
- variable_name (string)
- variable_category (string)
- value (bigint)
- industry_code_anzsic06 (string)

Impala

Databases > default > eisberg

PROPERTIES

Table ❄

External and stored in location

Created by admin on 09.07.2024 10:18 +03:00

STATS

Files 1 Rows 50985 Total size 143.66 KB

Data last updated on 09.07.2024 10:18 +03:00

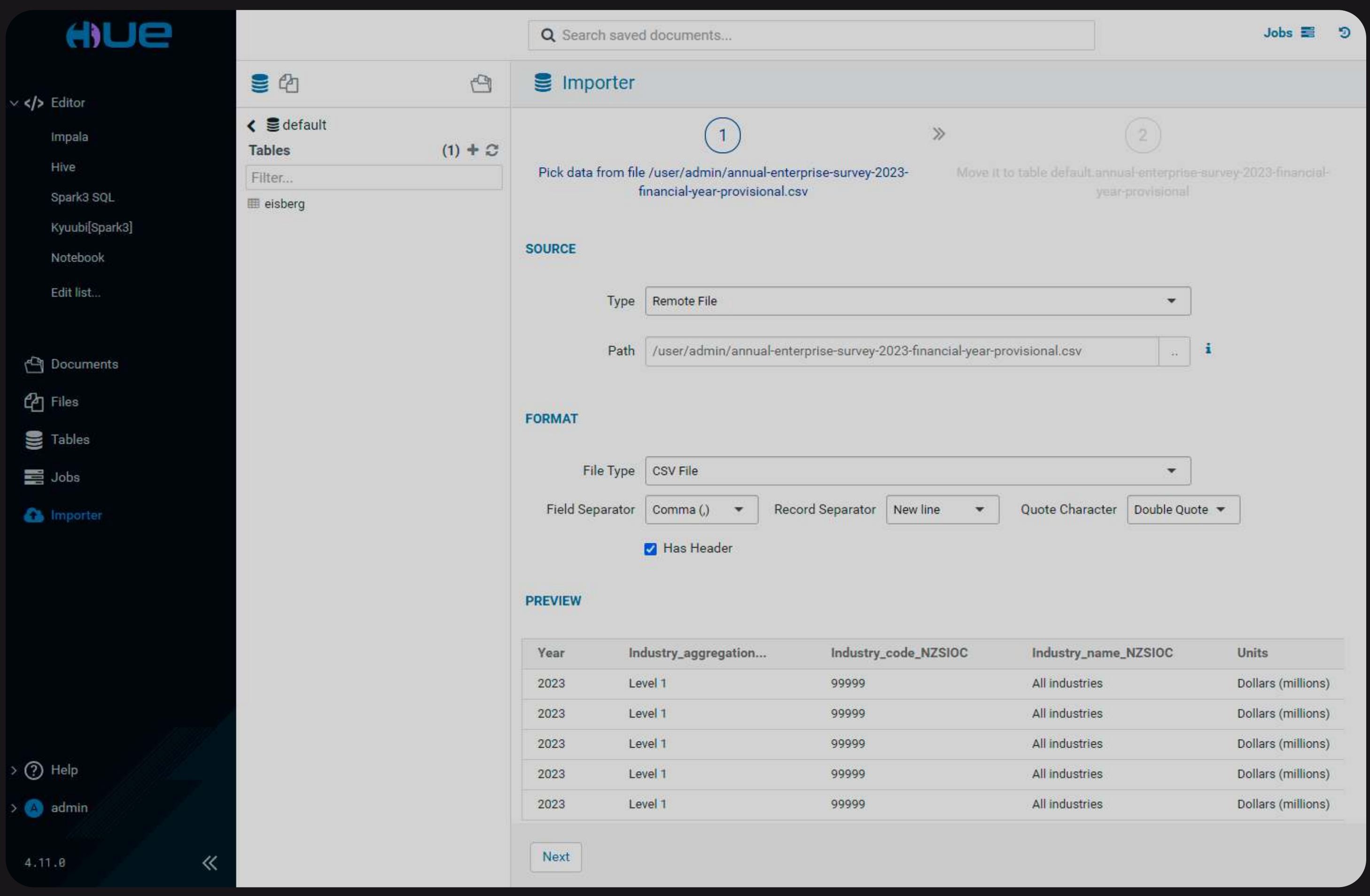
SCHEMA

	Column (10)	Type	Description	Sample	
i	year	bigint	Add a description...	2023	2023
i	industry_aggregation_nzsioc	string	Add a description...	Level 1	Level 1
i	industry_code_nzsioc	bigint	Add a description...	99999	99999

Пометка таблицы Iceberg

Добавление таблицы Iceberg

Таблицы Iceberg можно добавлять с помощью импорта таблиц из удаленных файлов с последующей конвертацией. Это можно сделать на странице ***Importer*** или нажав на иконку ➕ в меню быстрого просмотра (если выбрана база данных). Выберите тип `Remote File` и задайте пусть к файлу таблицы. Поправьте значения под заголовком ***Format***, если необходимо, и нажмите ***Next***.


HUE
Search saved documents...
Jobs
Editor
Impala
Hive
Spark3 SQL
Kyuubi[Spark3]
Notebook
Edit list...
Documents
Files
Tables
Jobs
Importer
Help
admin
4.11.0
default
Tables (1)
Filter...
eisberg
Importer
1
2
Next

Pick data from file /user/admin/annual-enterprise-survey-2023-financial-year-provisional.csv

Move it to table default.annual-enterprise-survey-2023-financial-year-provisional

SOURCE

Type: Remote File

Path: /user/admin/annual-enterprise-survey-2023-financial-year-provisional.csv

FORMAT

File Type: CSV File

Field Separator: Comma (,) Record Separator: New line Quote Character: Double Quote

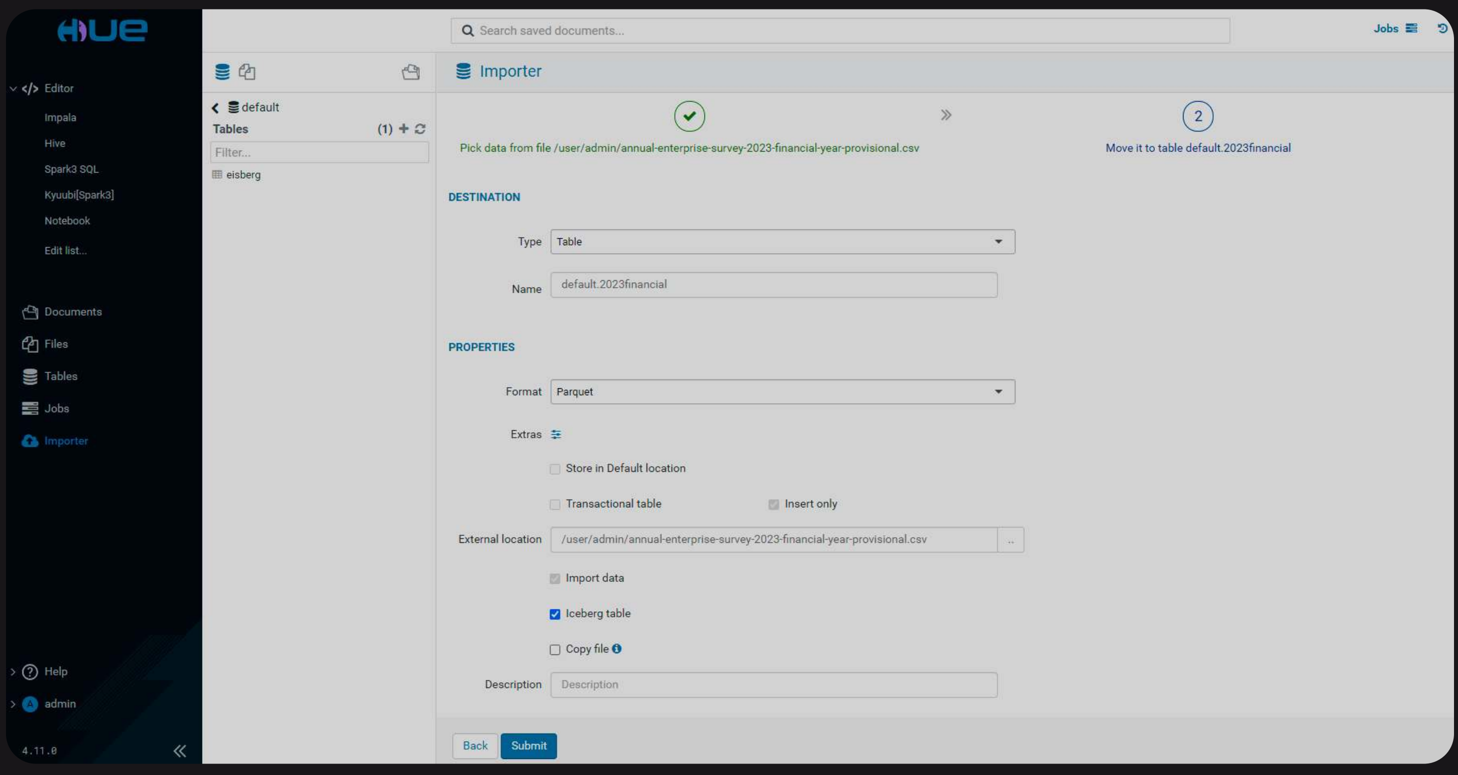
☑ Has Header

PREVIEW

Year	Industry_aggregation...	Industry_code_NZSIOC	Industry_name_NZSIOC	Units
2023	Level 1	99999	All industries	Dollars (millions)
2023	Level 1	99999	All industries	Dollars (millions)
2023	Level 1	99999	All industries	Dollars (millions)
2023	Level 1	99999	All industries	Dollars (millions)
2023	Level 1	99999	All industries	Dollars (millions)

Импорт таблицы из удаленного файла

На следующем экране нажмите ***Extras*** и установите флажок ***Iceberg table***. Проверьте и поправьте другие параметры, имена и типы полей, если необходимо. Нажмите ***Submit***.


HUE
Search saved documents...
Jobs
Editor
Impala
Hive
Spark3 SQL
Kyuubi[Spark3]
Notebook
Edit list...
Documents
Files
Tables
Jobs
Importer
Help
admin
4.11.0
default
Tables (1)
Filter...
eisberg
Importer
2
Back
Submit

Pick data from file /user/admin/annual-enterprise-survey-2023-financial-year-provisional.csv

Move it to table default.2023financial

DESTINATION

Type: Table

Name: default.2023financial

PROPERTIES

Format: Parquet

Extras

☐ Store in Default location

☐ Transactional table ☑ Insert only

External location: /user/admin/annual-enterprise-survey-2023-financial-year-provisional.csv

☑ Import data

☑ Iceberg table

☐ Copy file

Description: Description

Подробности конвертации в формат Iceberg

После успешного импорта новая таблица Iceberg появится в меню быстрого просмотра под именем соответствующей базы данных.

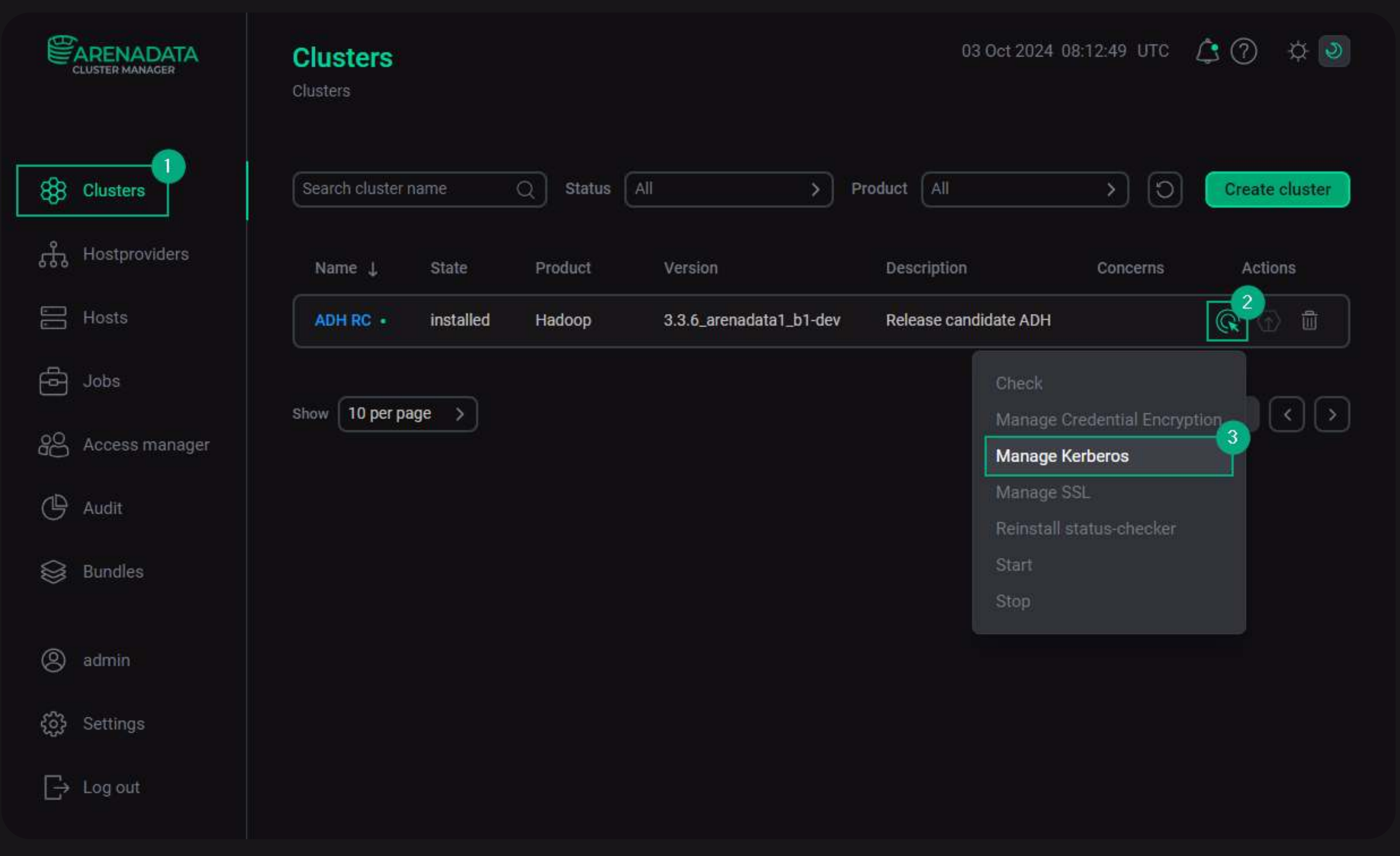
Аутентификация SPNEGO в HUE

Владимир Адаменков

Для веб-интерфейса HUE можно настроить SPNEGO-аутентификацию, что предоставляет удаленный защищенный доступ к нему. Этот тип аутентификации совместим с LDAP-аутентификацией.

Для настройки SPNEGO-аутентификации выполните следующие шаги:

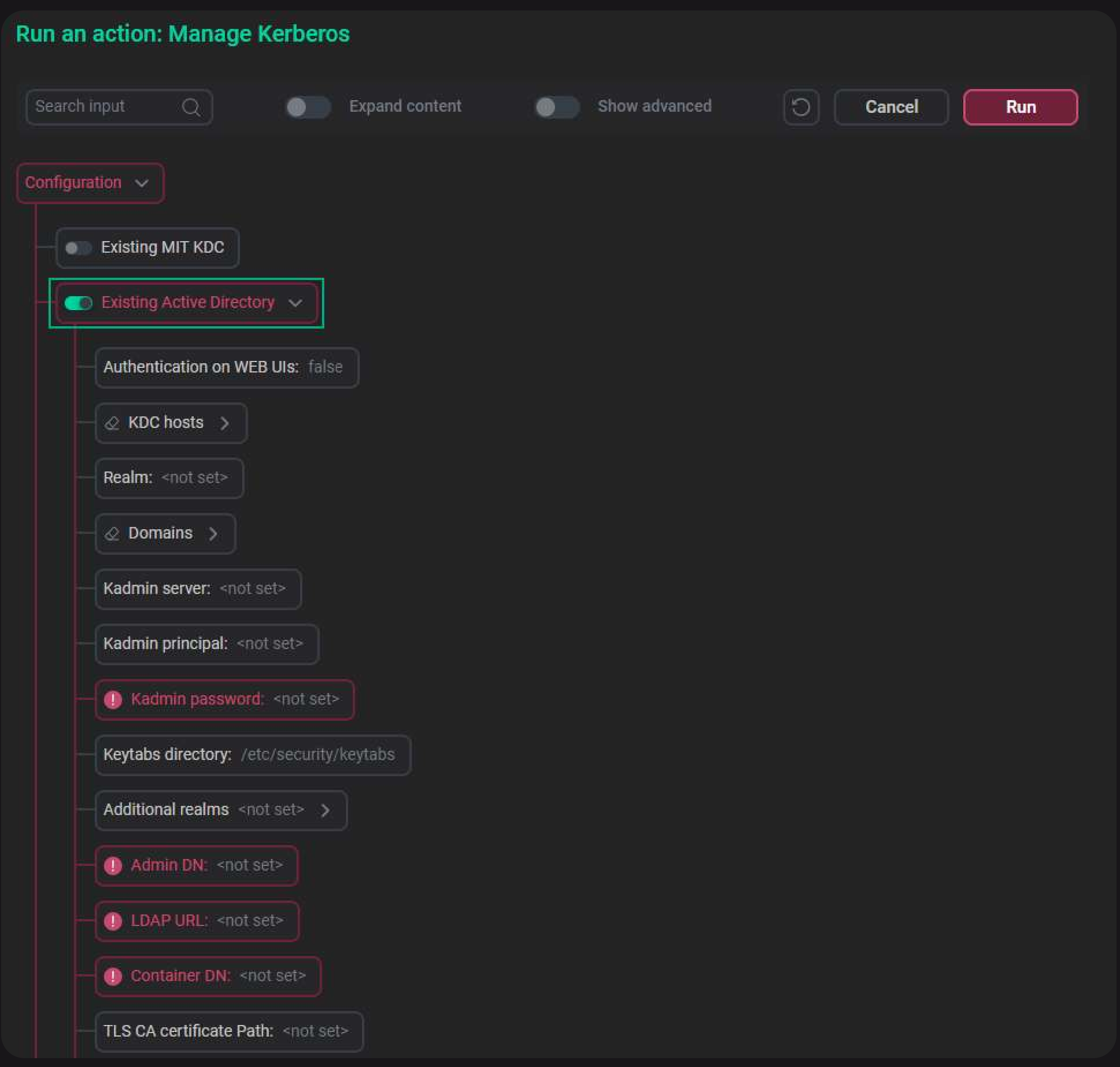
1. Откройте веб-интерфейс ADCM и выберите страницу ***Clusters***.
2. Нажмите на значок ***Action*** для нужного кластера и выберите ***Manage Kerberos***.


ARENADATA CLUSTER MANAGER
Clusters
Clusters
03 Oct 2024 08:12:49 UTC
Clusters
1
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Name
State
Product
Version
Description
Concerns
Actions
2
Show
10 per page
Check
Manage Credential Encryption
Manage Kerberos
3
Manage SSL
Reinstall status-checker
Start
Stop

Name	State	Product	Version	Description	Concerns	Actions
ADH RC	installed	Hadoop	3.3.6_arenadata1_b1-dev	Release candidate ADH		

Действие Manage Kerberos

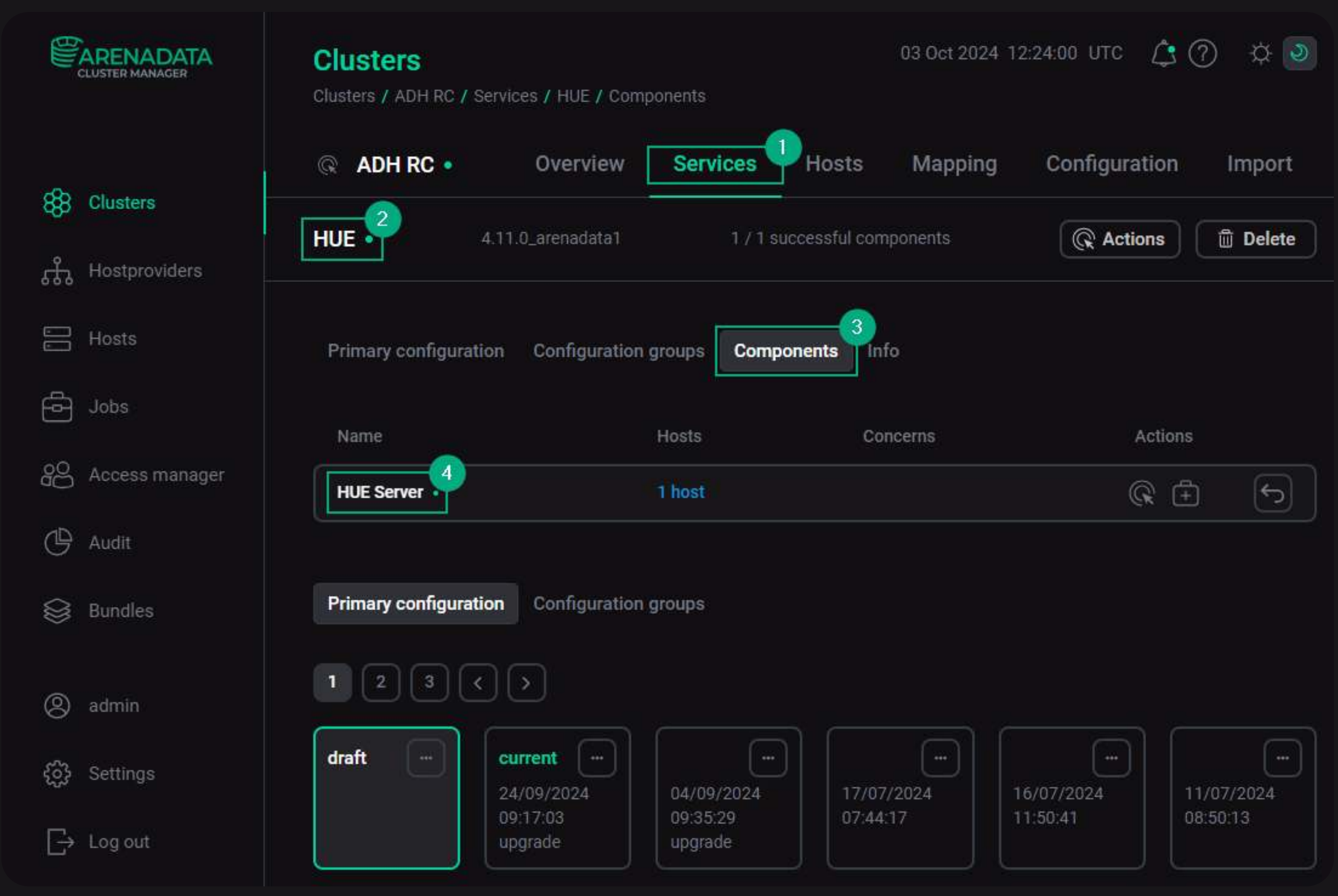
3. В появившемся окне активируйте переключатель ***Existing Active Directory***.


Run an action: Manage Kerberos
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
KDC hosts
Domains

Authentication on WEB UIs: false
Realm: <not set>
Kadmin server: <not set>
Kadmin principal: <not set>
Kadmin password: <not set>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
Admin DN: <not set>
LDAP URL: <not set>
Container DN: <not set>
TLS CA certificate Path: <not set>

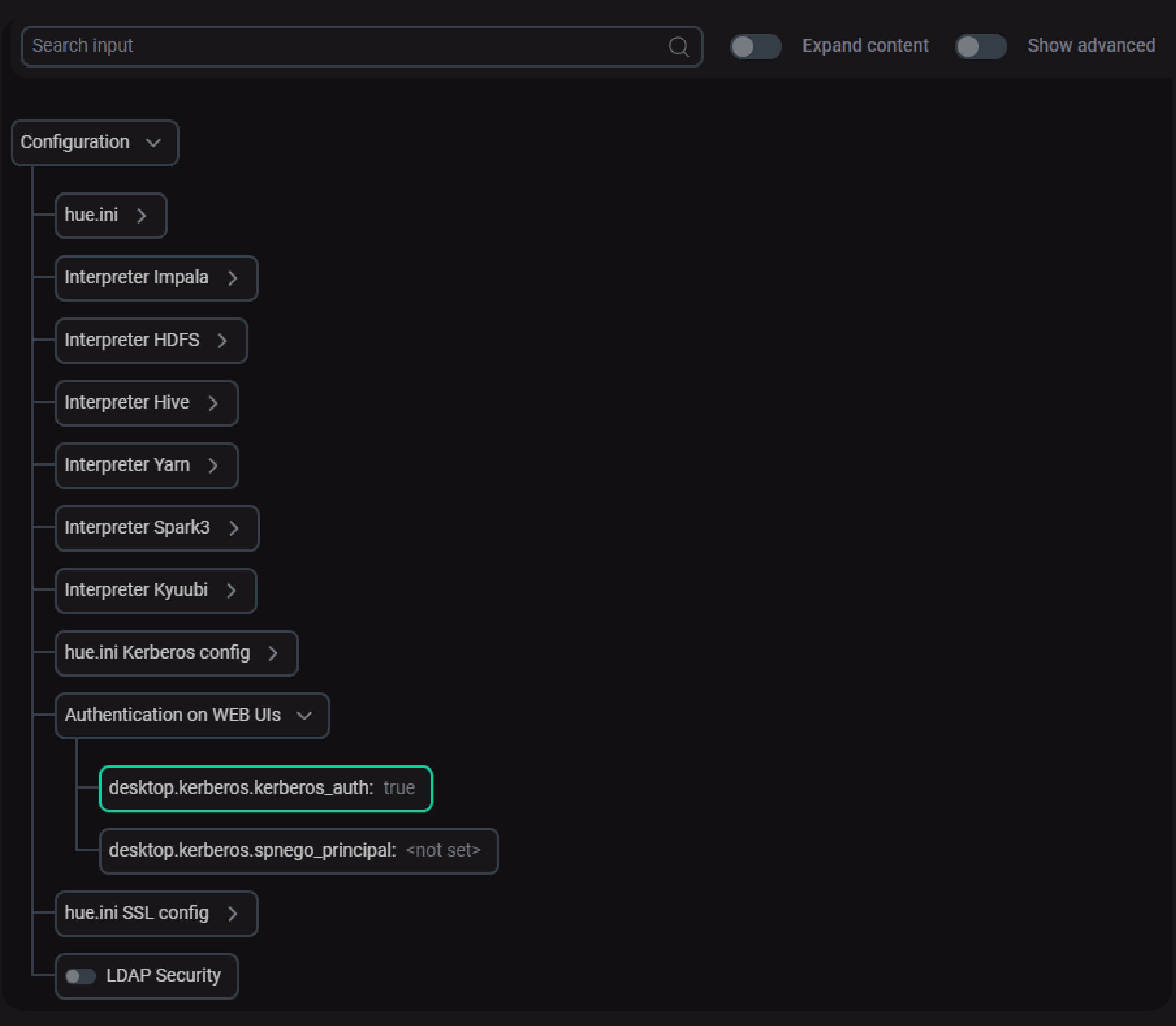
Группа параметров настройки Existing Active Directory

4. Выберите параметр ***Authentication on WEB UIs*** и укажите для него значение `true`.
5. Укажите значения для остальных параметров в этой группе согласно статье Настройка использования Active Directory для Kerberos-аутентификации с помощью ADCM.
6. Нажмите ***Run*** и дождитесь окончания действия.
7. Выберите нужный кластер, откройте страницу сервиса ***HUE*** и выберите компонент ***HUE Server*** во вкладке ***Components***.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH RC / Services / HUE / Components
03 Oct 2024 12:24:00 UTC
ADH RC
Overview
Services
1
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
HUE
2
4.11.0_arenadata1
1 / 1 successful components
Actions
Delete
Primary configuration
Configuration groups
Components
3
Info
Name
Hosts
Concerns
Actions
HUE Server
4
1 host
Primary configuration
Configuration groups
1
2
3
draft
current
24/09/2024
09:17:03
upgrade
04/09/2024
09:35:29
upgrade
17/07/2024
07:44:17
16/07/2024
11:50:41
11/07/2024
08:50:13

Компонент HUE Server

8. Во вкладке ***Primary configuration*** раскройте группу параметров ***Authentication on WEB UIs*** и укажите для параметра ***desktop.kerberos.kerberos_auth*** значение `true`.


Search input
Expand content
Show advanced
Configuration
hue.ini
Interpreter Impala
Interpreter HDFS
Interpreter Hive
Interpreter Yarn
Interpreter Spark3
Interpreter Kyuubi
hue.ini Kerberos config
Authentication on WEB UIs
hue.ini SSL config
LDAP Security

desktop.kerberos.kerberos_auth: true
desktop.kerberos.spnego_principal: <not set>

Главная конфигурация HUE Server

9. При необходимости укажите значение параметра ***desktop.kerberos.spnego_principal***. Он задает принципал Kerberos по умолчанию. Если этот параметр оставить пустым, то принципал будет извлекаться из тикета.
10. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, нажав ***Actions*** → ***Restart HUE Server***.

Перед использованием SPNEGO-аутентификации для веб-интерфейса HUE убедитесь, что на вашей локальной машине установлена и настроена служба Kerberos. Также ваш браузер должен быть настроен для использования Kerberos — обратитесь к статье Аутентификация SPNEGO за инструкциями по настройке браузеров Mozilla Firefox и Google Chrome. Перед установкой соединения следует запустить команду `kinit`.

Управление сервисом HUE через ADCM

Владимир Адаменков

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом HUE. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса HUE, перечислены в таблице ниже.

Действия с сервисом HUE

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты HUE
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются

Действия, доступные для компонентов сервиса HUE, перечислены в таблице ниже.

Действия с компонентами HUE

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Использование impala-shell для подключения к Impala

Евгения Кузина

Содержание

Назад к содержанию

- Запуск impala-shell
- Параметры конфигурации
- Конфигурационный файл impala-shell
- Подключение к impalad на определенном хосте
  - С помощью параметров командной строки
  - С помощью конфигурационного файла
  - С помощью команды CONNECT
- Подключение к определенной базе данных
- Выполнение операторов SQL в неинтерактивном режиме
- Выполнение команд и операторов SQL
- Подстановка переменных

Вы можете подключиться к Impala с помощью инструмента командной строки — `impala-shell`. Эта утилита доступна на хостах, на которых установлен компонент Impala Client. Оболочка Impala может удаленно подключаться к любому экземпляру Impala Daemon.

Утилита impala-shell позволяет создавать базы данных и таблицы, управлять ими, добавлять данные и выполнять запросы. Вы также можете выполнять операторы SQL в интерактивном режиме. Можно указывать параметры командной строки для обработки одного оператора или script-файла, содержащего большое количество операторов.

Кроме того, вы можете сохранить набор настроек по умолчанию для impala-shell в файле конфигурации.

Запуск impala-shell

Чтобы запустить impala-shell, используйте следующую команду:

```
$ impala-shell
```

Для выполнения запросов необходимо подключиться к демону impalad. Команда `impala-shell` подключается к impalad, работающему на том же хосте. Для получения информации о том, как подключиться к демону impalad на другом хосте, см. Подключение к impalad на определенном хосте.

ПРИМЕЧАНИЕ

Если при использовании impala-shell возникла ошибка `ImportError: no module named pkg_resources`, необходимо установить пакет `python-setuptools`. Для этого можно воспользоваться командой: `yum install python-setuptools`.

Параметры конфигурации

Для того чтобы настроить impala-shell под определенные задачи, используются параметры конфигурации. В таблице ниже перечислены параметры, которые могут быть переданы в качестве параметра командной строки или определены в конфигурационном файле.

Параметры конфигурации impala-shell

Параметр командной строки	Настройка в конфигурационном файле	Описание
-B или --delimited	write_delimited=true	Выводит результаты запроса в виде текстового файла с разделителями. Применяется для создания файлов данных, которые могут быть использованы с другими компонентами Hadoop. Эта опция также помогает избежать снижения производительности из-за форматированного вывода при выполнении тестов производительности, в которых используются запросы, возвращающие большие наборы результатов. Чтобы указать символ-разделитель, используйте параметр `--output_delimiter`. Чтобы определить имя файла, используйте опцию `-o` или `--output_file`
-b или --kerberos_host_fqdn	kerberos_host_fqdn=load-balancer-hostname	Переопределяет имя хоста принципала Kerberos демона Impala. impala-shell проверяет, соответствует ли принципал сервера этому имени хоста. Его можно использовать, когда доступ к impalad осуществляется через балансировщик нагрузки, но impala-shell должен напрямую подключаться к определённому демону impalad
--print_header	print_header=true	Определяет, следует ли печатать строку заголовка
-o filename или --output_file filename	output_file=filename	Указывает имя выходного файла. Обычно он используется для хранения результатов единичного запроса, выполненного из командной строки с параметром `-q`. Опция также работает для интерактивных сессий — отображаются сообщения, такие как количество извлеченных строк, но не фактический набор результатов. Чтобы эти сообщения не отображались (при сочетании параметров `-q` и `-o`), перенаправьте stderr в `/dev/null`
--output_delimiter=character	output_delimiter=character	Указывает символ, используемый в качестве разделителя между полями, когда результаты запроса печатаются в формате текста (параметр `-B`). Разделителем по умолчанию является табуляция (`\t`). Если выходное значение содержит символ-разделитель, такое поле может быть заключено в одинарные кавычки, или экранировано двойными кавычками, или и тем, и другим
-E или --vertical	vertical=true	Выводит итоговые данные запроса (строки) по вертикали. Эта опция не действует, если используется опция `-B`
-p или --show_profiles	show_profiles=true	Отображает план выполнения запроса (тот же результат, что и оператор `EXPLAIN`), а также более подробные этапы выполнения запроса
-h или --help	—	Отображает справочную информацию
—	history_max=1000	Устанавливает максимальное количество запросов для хранения в файле истории выполненных запросов в impala-shell
-i hostname или --impalad=hostname[:port_num]	impalad=hostname[:port_num]	Подключается к демону impalad, установленному на указанном хосте. Если параметр `port_number` не определен, используется значение по умолчанию `21050`. Эта опция позволяет подключиться к любому хосту в кластере. Если вы хотите получить доступ к экземпляру impalad, который запущен на другом порте (определенном параметром `fe_port`) добавьте этот порт в параметры опции
-q query или --query=query	query=query	Позволяет выполнить один запрос из командной строки, не запуская интерактивный интерпретатор. Вы можете использовать эту опцию для запуска impala-shell из script-файла оболочки или при вызове команды из Python, Perl или других скриптов. Запрос должен содержать один оператор, например, `SELECT`, `CREATE TABLE`, `SHOW TABLES` или любой другой оператор, распознаваемый impala-shell. Поскольку вы не можете передать оператор `USE` и другой запрос, используйте полные имена, включающие имя базы данных, для любой таблицы, которая не принадлежит базе данных по умолчанию. В качестве альтернативного решения используйте параметр `-f` для передачи файла с оператором `USE`, за которым могут следовать другие запросы
-f query_file или --query_file=query_file	query_file=path_to_query_file	Позволяет обрабатывать файл, содержащий несколько операторов SQL. Например, набор операторов DDL для создания группы таблиц и представлений. Несколько операторов должны быть разделены точкой с запятой (`;`). Вы можете указать имя файла как `-` для использования стандартного ввода. Эта функция позволяет использовать impala-shell как часть пайплайна Unix, где операторы SQL генерируются динамически другими инструментами
--query_option="option=value" или -Q "option=value"	Строка заголовка — `[impala.query_options]`, за которой следуют строки `option=value`, по одному параметру в строке	Устанавливает параметры запроса по умолчанию для команд impala-shell. Чтобы задать сразу несколько параметров запроса, используйте несколько экземпляров этой опции командной строки. Имена параметров запроса не чувствительны к регистру
-k или --kerberos	use_kerberos=true	Определяет использовать ли аутентификацию Kerberos при подключении к impalad. Если Kerberos не включен на экземпляре impalad, к которому подключается impala-shell, отображается ошибка. См. Enabling Kerberos Authentication for Impala
-s kerberos_service_name или --kerberos_service_name=name	kerberos_service_name=name	Определяет имя impala-shell для аутентификации в определённом принципале сервиса impalad. Если `kerberos_service_name` не указано, по умолчанию используется `impala`. Если этот параметр задаётся для соединения, в котором не поддерживается Kerberos, возвращается ошибка
-V или --verbose	verbose=true	Включает подробный вывод
--quiet	verbose=false	Отключает подробный вывод
-v или --version	version=true	Выводит информацию о версии impala-shell
-c	ignore_query_failure=true	Определяет игнорировать ли ситуацию, когда запрос не смог выполниться
-d default_db или --database=default_db	default_db=default_db	Указывает начальную базу данных. Эквивалентен выполнению оператора USE после подключения. Если параметр не указан, используется база данных `DEFAULT`
-ssl	ssl=true	Включает TLS/SSL для impala-shell
--ca_cert=path_to_certificate	ca_cert=path_to_certificate	Локальный путь к стороннему сертификату CA или к копии сертификата сервера для самоподписанных сертификатов сервера. Если `--ca_cert` не установлен, impala-shell включает TLS/SSL, но не проверяет сертификат сервера. Это полезно для подключения к доверенной Impala, которая работает только через TLS/SSL, когда копия сертификата недоступна, например, при отладке клиентских систем
-l	use_ldap=true	Включает LDAP-аутентификацию
-u	user=user_name	Указывает имя пользователя, когда аутентификация LDAP включена. Используйте короткое имя пользователя, а не полное отличительное имя LDAP. После использования этой опции оболочка запросит пароль
--ldap_password_cmd=command	ldap_password_cmd	Указывает команду для получения пароля LDAP, когда аутентификация LDAP включена. Если команда включает аргументы, разделенные пробелами, заключите команду и ее аргументы в кавычки. Например, `ldap_password_cmd=echo -n myPassword`, если вы храните пароль в *impalarc*, или `ldap_password_cmd=cat /etc/impala/conf/user-pass.pwd`, если вы храните пароль в файле
--config_file=path_to_config_file	—	Указывает путь к файлу, содержащему настройки конфигурации impala-shell. Путь по умолчанию — `/etc/impalarc`. Этот параметр можно указать только в командной строке
--live_progress	live_progress=true	Отображает индикатор выполнения, показывающий приблизительный процент выполнения каждого запроса. Информация обновляется в интерактивном режиме по мере обработки запроса. См. LIVE_PROGRESS Query Option
--disable_live_progress	live_progress=false	Позволяет отключить `live_progress` в интерактивном режиме
--live_summary	—	Отображает подробный отчет, аналогичный команде `SUMMARY`, в котором отображаются сведения о ходе выполнения каждой фазы запроса. Информация обновляется в интерактивном режиме по мере обработки запроса. См. LIVE_PROGRESS Query Option
-var=variable_name=value	—	Определяет подстановочную переменную, которую можно использовать в сессии impala-shell. Переменная может быть заменена в операторах, обрабатываемых опциями `-q` или `-f`, или в интерактивной сессии. В операторе SQL используется следующий синтаксис: `${var:variable_name}`
--auth_creds_ok_in_clear	auth_creds_ok_in_clear	Позволяет использовать аутентификацию LDAP при небезопасном подключении к impala-shell. Обратите внимание, что учетные данные для аутентификации отправляются в незашифрованном виде и могут быть уязвимы для атак
--protocol=protocol	—	Протокол, используемый для подключения к Impala. Допустимые значения: - `hs2` — impala-shell подключается к impalad через HiveServer2 по бинарному протоколу TCP. Это значение по умолчанию. - `hs2-http` — impala-shell подключается к impalad через HiveServer2 по HTTP. - `beeswax` — impala-shell подключается к impalad через Beeswax по бинарному протоколу TCP.
--hs2_fp_format=HS2_FP_FORMAT	hs2_fp_format=HS2_FP_FORMAT	Задает спецификацию формата вывода для значений с плавающей запятой при использовании протокола HS2. По умолчанию значения обрабатываются встроенным методом Python `str()`. Используйте `16G`, чтобы соответствовать выходному формату чисел с плавающей запятой протокола Beeswax

Конфигурационный файл impala-shell

Как упоминалось выше, можно сохранить набор настроек по умолчанию для impala-shell в конфигурационном файле.

Путь по умолчанию к глобальному файлу конфигурации impala-shell — */etc/impalarc*. Путь по умолчанию к файлу конфигурации пользовательского уровня — *~/.impalarc*. Обратите внимание, что имя файла глобального уровня не включает точку (`.`). Чтобы изменить путь по умолчанию к глобальному файлу конфигурации, установите переменную среды `$IMPALA_SHELL_GLOBAL_CONFIG_FILE`.

Поскольку файлы конфигурации ADH Impala расположены в каталоге */etc/impala/conf*, вы можете создать глобальный файл конфигурации impala-shell в этом каталоге и установить переменную `$IMPALA_SHELL_GLOBAL_CONFIG_FILE`:

```
$ export IMPALA_SHELL_GLOBAL_CONFIG_FILE=/etc/impala/conf
```

Также можно запустить impala-shell с параметром `--config_file`, устанавливающим путь к файлу конфигурации.

Параметры, указанные в файле конфигурации пользователя, имеют более высокий приоритет, чем параметры из глобального файла конфигурации. В свою очередь, любые параметры, указанные в командной строке impala-shell, переопределяют соответствующие параметры из файлов конфигурации.

Файл конфигурации impala-shell (глобальный или пользовательский) должен содержать заголовок `[impala]`, за которым следуют параметры.

Имена параметров в файле конфигурации аналогичны длинным формам аргументов командной строки impala-shell. Они перечислены в таблице выше.

Вы также можете указать опции, представляющие собой пары ключ/значение, используя `keyval`, аналогично параметру командной строки `--var`. Например, `keyval=variable1=value1`.

Файл конфигурации также может содержать параметры запроса в секции `[impala.query_options]`. Параметры, указанные в разделе `[impala]`, переопределяют параметры из раздела `[impala.query_options]`.

В приведенном ниже примере показан файл конфигурации, в котором включен подробный режим вывода, таким образом каждый вывод SQL-запроса сопровождается информацией о времени. impala-shell выполняет запросы к базе данных `test` по умолчанию. Вывод запроса форматируется как текст с разделителями и сохраняется в файле. Также включается опция `show_profiles`. Эта опция выводит подробную информацию о производительности после каждого запроса.

```
[impala]
verbose=true
default_db=test
write_delimited=true
output_delimiter=,
output_file=/home/test_result.csv
show_profiles=true
keyval=var1=value1,keyval=var2=value2
```

В следующем примере показан файл конфигурации, который подключается к определенному хосту Impala, выполняет один запрос к базе данных `test`, а затем завершает работу. Любые параметры запроса, предварительно определенные в разделе `[impala.query_options]`, применяются во время сессии.

```
[impala]
impalad=ees-adh1.ru-central1.internal
default_db=test
# Run the query and exit
query=select count(*) from customers

[impala.query_options]
mem_limit=32g
```

Вы можете использовать второй тип файла конфигурации с параметром командной строки `--config_file`, чтобы легко выбирать между множеством предопределенных запросов, которые можно запускать для разных баз данных, хостов или даже разных кластеров. Чтобы запустить последовательность операторов вместо одного запроса, создайте файл с операторами и укажите параметр конфигурации `query_file=path_to_query_file` в тексте запроса.

Подключение к impalad на определенном хосте

Для подключения к демону impalad на другом хосте необходимо указать параметры подключения одним из описанных ниже способов.

С помощью параметров командной строки

Вы можете использовать параметры командной строки при запуске команды impala-shell:

```
$ impala-shell -i ees-1adh3.ru-central1.internal:28000 --protocol='hs2-http'
```

Этот пример подключается к impalad на хосте `ees-1adh3.ru-central1.internal` и порте 28000, через HiveServer2 по HTTP.

С помощью конфигурационного файла

Параметры подключения можно указать в файле. Создайте файл с настройками подключения, как описано в разделе Конфигурационный файл impala-shell. Например:

```
$ cat > ~/custom_conf1
[impala]
impalad=ees-1adh2.ru-central1.internal
query=select version()
```

При использовании команды `cat` нажмите `CTRL+D`, чтобы выйти.

Запустите impala-shell:

```
$ impala-shell --config_file=~/custom_conf1
```

Результат:

```
Starting Impala Shell with no authentication using Python 2.7.5
Warning: live_progress only applies to interactive shell sessions, and is being skipped for now.
Opened TCP connection to ees-1adh2.ru-central1.internal:21050
Connected to ees-1adh2.ru-central1.internal:21050
Server version: impalad version 4.2.0-RELEASE RELEASE (build 978afcfae9aa626de182b2872c4469646f42e0f6)
Query: select version()
Query submitted at: 2023-07-25 14:36:57 (Coordinator: http://ees-1adh2.ru-central1.internal:25000)
Query progress can be monitored at: http://ees-1adh2.ru-central1.internal:25000/query_plan?
query_id=084b59535cac1baf:5f5e61f500000000
+-------------------------------------------------------------------------------------------+
| version()                                                                                 |
+-------------------------------------------------------------------------------------------+
| impalad version 4.2.0-RELEASE RELEASE (build 978afcfae9aa626de182b2872c4469646f42e0f6) |
| Built on Mon Jun  5 16:33:03 UTC 2023                                                     |
+-------------------------------------------------------------------------------------------+
Fetched 1 row(s) in 0.44s
```

С помощью команды CONNECT

Выполните команду `CONNECT` во время сессии impala-shell:

```
$ impala-shell
```

```
connect ees-1adh3.ru-central1.internal;
```

Результат:

```
Opened TCP connection to ees-1adh3.ru-central1.internal:21050
Connected to ees-1adh3.ru-central1.internal:21050
Server version: impalad version 4.2.0-RELEASE RELEASE (build 978afcfae9aa626de182b2872c4469646f42e0f6)
[ees-1adh3.ru-central1.internal:21050] default>
```

Подключение к определенной базе данных

Вы можете использовать все те же параметры подключения, что и в предыдущих примерах, и добавить параметр `-d` с именем базы данных:

```
$ impala-shell -i localhost -d my_db
```

`localhost` ссылается на тот же хост, на котором работает impala-shell.

Выполнение операторов SQL в неинтерактивном режиме

Вызовите impala-shell с параметром `-q` для запуска единичного оператора или с параметром `-f` для запуска последовательности операторов из файла. Команда impala-shell возвращает результат сразу, без перехода в интерактивный режим:

```
$ impala-shell -i localhost -d my_db -q 'show tables'

$ impala-shell -i localhost -d my_db -f recreate_tables.sql
```

Выполнение команд и операторов SQL

Интерпретатор impala-shell принимает операторы SQL, описанные в статье Impala SQL Statements, и команды оболочки, которые можно использовать для настройки работоспособности и диагностики проблем.

В неинтерактивном режиме результат запроса выводится в stdout или в файл, указанный в параметре `-o`. Ошибки выводятся в stderr. Таким образом, вы можете обрабатывать результат запроса как часть пайплайна Unix.

В интерактивном режиме impala-shell использует библиотеку Readline для вызова и редактирования предыдущих команд.

Вы можете использовать следующие сочетания клавиш и правила синтаксиса для запуска команд и операторов SQL в impala-shell:

- Чтобы увидеть полный список доступных команд, дважды нажмите `TAB`.
- Чтобы пролистать предыдущие команды, нажмите клавиши со стрелками вверх и вниз.
- Используйте стандартный набор сочетаний клавиш в библиотеке GNU Readline для редактирования и перемещения курсора, например `Ctrl+A` для перемещения курсора в начало строки и `Ctrl+E` — в конец строки.
- Команды и операторы SQL могут занимать несколько строк и должны заканчиваться точкой с запятой (`;`).
- Используйте `--`, чтобы закомментировать однострочный комментарий, и `/* <content> */`, чтобы закомментировать несколько строк. Комментарий рассматривается как часть оператора, которому он предшествует. Когда вы вводите комментарий, вы получаете запрос на продолжение, пока не введете точку с запятой. Например:

```
default> -- Show tables starting with t
> SHOW TABLES LIKE 't*';
```

- Если комментарий содержит `${variable_name}` и не предназначен для замены переменной, символ `$` должен быть заэкранирован, например — `\${hello}`.

Приведенный ниже пример использует и команды impala-shell и операторы SQL, чтобы создать базу данных `test` и таблицу `table1`:

```
create database test;
use test;
create table table1 (field1 int, field2 boolean, field3 string);
insert into table1 values ((1,true,"row1"), (2,true,"row2"), (3,false,"row3"));
select * from table1;
```

Результат:

```
+--------+--------+--------+
| field1 | field2 | field3 |
+--------+--------+--------+
| 1      | true   | row1   |
| 2      | true   | row2   |
| 3      | false  | row3   |
+--------+--------+--------+
```

Подстановка переменных

Можно определить переменные, которые будут подставляться в операторы SQL, обрабатываемые impala-shell. Для этого в командной строке укажите параметр `--var=variable_name=value`. В интерактивной сессии или в файле скрипта, переданного с помощью параметра `-f`, используйте команду `SET VAR:variable_name=value`. Чтобы добавить переменную в оператор SQL, используйте запись: `${VAR:variable_name}`.

Примеры ниже содержат несколько команд, определяющих переменные, и используют их в операторах SQL, выполняемых с помощью `-q`. Строки аргументов `-q`, в которых есть ссылки на переменные `${var:value}`, заключены в одинарные кавычки.

Пример 1:

```
$ impala-shell --var=tname=table1 --var=colname=col1 --var=coltype=string -q 'CREATE TABLE ${var:tname} (${var:colname}
${var:coltype}) STORED AS PARQUET'
```

Результат:

```
Query: CREATE TABLE table1 (col1 string) STORED AS PARQUET
+-------------------------+
| summary                 |
+-------------------------+
| Table has been created. |
+-------------------------+
```

Пример 2:

```
$ impala-shell --quiet --var=tname=table1
```

```
use test;
select count(*) from ${var:tname};
```

Результат:

```
+----------+
| count(*) |
+----------+
| 3        |
+----------+
```

Пример 3:

```
$ impala-shell
```

```
set var:tname=table1;
select count(*) from ${var:tname};
```

Результат:

```
+----------+
| count(*) |
+----------+
| 3        |
+----------+
```

Подключение к Impala через JDBC

Евгения Кузина

Содержание

Назад к содержанию

Настройка порта JDBC

Включение поддержки Impala JDBC на клиентских системах

Строки подключения для драйвера Hive JDBC

Ограничения JDBC

Impala поддерживает стандартный интерфейс JDBC, который может быть использован для обеспечения доступа к данным из коммерческих инструментов Business Intelligence и другого программного обеспечения.

Чтобы настроить подключение JDBC к Impala, выполните следующие действия:

1. Проверьте порт, на котором демоны Impala прослушивают входящие запросы JDBC.
2. Установите драйвер JDBC на каждый хост, на котором работает приложение с поддержкой JDBC.
3. Укажите строку подключения для приложения JDBC к одному из серверов, на котором работает демон impalad, с соответствующими настройками безопасности.

Настройка порта JDBC

Значения портов по умолчанию, на которых демоны Impala принимают соединения JDBC, перечислены в таблице ниже.

Протокол	Значение по умолчанию	Параметр
HTTP	28000	hs2_http_port
Binary TCP	21050	hs2_port

Чтобы изменить эти порты, перейдите на страницу ***Components*** сервиса ***Impala*** в пользовательском интерфейсе ADCM, выберите ***Impala Daemon***, откройте вкладку ***Primary Configuration*** и укажите необходимый параметр (***hs2_http_port*** или ***hs2_port***).

Включение поддержки Impala JDBC на клиентских системах

Драйвер Hive JDBC установлен на хостах с Impala в кластере ADH. Он представляет собой пакет *hive-jdbc.noarch*.

Если вы используете приложения с поддержкой JDBC на хостах за пределами кластера ADH, установите драйвер JDBC (пакет *hive-jdbc*) по крайней мере на одном хосте внешнего кластера. Для этого можно использовать менеджер пакетов Linux. Затем загрузите следующие файлы JAR на каждую клиентскую машину, которая должна использовать JDBC с Impala:

- commons-logging-X.X.X.jar
- hadoop-common.jar
- hive-common-X.XX.X.jar
- hive-jdbc-X.XX.X.jar
- hive-metastore-X.XX.X.jar
- hive-service-X.XX.X.jar
- httpclient-X.X.X.jar
- httpcore-X.X.X.jar
- libfb303-X.X.X.jar
- libthrift-X.X.X.jar
- log4j-X.X.XX.jar
- slf4j-api-X.X.X.jar
- slf4j-logXjXX-X.X.X.jar

Чтобы включить поддержку JDBC для Impala в системе, где вы запускаете приложение JDBC, выполните следующие шаги:

1. Загрузите файлы JAR, перечисленные выше, на каждый клиентский хост. Для пользователей Maven обратитесь к странице the sample github page, содержащей пример зависимостей, которые вы можете добавить в файл *pom* вместо загрузки отдельных JAR-файлов.
2. Сохраните файлы JAR. Можно сделать это в каталоге, на который уже есть ссылка в CLASSPATH. В Linux можно использовать расположение */opt/jars/*. В Windows вы можете сохранить файлы в подкаталог папки *C:\Program Files*.
3. Установите CLASSPATH, включающий путь к файлам JAR. Чтобы успешно загрузить драйвер Impala JDBC, клиентские программы должны найти связанные файлы JAR. Для этого в большинстве случаев нужно установить переменную CLASSPATH. Несколько примеров установки переменной приведены ниже. За дополнительными сведениями об установке драйверов JDBC обратитесь к документации вашего клиента JDBC.
   В Linux, если вы извлекли файлы JAR в */opt/jars/*, можно выполнить следующую команду:

```
$ export CLASSPATH=/opt/jars/*.jar:$CLASSPATH
```

В Windows откройте ***Свойства системы*** в ***Панели управления*** и измените переменные среды. Установите переменные среды таким образом, чтобы они включали путь к извлеченным файлам JAR.

Имя класса драйвера Hive JDBC — `org.apache.hive.jdbc.HiveDriver`.

Строки подключения для драйвера Hive JDBC

После того как Impala настроена для работы с JDBC, можно устанавливать соединение.

ВАЖНО

При использовании HTTP в качестве транспорта следует выполнить команду `kinit` перед установкой соединения с БД.

Без Kerberos

Для кластера, который не использует аутентификацию Kerberos, используйте следующий синтаксис строки подключения:

```
jdbc:hive2://<hostname>:<port>/;auth=noSasl
```

Пример для TCP:

```
jdbc:hive2://host1.example.com:21050/;auth=noSasl
```

Пример для HTTP:

```
jdbc:hive2://host1.example.com:28000/;transportMode=http;auth=noSasl
```

SSL

Чтобы подключиться к экземпляру Impala с использованием SSL:

```
jdbc:hive2://<hostname>:<port>/;ssl=true;sslTrustStore=<trust_store_path>;trustStorePassword=<trust_store_password>;principal=
<principal_name>
```

Пример для TCP:

```
jdbc:hive2://host1.example.com:21050/;ssl=true;sslTrustStore=/tmp/truststore.
jks;trustStorePassword=bigdata;principal=hive/_HOST@EXAMPLE.COM
```

Пример для HTTP:

```
jdbc:hive2://host1.example.com:28000/;transportMode=http;ssl=true;sslTrustStore=/tmp/truststore.
jks;trustStorePassword=bigdata;principal=hive/_HOST@EXAMPLE.COM
```

SSL+Kerberos

Чтобы подключиться к экземпляру Impala, для которого требуется аутентификация Kerberos с использованием SSL:

```
jdbc:hive2://<hostname>:<port>/;principal=<principal_name>/hostname@EXAMPLE.COM;ssl=true;sslTrustStore=
<truststore_path>;trustStorePassword=<truststore_pass>
```

Принципал должен быть принципалом того же пользователя, который используется при запуске Impala.

Пример для TCP:

```
jdbc:hive2://host1.example.com:21050/;principal=impala/host1.example.com@EXAMPLE.COM;ssl=true;sslTrustStore=/tmp/truststore.
jks;trustStorePassword=bigdata
```

Пример для HTTP:

```
jdbc:hive2://host1.example.com:28000/;transportMode=http;principal=impala/host1.example.com@EXAMPLE.
COM;ssl=true;sslTrustStore=/tmp/truststore.jks;trustStorePassword=bigdata
```

LDAP

Чтобы подключиться к экземпляру Impala, для которого требуется аутентификация LDAP:

```
jdbc:hive2://<hostname>:<port>/<db_name>;user=<ldap_userid>;password=<ldap_password>
```

Пример для TCP:

```
jdbc:hive2://host1.example.com:21050/test_db;user=user1;password=password1
```

Пример для HTTP:

```
jdbc:hive2://host1.example.com:28000/test_db;transportMode=http;user=user1;password=password1
```

Ограничения JDBC

Большинство функций SQL Impala работают одинаково через интерпретатор impala-shell и через API JDBC, но есть некоторые исключения:

- Запросы со сложными типами (`ARRAY`, `STRUCT` и `MAP`) требуют обозначения, которое может быть недоступно на всех уровнях драйвера JDBC. Вы можете создать представление (view), являющееся упрощенной версией сложных типов, и настроить приложение на работу с представлением. За дополнительной информацией обратитесь к статье Complex Types.
- Сложные типы Impala поддерживаются методом JDBC `getColumns()`. И `MAP`, и `ARRAY` обрабатываются как SQL-тип JDBC `ARRAY`. Такое поведение соответствует поведению Hive. Тип `STRUCT` обрабатывается как SQL-тип JDBC `STRUCT`. Для поддержки совместимости с Hive, поле `TYPE_NAME` для скалярных типов заполняется именами примитивных типов. Для сложных типов выполняется метод `toSQL()`. Полученные имена типов не полностью консистентны. В таблице ниже показано, как `toSQL()` преобразует типы.

Исходный тип	Результирующий тип
DECIMAL(10,10)	DECIMAL
CHAR(10)	CHAR
VARCHAR(10)	VARCHAR
ARRAY<DECIMAL(10,10)>	ARRAY<DECIMAL(10,10)>
ARRAY<CHAR(10)>	ARRAY<CHAR(10)>
ARRAY<VARCHAR(10)>	ARRAY<VARCHAR(10)>

- В настоящее время операторы Impala `INSERT`, `UPDATE` и другие операторы DML, выполняемые через интерфейс JDBC над таблицами Kudu, не возвращают ошибок JDBC при таких условиях, как, например, дублирование столбцов первичного ключа. Тем не менее в приложениях, которые выполняют большое количество операторов DML, лучше использовать непосредственно Kudu Java API вместо JDBC.

Обзор веб-интерфейсов Impala

Евгения Кузина

Содержание

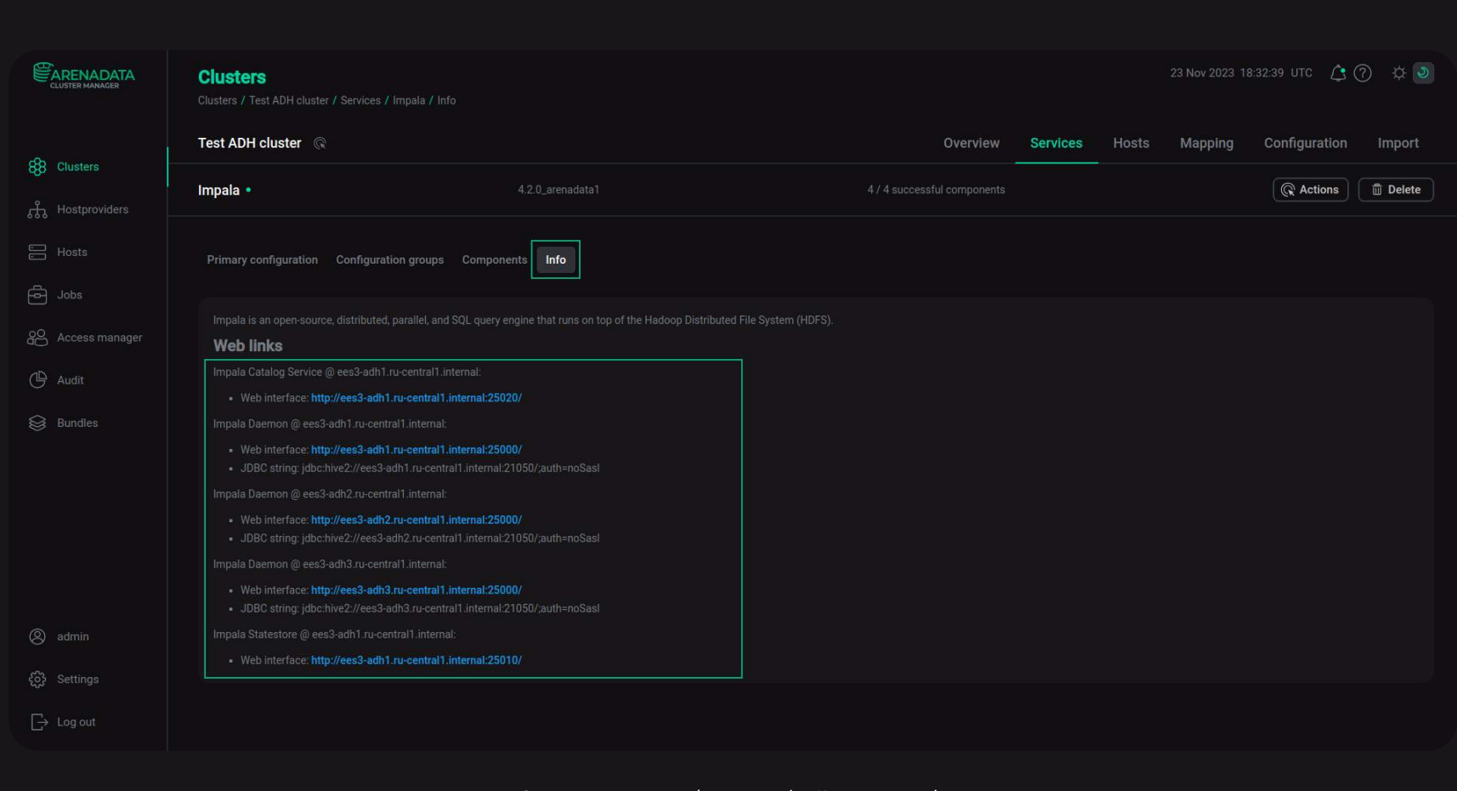
Назад к содержанию

Веб-интерфейс Impala Daemon (impalad)

Веб-интерфейс Impala Statestore (statestored)

Веб-интерфейс Impala Catalog Service (catalogd)

У каждого из демонов Impala (*impalad*, *statestored* и *catalogd*) есть встроенный веб-сервер, который отображает диагностическую информацию и информацию о состоянии демона. Ссылки на веб-интерфейсы можно найти на вкладке ***Info*** сервиса Impala в пользовательском интерфейсе ADCM.



Impala is an open-source, distributed, parallel, and SQL query engine that runs on top of the Hadoop Distributed File System (HDFS).

Web links

Impala Catalog Service @ ees3-adh1.ru-central1.internal:
- Web interface: http://ees3-adh1.ru-central1.internal:25020/

Impala Daemon @ ees3-adh1.ru-central1.internal:
- Web interface: http://ees3-adh1.ru-central1.internal:25000/
- JDBC string: jdbc:hive2://ees3-adh1.ru-central1.internal:21050/;auth=noSasl

Impala Daemon @ ees3-adh2.ru-central1.internal:
- Web interface: http://ees3-adh2.ru-central1.internal:25000/
- JDBC string: jdbc:hive2://ees3-adh2.ru-central1.internal:21050/;auth=noSasl

Impala Daemon @ ees3-adh3.ru-central1.internal:
- Web interface: http://ees3-adh3.ru-central1.internal:25000/
- JDBC string: jdbc:hive2://ees3-adh3.ru-central1.internal:21050/;auth=noSasl

Impala Statestore @ ees3-adh1.ru-central1.internal:
- Web interface: http://ees3-adh1.ru-central1.internal:25010/

Ссылки на веб-интерфейсы Impala

URL-адрес имеет следующую форму: `http://<server-hostname>:<port>/`.

Пример: `http://adh-host1.ru-central1.internal:25000/`.

Для защищённых кластеров используйте `https://` вместо `http://`.

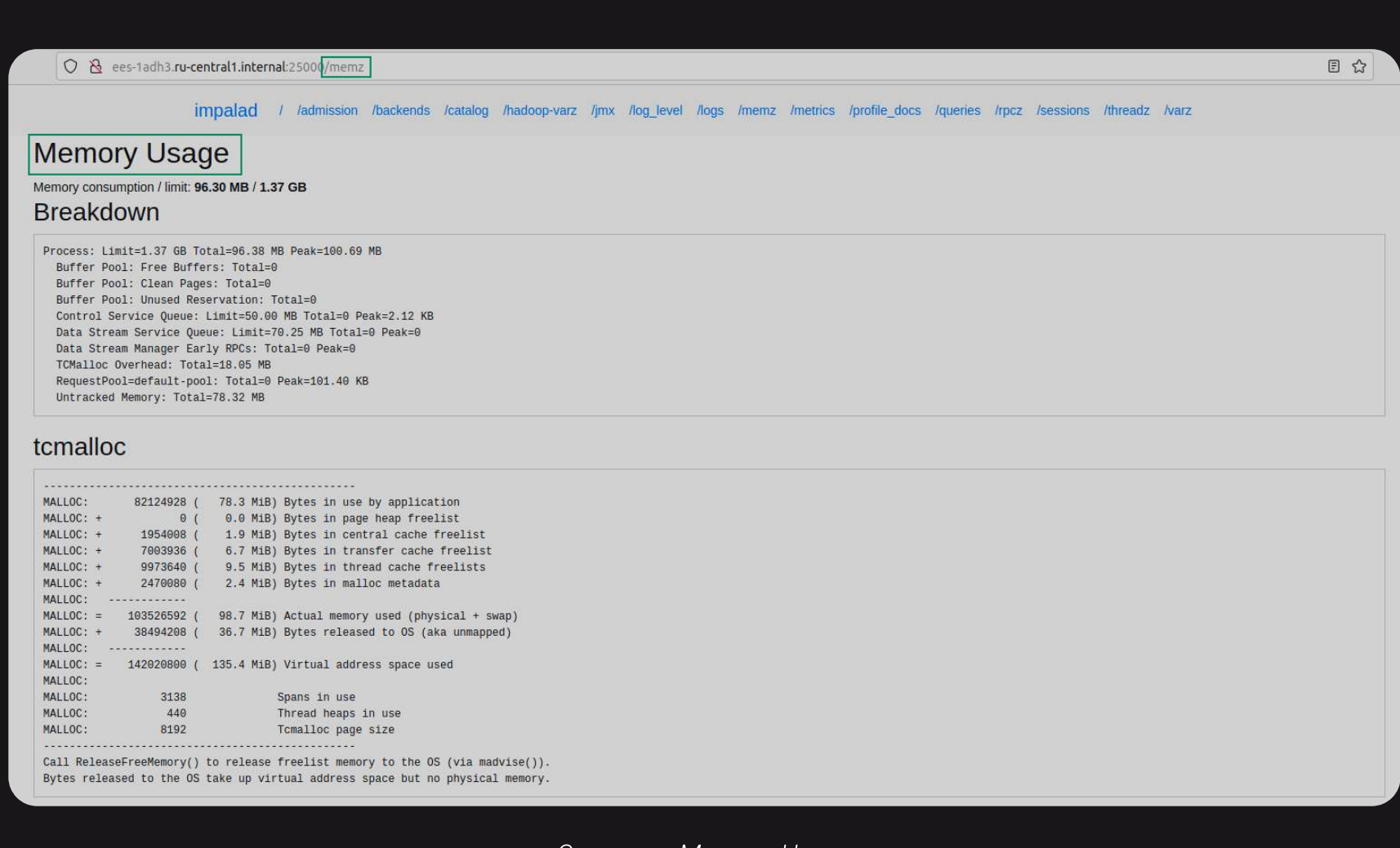
Порты веб-серверов, используемые по умолчанию, перечислены в таблице ниже.

Компонент	Демон	Порт
Impala Daemon	impalad	25000
Impala Statestore	statestored	25010
Impala Catalog Service	catalogd	25020

Чтобы изменить порт веб-сервера, откройте вкладку ***Primary Configuration*** соответствующего компонента Impala и установите параметр ***webserver_port***. Чтобы отключить веб-интерфейс, снимите флажок ***enable_webserver*** и перезапустите требуемый компонент с помощью действия компонента ***Restart Impala Daemon***, или ***Restart Impala Statestore***, или ***Restart Impala Catalog***.

Веб-интерфейсы Impala поддерживают SPNEGO, чтобы обеспечить защиту по протоколу Kerberos. SPNEGO передает токены согласования через заголовки HTTP для защиты HTTP-запросов с помощью Kerberos. Чтобы включить авторизацию с помощью SPNEGO, установите флажок ***webserver_require_spnego*** (на вкладке компонента ***Primary Configuration*** в интерфейсе ADCM) для каждого компонента, который должен использовать SPNEGO.

Вы можете использовать меню в верхней части веб-интерфейсов для навигации по страницам, описанным ниже. Также можно добавить слеш (`/`) и имя страницы (например, `/memz`) к HTTP-адресу в браузере, чтобы на неё перейти.



Memory Usage

Memory consumption / limit: **96.30 MB / 1.37 GB**

Breakdown

```
Process: Limit=1.37 GB Total=96.38 MB Peak=100.69 MB
  Buffer Pool: Free Buffers: Total=0
  Buffer Pool: Clean Pages: Total=0
  Buffer Pool: Unused Reservation: Total=0
  Control Service Queue: Limit=50.00 MB Total=0 Peak=2.12 KB
  Data Stream Service Queue: Limit=70.25 MB Total=0 Peak=0
  Data Stream Manager Early RPCs: Total=0 Peak=0
  TCMalloc Overhead: Total=18.05 MB
  RequestPool=default-pool: Total=0 Peak=101.40 KB
  Untracked Memory: Total=78.32 MB
```

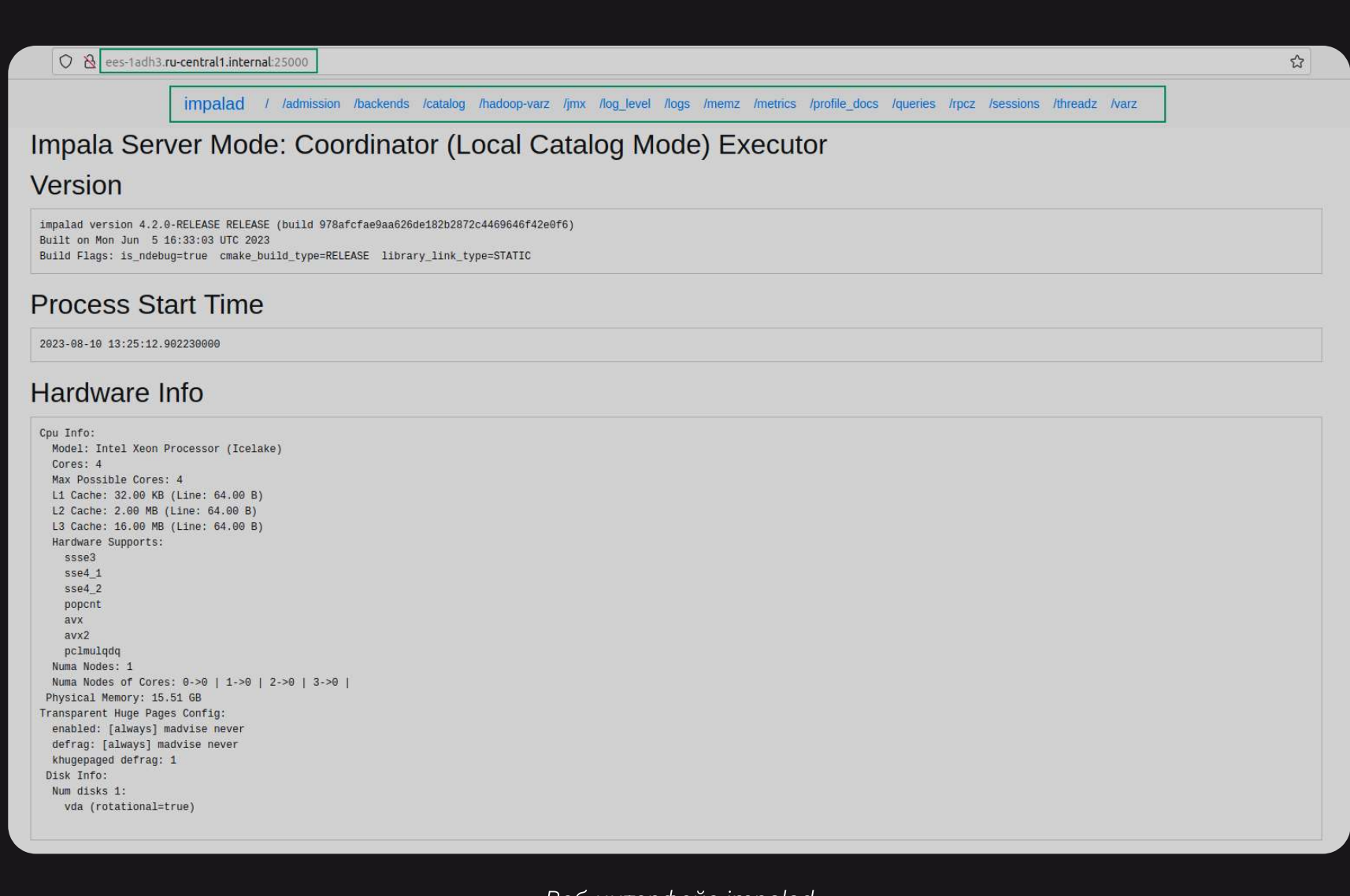
tcmalloc

```
------------------------------------------------
MALLOC:       82124928 (   78.3 MiB) Bytes in use by application
MALLOC: +            0 (    0.0 MiB) Bytes in page heap freelist
MALLOC: +      1954008 (    1.9 MiB) Bytes in central cache freelist
MALLOC: +      7003936 (    6.7 MiB) Bytes in transfer cache freelist
MALLOC: +      9973640 (    9.5 MiB) Bytes in thread cache freelists
MALLOC: +      2470080 (    2.4 MiB) Bytes in malloc metadata
MALLOC:   ------------
MALLOC: =    103526592 (   98.7 MiB) Actual memory used (physical + swap)
MALLOC: +     38494208 (   36.7 MiB) Bytes released to OS (aka unmapped)
MALLOC:   ------------
MALLOC: =    142020800 (  135.4 MiB) Virtual address space used
MALLOC:
MALLOC:           3138              Spans in use
MALLOC:            440              Thread heaps in use
MALLOC:           8192              Tcmalloc page size
------------------------------------------------
Call ReleaseFreeMemory() to release freelist memory to the OS (via madvise()).
Bytes released to the OS take up virtual address space but no physical memory.
```

Страница Memory Usage

Веб-интерфейс Impala Daemon (impalad)

Веб-интерфейс impalad содержит информацию о параметрах конфигурации, выполняемых и завершенных запросах, а также о производительности и использовании ресурсов.



Impala Server Mode: Coordinator (Local Catalog Mode) Executor

Version

```
impalad version 4.2.0-RELEASE RELEASE (build 978afcfae9aa626de182b2872c4469646f42e0f6)
Built on Mon Jun  5 16:33:03 UTC 2023
Build Flags: is_ndebug=true  cmake_build_type=RELEASE  library_link_type=STATIC
```

Process Start Time

```
2023-08-10 13:25:12.902230000
```

Hardware Info

```
Cpu Info:
  Model: Intel Xeon Processor (Icelake)
  Cores: 4
  Max Possible Cores: 4
  L1 Cache: 32.00 KB (Line: 64.00 B)
  L2 Cache: 2.00 MB (Line: 64.00 B)
  L3 Cache: 16.00 MB (Line: 64.00 B)
  Hardware Supports:
    ssse3
    sse4_1
    sse4_2
    popcnt
    avx
    avx2
    pclmulqdq
  Numa Nodes: 1
  Numa Nodes of Cores: 0->0 | 1->0 | 2->0 | 3->0 |
 Physical Memory: 15.51 GB
Transparent Huge Pages Config:
  enabled: [always] madvise never
  defrag: [always] madvise never
  khugepaged defrag: 1
 Disk Info:
  Num disks 1:
    vda (rotational=true)
```

Веб-интерфейс impalad

Каждый хост impalad имеет собственный экземпляр веб-интерфейса с подробной информацией о запросах, для которых этот хост был координатором. Для исследования проблемных запросов необходимо выбрать ноду, которая была для них координатором.

Главная страница веб-интерфейса impalad содержит следующую информацию об impalad:

- ***Version*** — версия демона и информация о том, когда была собрана Impala и какие флаги сборки использовались.
- ***Process start time*** — время запуска демона.
- ***Hardware info*** — информация об оборудовании хоста.
- ***OS information*** — информация об операционной системе, которая установлена на хосте.
- ***Process information*** — информация о процессе демона.
- ***CGroup information*** — информация о настройках CGroup.

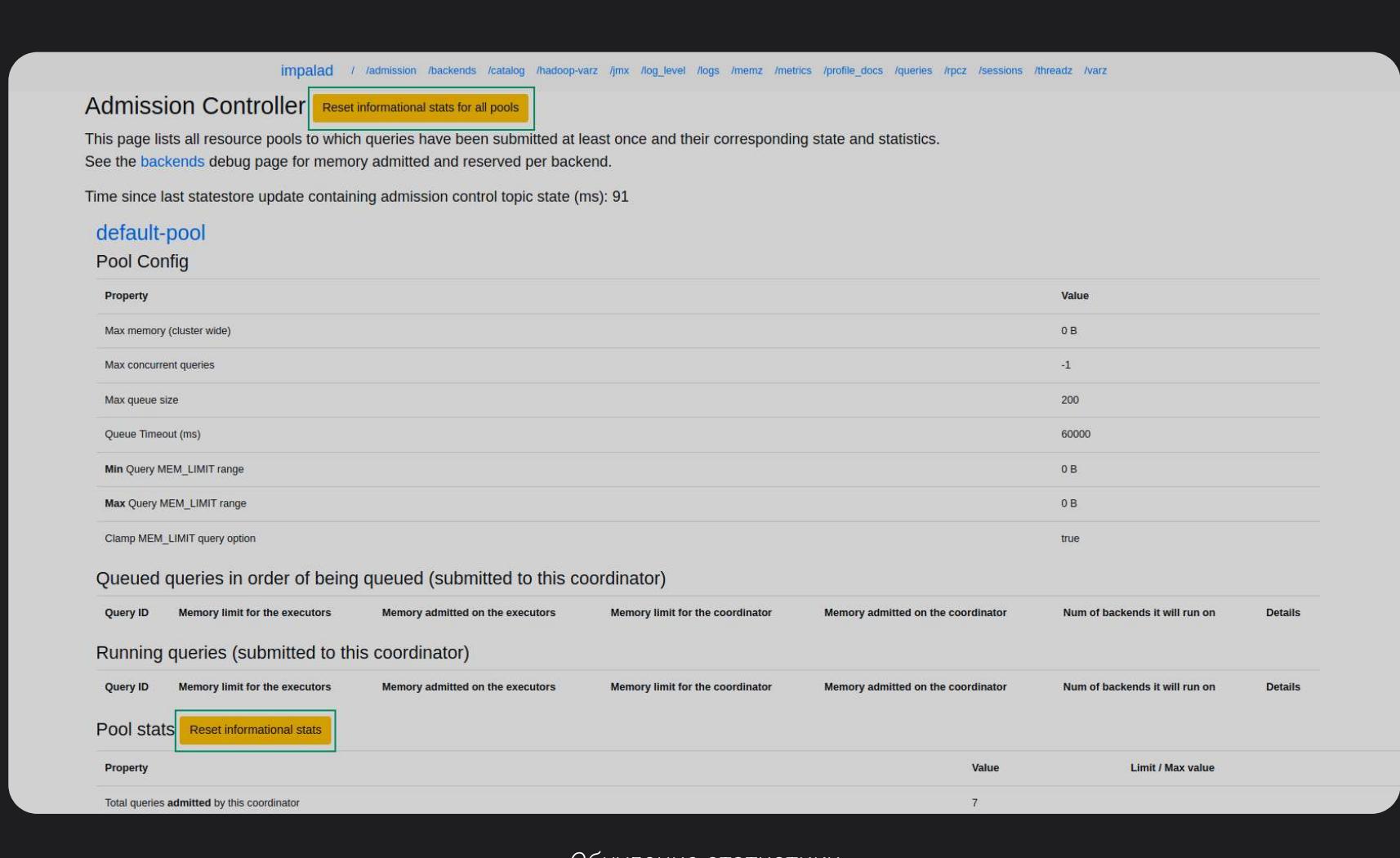
admission

На странице ***Admission Controller*** перечислены все пулы ресурсов, в которые запросы были отправлены хотя бы один раз, а также их состояние и статистика.

На странице представлена следующая информация о каждом пуле ресурсов:

- Время, прошедшее с того момента, как statestored получил последнее обновление.
- Предупреждение, если этот impalad отключен от statestored и информация на странице может быть устаревшей.
- Конфигурация пула.
- Запросы в очереди, отправленные этому координатору, в порядке отправки.
- Выполняемые запросы этого координатора.
- Статистика пула. Время в очереди рассчитывается как экспоненциальное скользящее среднее, которое представляет собой среднее время в очереди последних 10–12 запросов. Если запрос принимается немедленно, это среднее время рассчитывается с использованием времени ожидания `0`.
- Гистограмма пикового распределения памяти. Она учитывает память, используемую запросами, допущенными к пулу. На гистограмме отображаются данные для всех запросов, допущенных в пул, включая завершенные, отмененные и закончившиеся неудачей.

Вы можете кликнуть ***Reset informational stats***, чтобы обнулить статистические данные, в том числе итоговую статистику, время в очереди и гистограмму для определенного пула или нажать ***Reset informational stats for all pulls***, чтобы очистить статистику для всех пулов сразу.



Admission Controller

This page lists all resource pools to which queries have been submitted at least once and their corresponding state and statistics.
See the backends debug page for memory admitted and reserved per backend.

Time since last statestore update containing admission control topic state (ms): 91

default-pool

Pool Config

Property	Value
Max memory (cluster wide)	0 B
Max concurrent queries	-1
Max queue size	200
Queue Timeout (ms)	60000
Min Query MEM_LIMIT range	0 B
Max Query MEM_LIMIT range	0 B
Clamp MEM_LIMIT query option	true

Queued queries in order of being queued (submitted to this coordinator)

Query ID	Memory limit for the executors	Memory admitted on the executors	Memory limit for the coordinator	Memory admitted on the coordinator	Num of backends it will run on	Details

Running queries (submitted to this coordinator)

Query ID	Memory limit for the executors	Memory admitted on the executors	Memory limit for the coordinator	Memory admitted on the coordinator	Num of backends it will run on	Details

Pool stats

Property	Value	Limit / Max value
Total queries **admitted** by this coordinator	7	

Обнуление статистики

Статистика Admission Controller также доступна в виде объекта JSON из следующей конечной точки HTTP (HTTP endpoint): `http://<server-hostname>:<port>/admission?json`. Пример: `/http://adh-host1.ru-central1.internal:25000/admission?json`. За дополнительной информацией обратитесь к статье Admission Control and Query Queuing.

backends

На этой странице приведена следующая информация для каждого из узлов impalad в кластере:

- ***Address*** — адрес ноды (имя хоста и порт).
- ***KRPC Address*** — KRPC-адрес ноды. Может быть использован в команде `SHUTDOWN`.
- ***Backend Id***.
- ***Coordinator*** — логическое значение, показывающее, является ли нода координатором.
- ***Executor*** — логическое значение, показывающее, является ли нода исполнителем запросов.
- ***Memory Limit for Admission*** — объем памяти, который выделен контроллером допуска этой ноде.
- ***Memory Reserved*** — объем памяти, зарезервированный для запросов на этой ноде.
- ***Memory Admitted by Queries Submitted to this Coordinator*** — память, необходимая для запросов, обрабатываемых этим координатором.
- ***Admission Control Slots In Use*** — количество используемых слотов контроллера доступа.
- ***Num. Queries Admitted by this Coordinator*** — количество запросов, обрабатываемых текущим координатором.
- ***Executor Groups***.

Если в кластере есть какой-либо остановленный демон impalad, ноды разбиваются на две таблицы: ***Active backend(s)*** и ***Quiescing backend(s)***.

Поскольку каждый демон impalad знает о других демонах impalad через Statestore, все веб-серверы impalad должны отображать одну и ту же информацию.

catalog

На странице ***Catalog*** отображается список баз данных и связанных с ними таблиц. Вы можете использовать эту страницу, чтобы определить, какая база данных включает определённую таблицу, проверить правильность написания имени базы данных или таблицы, найти одинаковые имена таблиц в нескольких базах данных. Основным сценарием использования этой страницы при отладке является проверка того, знает ли экземпляр impalad о таблице, которая находится в определенной базе данных.

hadoop-varz

На этой странице отображаются общие параметры конфигурации Hadoop, с которыми работает Impala.

jmx

На этой странице отображается информация мониторинга о различных подсистемах JVM в формате JSON, например, о пулах памяти, управлении потоками, времени выполнения и других.

log_level

На этой странице отображаются текущие уровни важности сообщений, попадающих в логи Java и серверной части демона. Также страница позволяет динамически изменять уровни логирования без перезапуска демона.

logs

Страница отображает последние записи из лог-файла *impalad.INFO*, включающего сообщения с уровнем `info`, `warning` и `error`. Позволяет просмотреть сведения о последних операциях, чтобы определить, были ли операции успешными или возникли ошибки.

Эта страница позволяет избежать поиска лог-файлов в файловой системе.

memz

На этой странице отображается сводная и подробная информация об использовании памяти демоном impalad.

metrics

На странице отображается текущий набор метрик, счетчиков и флагов, представляющих различные аспекты внутренних операций демона.

profile_docs

На странице перечислены счетчики оператора `PROFILE`.

queries

Страница содержит следующие таблицы:

- ***Queries in flight*** — выполняемые запросы.
- ***Waiting to be closed*** — запросы, которые закончили выполнять действия, но еще не были завершены.
- ***Last 100 completed queries*** — последние завершенные запросы. Вы можете управлять количеством запросов и, следовательно, объемом памяти, выделенной для хранения информации о завершенных запросах, используя параметр запуска `--query_log_size` для impalad.
- ***Query Locations*** — отображает, как запущенные запросы распределяются между хостами impalad.

Запросы перечислены в обратном хронологическом порядке, самые последние находятся вверху таблицы.

Ссылка ***Details*** для каждого запроса отображает альтернативные представления запроса, включая графическое представление плана и вывод операторов `EXPLAIN`, `SUMMARY` и `PROFILE` из impala-shell. На вкладке ***Profile*** страницы ***Details*** можно экспортировать выходные данные в Thrift, JSON или текстовый формат.

rpcz

На странице отображается информация о продолжительности RPC-соединений демона с другими демонами Impala.

sessions

На странице отображается информация о сессиях, подключенных к этому экземпляру impalad. Например, сессии могут включать соединения из impala-shell или JDBC.

threadz

Страница отображает информацию о потоках (threads), используемых демоном, и о категориях, в которые они сгруппированы.

varz

Страница отображает параметры конфигурации, которые используются, когда этот демон взаимодействует с другими компонентами Hadoop, такими как HDFS и YARN. Параметры собираются из конфигурационных файлов.

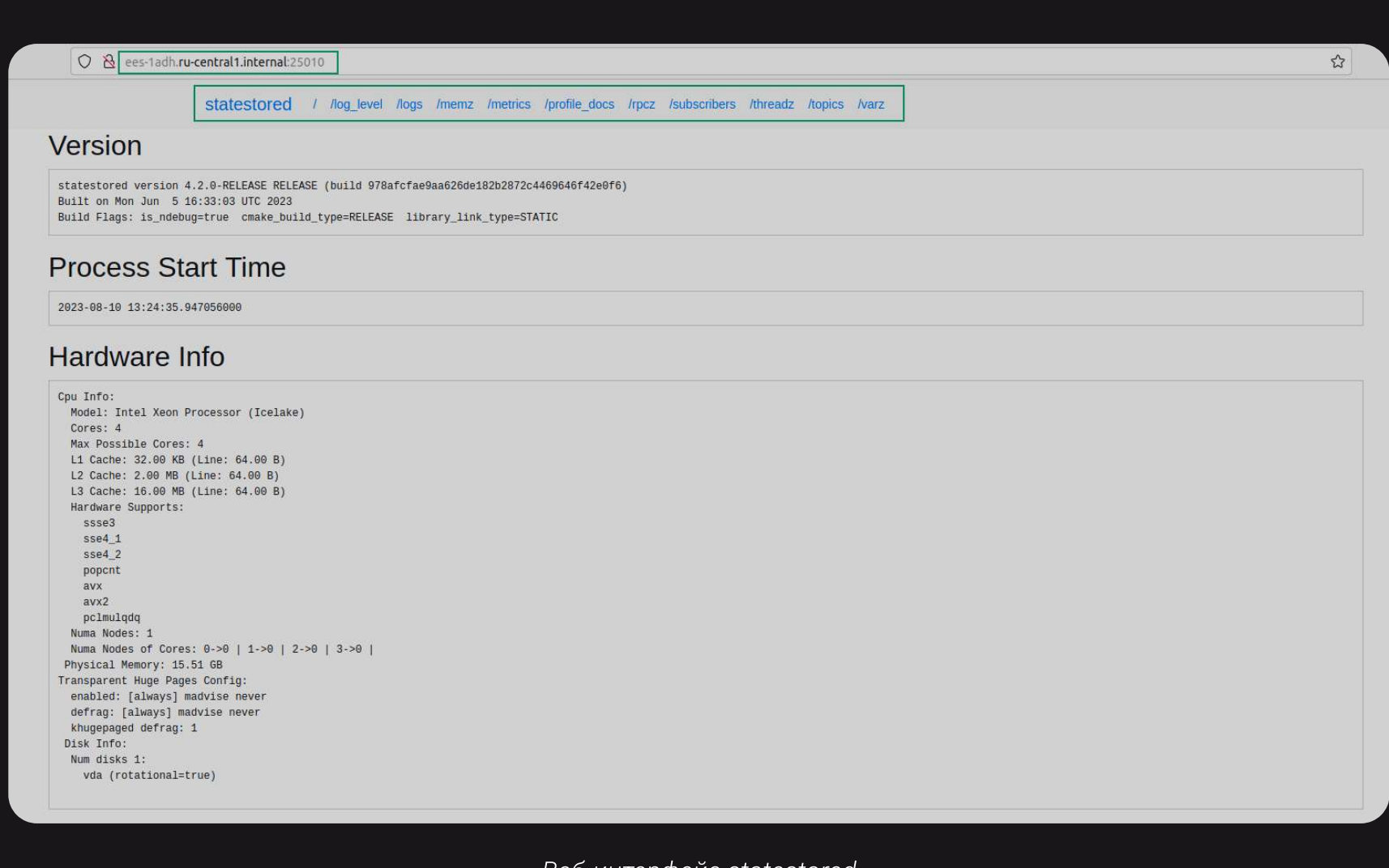
Prometheus metrics

На странице представлены метрики в формате Prometheus. Prometheus может использовать эти данные для мониторинга и оповещения.

Меню не содержит опции `/metrics_prometheus`. Чтобы отобразить эту страницу, добавьте `/metrics_prometheus` к HTTP-адресу в браузере.

Веб-интерфейс Impala Statestore (statestored)

Веб-интерфейс statestored включает информацию об использовании памяти, параметрах конфигурации и текущих проверках работоспособности. Поскольку в кластере есть только одна нода с statestored, вы можете получить доступ к веб-интерфейсу statestored только на хосте с компонентом Impala Statestore.



Version

```
statestored version 4.2.0-RELEASE RELEASE (build 978afcfae9aa626de182b2872c4469646f42e0f6)
Built on Mon Jun  5 16:33:03 UTC 2023
Build Flags: is_ndebug=true  cmake_build_type=RELEASE  library_link_type=STATIC
```

Process Start Time

```
2023-08-10 13:24:35.947056000
```

Hardware Info

```
Cpu Info:
  Model: Intel Xeon Processor (Icelake)
  Cores: 4
  Max Possible Cores: 4
  L1 Cache: 32.00 KB (Line: 64.00 B)
  L2 Cache: 2.00 MB (Line: 64.00 B)
  L3 Cache: 16.00 MB (Line: 64.00 B)
  Hardware Supports:
    ssse3
    sse4_1
    sse4_2
    popcnt
    avx
    avx2
    pclmulqdq
  Numa Nodes: 1
  Numa Nodes of Cores: 0->0 | 1->0 | 2->0 | 3->0 |
 Physical Memory: 15.51 GB
Transparent Huge Pages Config:
  enabled: [always] madvise never
  defrag: [always] madvise never
  khugepaged defrag: 1
 Disk Info:
  Num disks 1:
    vda (rotational=true)
```

Веб-интерфейс statestored

Главная страница веб-интерфейса statestored содержит следующую информацию о statestored:

- ***Version*** — версия демона и информация о том, когда была собрана Impala и какие флаги сборки использовались.
- ***Process start time*** — время запуска демона.
- ***Hardware info*** — информация об оборудовании хоста.
- ***OS information*** — информация об операционной системе, которая установлена на хосте.
- ***Process information*** — информация о процессе демона.
- ***CGroup information*** — информация о настройках CGroup.

log_level

На этой странице отображаются текущие уровни важности сообщений, попадающих в логи Java и серверной части демона. Также страница позволяет динамически изменять уровни логирования без перезапуска демона.

logs

Страница отображает последние записи из лог-файла *statestored.INFO*, включающего сообщения с уровнем `info`, `warning` и `error`. Позволяет просмотреть сведения о последних операциях, чтобы определить, были ли операции успешными или возникли ошибки.

Эта страница позволяет избежать поиска лог-файлов в файловой системе.

memz

На этой странице отображается сводная и подробная информация об использовании памяти демоном statestored.

metrics

На странице отображается текущий набор метрик, счетчиков и флагов, представляющих различные аспекты внутренних операций демона.

profile_docs

На странице перечислены счетчики оператора `PROFILE`.

rpcz

На странице отображается информация о продолжительности RPC-соединений демона с другими демонами Impala.

subscribers

На странице отображается информация о демонах Impala, зарегистрированных в statestored для получения и отправки обновлений.

threadz

Страница отображает информацию о потоках (threads), используемых демоном, и о категориях, в которые они сгруппированы.

topics

На странице отображается информация о разделах метаданных (topics), на которые подписаны другие демоны Impala для получения обновлений.

varz

Страница отображает параметры конфигурации, которые используются, когда этот демон взаимодействует с другими компонентами Hadoop, такими как HDFS и YARN. Параметры собираются из конфигурационных файлов.

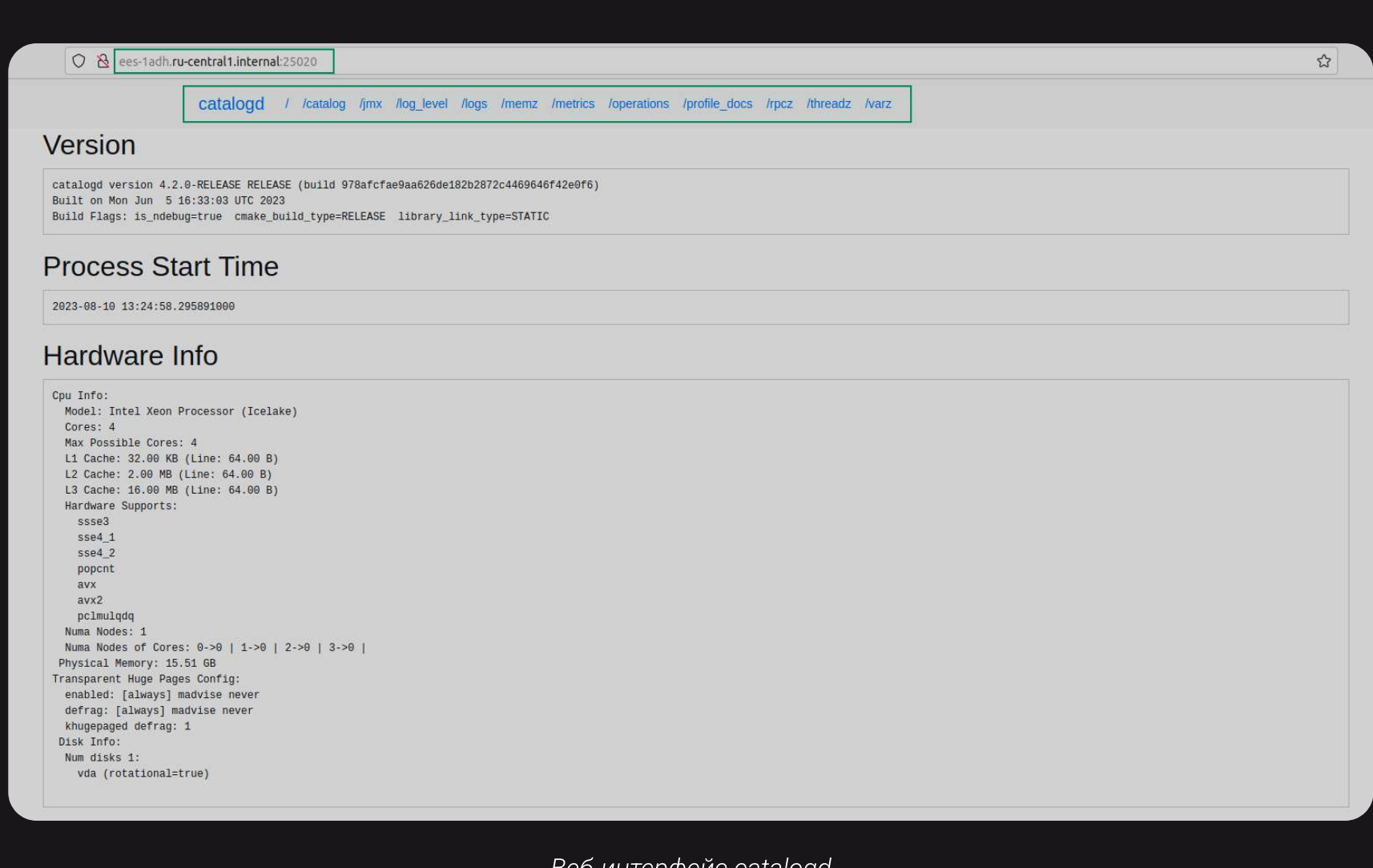
Prometheus metrics

На странице представлены метрики в формате Prometheus. Prometheus может использовать эти данные для мониторинга и оповещения.

Меню не содержит опции `/metrics_prometheus`. Чтобы отобразить эту страницу, добавьте `/metrics_prometheus` к HTTP-адресу в браузере.

Веб-интерфейс Impala Catalog Service (catalogd)

Веб-интерфейс catalogd включает информацию о базах данных, таблицах и других объектах, которыми управляет Impala, в дополнение к использованию ресурсов и параметрам конфигурации catalogd. Поскольку в кластере есть только одна нода с catalogd, вы можете получить доступ к веб-интерфейсу catalogd только на хосте с компонентом Impala Catalog Service.



Version

```
catalogd version 4.2.0-RELEASE RELEASE (build 978afcfae9aa626de182b2872c4469646f42e0f6)
Built on Mon Jun  5 16:33:03 UTC 2023
Build Flags: is_ndebug=true  cmake_build_type=RELEASE  library_link_type=STATIC
```

Process Start Time

```
2023-08-10 13:24:58.295891000
```

Hardware Info

```
Cpu Info:
  Model: Intel Xeon Processor (Icelake)
  Cores: 4
  Max Possible Cores: 4
  L1 Cache: 32.00 KB (Line: 64.00 B)
  L2 Cache: 2.00 MB (Line: 64.00 B)
  L3 Cache: 16.00 MB (Line: 64.00 B)
  Hardware Supports:
    ssse3
    sse4_1
    sse4_2
    popcnt
    avx
    avx2
    pclmulqdq
  Numa Nodes: 1
  Numa Nodes of Cores: 0->0 | 1->0 | 2->0 | 3->0 |
 Physical Memory: 15.51 GB
Transparent Huge Pages Config:
  enabled: [always] madvise never
  defrag: [always] madvise never
  khugepaged defrag: 1
 Disk Info:
  Num disks 1:
    vda (rotational=true)
```

Веб-интерфейс catalogd

Главная страница веб-интерфейса catalogd содержит следующую информацию о catalogd:

- ***Version*** — версия демона и информация о том, когда была собрана Impala и какие флаги сборки использовались.
- ***Process start time*** — время запуска демона.
- ***Hardware info*** — информация об оборудовании хоста.
- ***OS information*** — информация об операционной системе, которая установлена на хосте.
- ***Process information*** — информация о процессе демона.
- ***CGroup information*** — информация о настройках CGroup.

catalog

На странице ***Catalog*** отображается список баз данных и связанных с ними таблиц. Вы можете использовать эту страницу, чтобы определить, какая база данных включает определённую таблицу, проверить правильность написания имени базы данных или таблицы, найти одинаковые имена таблиц в нескольких базах данных. Основным сценарием использования этой страницы при отладке является проверка того, знает ли экземпляр impalad о таблице, которая находится в определенной базе данных.

Помимо данных, которые есть на странице ***catalog*** демона impalad, эта страница содержит списки таблиц с самыми высокими требованиями к памяти, с наибольшим количеством операций с метаданными, с наибольшим количеством файлов и с наибольшим временем загрузки метаданных.

jmx

На этой странице отображается информация мониторинга о различных подсистемах JVM в формате JSON, например, о пулах памяти, управлении потоками, времени выполнения и других.

log_level

На этой странице отображаются текущие уровни важности сообщений, попадающих в логи Java и серверной части демона. Также страница позволяет динамически изменять уровни логирования без перезапуска демона.

logs

Страница отображает последние записи из лог-файла *catalogd.INFO*, включающего сообщения с уровнем `info`, `warning` и `error`. Позволяет просмотреть сведения о последних операциях, чтобы определить, были ли операции успешными или возникли ошибки.

Эта страница позволяет избежать поиска лог-файлов в файловой системе.

memz

На этой странице отображается сводная и подробная информация об использовании памяти демоном catalogd.

metrics

На странице отображается текущий набор метрик, счетчиков и флагов, представляющих различные аспекты внутренних операций демона.

operations

На странице отображается информация об операциях catalogd.

profile_docs

На странице перечислены счетчики оператора `PROFILE`.

rpcz

На странице отображается информация о продолжительности RPC-соединений демона с другими демонами Impala.

threadz

Страница отображает информацию о потоках (threads), используемых демоном, и о категориях, в которые они сгруппированы.

varz

Страница отображает параметры конфигурации, которые используются, когда этот демон взаимодействует с другими компонентами Hadoop, такими как HDFS и YARN. Параметры собираются из конфигурационных файлов.

Prometheus metrics

На странице представлены метрики в формате Prometheus. Prometheus может использовать эти данные для мониторинга и оповещения.

Меню не содержит опции `/metrics_prometheus`. Чтобы отобразить эту страницу, добавьте `/metrics_prometheus` к HTTP-адресу в браузере.

Настройка производительности

Евгения Кузина

Содержание

Назад к содержанию

Операторы для настройки производительности
- EXPLAIN
- SUMMARY
- PROFILE

Страница queries веб-интерфейса impalad

В Impala есть операторы, с помощью которых можно получить информацию о производительности запросов. Impala также предоставляет веб-интерфейсы, которые отображают статистику производительности. Наиболее полезной является страница queries веб-интерфейса impalad. Она содержит планы запросов, отчеты `PROFILE` и `SUMMARY`, и другую информацию о производительности запросов.

Операторы для настройки производительности

В Impala есть следующие операторы, которые можно использовать для настройки производительности:

- EXPLAIN — отображает план выполнения запроса.
- SUMMARY — выводит сводный отчёт о последнем выполненном запросе.
- PROFILE — генерирует подробный низкоуровневый отчет о последнем выполненном запросе.

EXPLAIN

Оператор EXPLAIN отображает план выполнения, который Impala генерирует для указанного запроса. При вызове этой команды запрос не выполняется.

План включает механизмы низкого уровня, которые Impala будет использовать для чтения данных. Он описывает, как операции будут распределяться между нодами кластера и как промежуточные результаты будут объединяться для получения окончательного результата. Вы можете просмотреть эти сведения перед выполнением запроса, чтобы убедиться, что запрос будет выполнен эффективно. Если полученный план не соответствует вашим требованиям, вы можете изменить запрос, чтобы повысить производительность. Например, можно изменить условия в выражении `WHERE`, порядок таблиц в операторе `JOIN`, добавить подзапросы, изменить партиционирование таблицы, а также выполнить сбор статистики по столбцам и таблицам в Hive (см. COMPUTE STATS Statement).

Например, запрос `explain select count(*) from table1;` имеет следующий результат:

```
Query: explain select count(*) from table1
+------------------------------------------------------------+
| Explain String                                             |
+------------------------------------------------------------+
| Max Per-Host Resource Reservation: Memory=8.00KB Threads=2 |
| Per-Host Resource Estimates: Memory=10MB                   |
| Codegen disabled by planner                                |
|                                                            |
| PLAN-ROOT SINK                                             |
| |                                                          |
| 03:AGGREGATE [FINALIZE]                                    |
| |  output: sum(count(*))                                   |
| |                                                          |
| 02:EXCHANGE [UNPARTITIONED]                                |
| |                                                          |
| 01:AGGREGATE                                               |
| |  output: count(*)                                        |
| |                                                          |
| 00:SCAN HDFS [default.table1]                              |
|    HDFS partitions=1/1 files=5 size=3.99KB                 |
|    row-size=8B cardinality=5                               |
+------------------------------------------------------------+
```

Прочитаем план, созданный `EXPLAIN`, снизу вверх:

- Последняя часть (`00:SCAN HDFS [default.table1]`) плана показывает информацию низкого уровня, такую как ожидаемый объем данных, которые будут прочитаны. Это позволяет оценить эффективность стратегии партиционирования и время сканирования таблицы на основе общего размера данных.
- На следующем шаге вы увидите операции, которые будут распараллелены и выполнены на каждой ноде Impala.
- Следующие уровни показывают, как промежуточные наборы результатов объединяются и передаются от одного узла к другому.

Вы можете использовать опцию запроса EXPLAIN_LEVEL для управления уровнем детализации, предоставляемой в результате `EXPLAIN`. Допустимые значения `EXPLAIN_LEVEL:`

- `0` или `MINIMAL` — отображает список с одной строкой для каждой операции. В первую очередь это применяется для проверки порядка объединения в операторе `JOIN` в больших запросах, где обычный вывод `EXPLAIN` слишком длинный, чтобы его можно было легко прочитать.
- `1` или `STANDARD` — значение по умолчанию, показывающее план выполнения запроса на логическом уровне.
- `2` или `EXTENDED` — включает дополнительную информацию, иллюстрирующую, как планировщик запросов использует статистику в процессах принятия решений.
- `3` или `VERBOSE` — обеспечивает максимальный уровень детализации и показывает, как запрос разбивается на фрагменты, которые собираются в пайплайн для каждой ноды. Эта информация полезна для низкоуровневого тестирования производительности и настройки Impala, а не для оптимизации SQL-кода на уровне пользователя.

На всех уровнях детализации вывод `EXPLAIN` содержит предупреждение, если в каких-либо таблицах запроса отсутствует статистика. Используйте оператор COMPUTE STATS для сбора статистики для каждой таблицы и предотвращения появления этого предупреждения.

Результат `EXPLAIN` также печатается в начале отчета команды PROFILE, описанной ниже.

SUMMARY

Команда `SUMMARY` отображает время выполнения этапов запроса. Это позволяет определить потенциальные проблемы с производительностью. Вы можете запустить `SUMMARY` после выполнения запроса, чтобы увидеть фактические характеристики его производительности.

Пример:

```sql
SELECT AVG(salary) FROM table1 WHERE position like 'manager%';

summary;
```

Результат:

```
+--------------+--------+--------+----------+----------+-------+------------+----------+---------------+-----------------+
| Operator     | #Hosts | #Inst  | Avg Time | Max Time | #Rows | Est. #Rows | Peak Mem | Est. Peak Mem | Detail          |
+--------------+--------+--------+----------+----------+-------+------------+----------+---------------+-----------------+
| 03:AGGREGATE | 1      | 1      | 1.03ms   | 1.03ms   | 1     | 1          | 48.00 KB | -1 B          | MERGE FINALIZE  |
| 02:EXCHANGE  | 1      | 1      | 0ns      | 0ns      | 1     | 1          | 0 B      | -1 B          | UNPARTITIONED   |
| 01:AGGREGATE | 1      | 1      |30.79ms   | 30.79ms  | 1     | 1          | 80.00 KB | 10.00 MB      |                 |
| 00:SCAN HDFS | 1      | 1      | 5.45s    | 5.45s    | 2.21M | -1         | 64.05 MB | 432.00 MB     | default.table1  |
+--------------+--------+--------+----------+----------+-------+------------+----------+---------------+-----------------+
```

Различные этапы запроса и их время показаны с фактическими и расчетными значениями, используемыми в плане выполнения запроса `EXPLAIN`. Функция AVG() вычисляется для подмножества данных на каждой ноде (этап 01), а затем в конце объединяются агрегированные результаты со всех нод (этап 03). Вы можете увидеть, какие этапы заняли больше всего времени и существенно ли какие-либо оценки отличались от фактических.

В Impala есть параметр запроса MT_DOP, определяющий степень параллелизма внутри одной ноды, используемого для определенных операций, которые могут выиграть от многопоточного выполнения. Если `MT_DOP` установлен больше 0, столбец `#Inst` в выходных данных показывает количество экземпляров фрагмента. Impala разбивает каждый запрос на более мелкие блоки, распределенные по кластеру, и эти блоки называются фрагментами. Когда для параметра запроса `MT_DOP` установлено значение 0, столбец `#Inst` показывает то же значение, что и столбец `#Hosts`, поскольку для каждого хоста существует только один фрагмент.

Результат `SUMMARY` также печатается в начале отчета команды PROFILE, описанной ниже.

PROFILE

Команда `PROFILE` создает подробный низкоуровневый отчет, показывающий, как был выполнен последний запрос. Отчет содержит множество параметров. Вы можете найти фрагменты вывода `PROFILE` ниже.

```
Query Runtime Profile:
Query (id=36493a18ee92e275:1c04f39000000000):
   - InactiveTotalTime: 0.000ns
   - TotalTime: 0.000ns
  Summary:
    Session ID: ed440ba9e107f951:3ae2c27f67a7d6ae
    Session Type: HIVESERVER2
    HiveServer2 Protocol Version: V6
    Start Time: 2023-08-24 13:23:10.491153000
    End Time: 2023-08-24 13:23:10.604175000
    Query Type: QUERY
    Query State: FINISHED
    Impala Query State: FINISHED
    Query Status: OK
    Impala Version: impalad version 4.2.0-RELEASE RELEASE (build 978afcfae9aa626de182b2872c4469646f42e0f6)
    User: admin
    Connected User: admin
    Delegated User:
    Network Address: 10.92.6.52:47426
    Default Db: default
    Sql Statement: SELECT AVG(field1) FROM table1 WHERE field3 like 'row%'
    Coordinator: ees-1adh.ru-central1.internal:27000
    Query Options (set by configuration): EXPLAIN_LEVEL=VERBOSE,TIMEZONE=UTC,CLIENT_IDENTIFIER=Impala Shell v4.2.0-RELEASE
(978afcf) built on Mon Jun  5 16:33:03 UTC 2023,DEFAULT_FILE_FORMAT=PARQUET
    Query Options (set by configuration and planner):
EXPLAIN_LEVEL=VERBOSE,NUM_NODES=1,NUM_SCANNER_THREADS=1,RUNTIME_FILTER_MODE=OFF,MT_DOP=0,TIMEZONE=UTC,CLIENT_IDENTIFIER=Impala
Shell v4.2.0-RELEASE (978afcf) built on Mon Jun  5 16:33:03 UTC 2023,DEFAULT_FILE_FORMAT=PARQUET,SPOOL_QUERY_RESULTS=0
...

Estimated Per-Host Mem: 16793600
Request Pool: default-pool
Per Host Min Memory Reservation: ees-1adh.ru-central1.internal:27000(16.00 KB)
Per Host Number of Fragment Instances: ees-1adh.ru-central1.internal:27000(1)
Admission result: Admitted immediately
Cluster Memory Admitted: 16.02 MB
Executor Group: empty group (using coordinator only)
ExecSummary:
Operator       #Hosts  #Inst  Avg Time  Max Time  #Rows  Est. #Rows  Peak Mem  Est. Peak Mem  Detail
---------------------------------------------------------------------------------------------------------
F00:ROOT            1      1   0.000ns   0.000ns                            0              0
01:AGGREGATE        1      1   0.000ns   0.000ns      1           1  37.00 KB       16.00 KB  FINALIZE
00:SCAN HDFS        1      1   5.000ms   5.000ms     15           1  85.00 KB       16.00 MB  default.table1
    Errors:
    Query Compilation: 8.502ms
       - Metadata of all 1 tables cached: 1.585ms (1.585ms)
       - Analysis finished: 3.044ms (1.459ms)
       - Authorization finished (noop): 3.203ms (159.571us)
       - Value transfer graph computed: 3.362ms (158.312us)
       - Single node plan created: 6.038ms (2.676ms)
       - Distributed plan created: 6.121ms (83.217us)
       - Lineage info computed: 6.312ms (190.851us)
       - Planning finished: 8.502ms (2.189ms)
    Query Timeline: 113.002ms
       - Query submitted: 0.000ns (0.000ns)
       - Planning finished: 10.000ms (10.000ms)
       - Submit for admission: 10.000ms (0.000ns)
       - Completed admission: 10.000ms (0.000ns)
       - Ready to start on 1 backends: 10.000ms (0.000ns)
       - All 1 execution backends (1 fragment instances) started: 11.000ms (1.000ms)
       - Rows available: 16.000ms (5.000ms)
       - First row fetched: 112.002ms (96.001ms)
       - Last row fetched: 112.002ms (0.000ns)
       - Released admission control resources: 113.002ms (1.000ms)
       - Unregister query: 113.002ms (0.000ns)
     - AdmissionControlTimeSinceLastUpdate: 18.000ms
     - ComputeScanRangeAssignmentTimer: 0.000ns
     - InactiveTotalTime: 0.000ns
     - TotalTime: 0.000ns
...
```

Вывод показывает физические детали запроса (количество прочитанных байтов, максимальное использование памяти и т. д.) для каждой ноды. Можно использовать эту информацию, чтобы определить, существенно ли запрос зависит от операций ввода-вывода или работы процессора, приводит ли какое-либо состояние сети к проблемам, влияет ли замедление выполнения запроса только на некоторые ноды, или на все сразу.

Вы можете найти описание параметров `PROFILE` на странице ***/profile_docs*** в веб-интерфейсе impalad. Для доступа к этой странице используйте следующий шаблон адреса: `http://<server-hostname>:<port>/profile_docs`. Пример: `http://ees-1adh.ru-central1.internal:25000/profile_docs`.

Значения в выводе `PROFILE` отражают время, затраченное на операцию, в режиме реального времени. Для значений, которые представляют системное время или время пользователя, способ измерения отображается в имени метрики, например `ScannerThreadsSysTime` или `ScannerThreadsUserTime`. Например, многопоточная операция ввода-вывода может показывать небольшое значение в режиме реального времени, в то время как соответствующее системное время может быть больше. Оно представляет собой сумму времени процессора, затрачиваемого каждым потоком. Значение в режиме реального времени также может быть больше, поскольку оно может включать время ожидания, в то время как соответствующие значения системного и пользовательского времени измеряют только время, в течение которого операция активно использует ЦП.

Impala отображает результаты `EXPLAIN` и `SUMMARY` в начале отчета `PROFILE`, что позволяет изучить как логический, так и физический уровни выполнения запроса.

В результате команды `PROFILE` присутствует раздел `Per Node Profiles`. Он включает следующие показатели, которыми можно управлять с помощью параметра запроса RESOURCE_TRACE_RATIO:

- **CpuIoWaitPercentage** — процент времени, в течение которого ЦП (один или несколько) простаивал, и система имела ожидающие запросы дискового ввода-вывода.
- **CpuSysPercentage** — процент использования процессора системой.
- **CpuUserPercentage** — процент использования процессора пользователем.
- **HostDiskReadThroughput** — данные, считываемые хостом в рамках выполнения этого запроса, data-нодой HDFS и другими процессами, работающими в той же системе.
- **HostDiskWriteThroughput** — данные, записываемые хостом в рамках выполнения этого запроса, data-нодой HDFS и другими процессами, работающими в той же системе.
- **HostNetworkRx** — данные, полученные хостом в рамках выполнения этого запроса, других запросов и других процессов, выполняемых в той же системе.
- **HostNetworkTx** — данные, передаваемые хостом в рамках выполнения этого запроса, других запросов и других процессов, выполняемых в той же системе.

В настоящее время страница ***/profile_docs*** не содержит эти показатели.

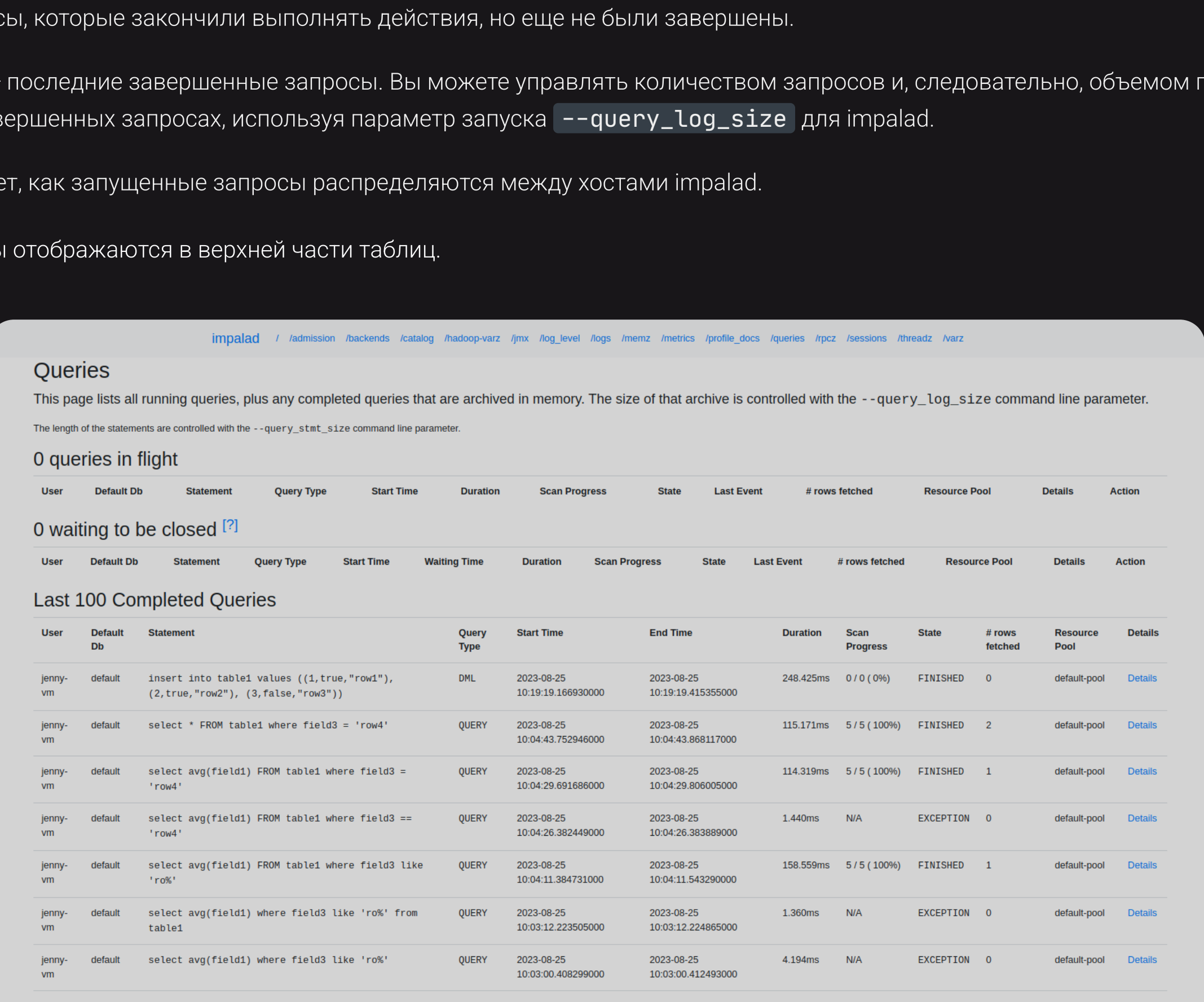
Страница queries веб-интерфейса impalad

Страница ***queries*** собирает статистику выполнения запросов. Для доступа к этой странице используйте следующий шаблон адреса: `http://<server-hostname>:<port>/queries`. Пример: `http://ees-1adh.ru-central1.internal:25000/queries`.

Страница ***queries*** группирует запросы в несколько таблиц:

- ***Queries in flight*** — выполняемые запросы.
- ***Waiting to be closed*** — запросы, которые закончили выполнять действия, но еще не были завершены.
- ***Last 100 completed queries*** — последние завершенные запросы. Вы можете управлять количеством запросов и, следовательно, объемом памяти, выделенной для хранения информации о завершенных запросах, используя параметр запуска `--query_log_size` для impalad.
- ***Query Locations*** — отображает, как запущенные запросы распределяются между хостами impalad.

Последние по времени запросы отображаются в верхней части таблиц.



Queries

This page lists all running queries, plus any completed queries that are archived in memory. The size of that archive is controlled with the `--query_log_size` command line parameter.

The length of the statements are controlled with the `--query_stmt_size` command line parameter.

0 queries in flight

User	Default Db	Statement	Query Type	Start Time	Duration	Scan Progress	State	Last Event	# rows fetched	Resource Pool	Details	Action

0 waiting to be closed [?]

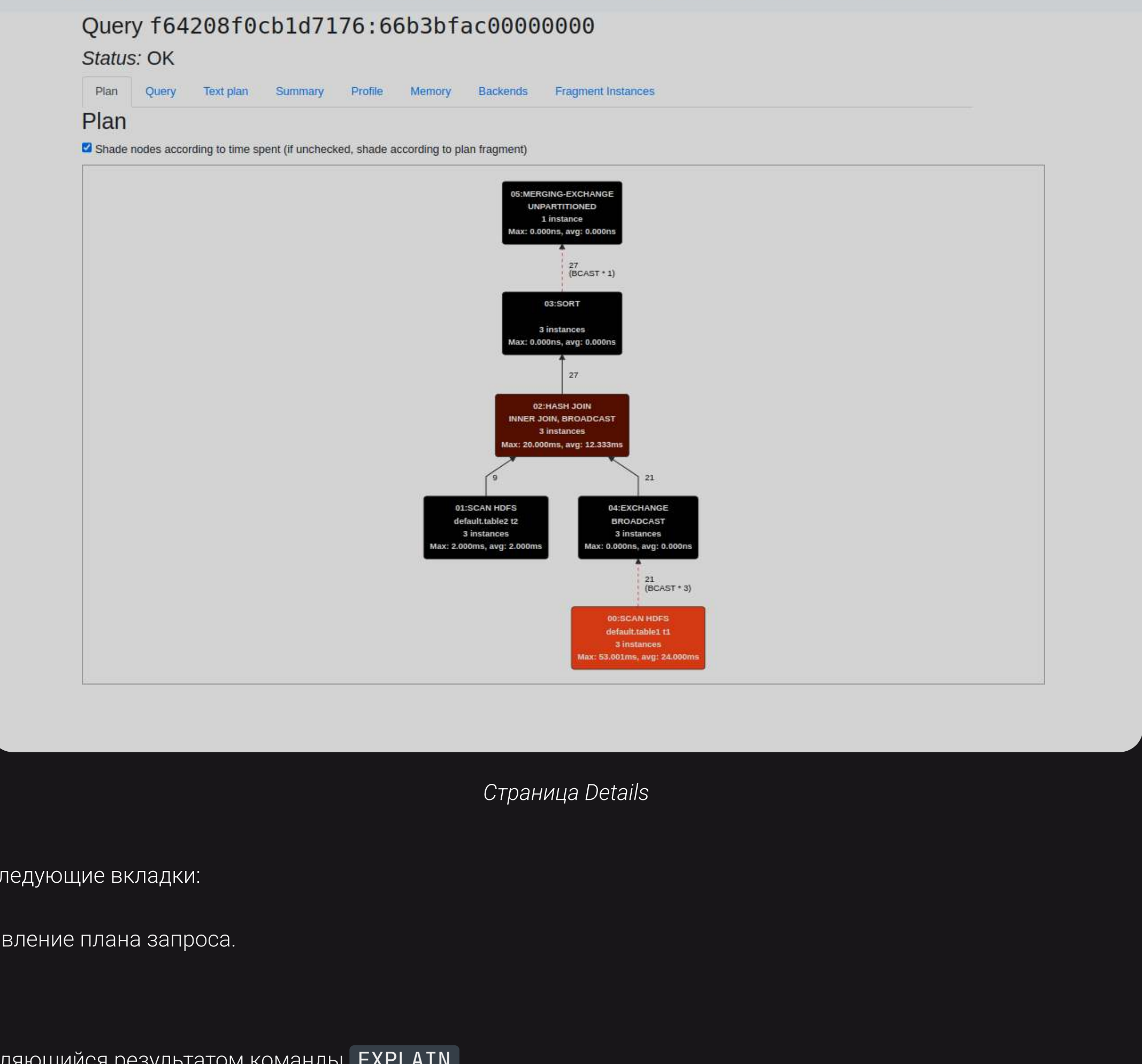
User	Default Db	Statement	Query Type	Start Time	Waiting Time	Duration	Scan Progress	State	Last Event	# rows fetched	Resource Pool	Details	Action

Last 100 Completed Queries

User	Default Db	Statement	Query Type	Start Time	End Time	Duration	Scan Progress	State	# rows fetched	Resource Pool	Details
jenny-vm	default	`insert into table1 values ((1,true,"row1"), (2,true,"row2"), (3,false,"row3"))`	DML	2023-08-25 10:19:19.166930000	2023-08-25 10:19:19.415355000	248.425ms	0 / 0 (0%)	FINISHED	0	default-pool	Details
jenny-vm	default	`select * FROM table1 where field3 = 'row4'`	QUERY	2023-08-25 10:04:43.752946000	2023-08-25 10:04:43.868117000	115.171ms	5 / 5 (100%)	FINISHED	2	default-pool	Details
jenny-vm	default	`select avg(field1) FROM table1 where field3 = 'row4'`	QUERY	2023-08-25 10:04:29.691686000	2023-08-25 10:04:29.806005000	114.319ms	5 / 5 (100%)	FINISHED	1	default-pool	Details
jenny-vm	default	`select avg(field1) FROM table1 where field3 == 'row4'`	QUERY	2023-08-25 10:04:26.382449000	2023-08-25 10:04:26.383889000	1.440ms	N/A	EXCEPTION	0	default-pool	Details
jenny-vm	default	`select avg(field1) FROM table1 where field3 like 'ro%'`	QUERY	2023-08-25 10:04:11.384731000	2023-08-25 10:04:11.543290000	158.559ms	5 / 5 (100%)	FINISHED	1	default-pool	Details
jenny-vm	default	`select avg(field1) where field3 like 'ro%' from table1`	QUERY	2023-08-25 10:03:12.223505000	2023-08-25 10:03:12.224865000	1.360ms	N/A	EXCEPTION	0	default-pool	Details
jenny-vm	default	`select avg(field1) where field3 like 'ro%'`	QUERY	2023-08-25 10:03:00.408299000	2023-08-25 10:03:00.412493000	4.194ms	N/A	EXCEPTION	0	default-pool	Details
jenny-vm	default	`select * from table1`	QUERY	2023-08-25 10:02:24.423594000	2023-08-25 10:02:24.535789000	112.195ms	5 / 5 (100%)	FINISHED	15	default-pool	Details

Страница queries

Ссылка ***Details*** для каждого запроса отображает подробную информацию о запросе, включая графическое представление плана.



Query `f64208f0cb1d7176:66b3bfac00000000`

Status: OK

Plan

☑ Shade nodes according to time spent (if unchecked, shade according to plan fragment)

Страница Details

На странице ***Details*** доступны следующие вкладки:

- ***Plan*** — графическое представление плана запроса.
- ***Query*** — текст запроса.
- ***Text plan*** — план запроса, являющийся результатом команды `EXPLAIN`.
- ***Summary*** — результат команды `SUMMARY`.
- ***Profile*** — результат команды `PROFILE`.
- ***Memory*** — потребление памяти. Доступно во время выполнения запроса.
- ***Backends*** — используемые экземпляры Impala. Доступно во время выполнения запроса.
- ***Fragment instances*** — информация об экземплярах фрагментов. Доступно во время выполнения запроса.

На вкладке ***Profile*** можно экспортировать результат в Thrift, JSON или текстовый формат.

Impala и Hive

Евгения Кузина

Содержание

Назад к содержанию

- Использование одной и той же таблицы в Hive и Impala
- Сбор статистики по таблицам и столбцам для оптимизации запросов
  - Статистика таблицы
  - Статистика столбцов

Impala может использовать существующую инфраструктуру Apache Hive для выполнения длительных, пакетных SQL-запросов. Impala хранит определения таблиц в базе данных MySQL или PostgreSQL, известной как Metastore, в той же базе данных, где Hive хранит данные такого типа. Таким образом, Impala может получить доступ к таблицам, определенным или загруженным через Hive, если во всех столбцах используются поддерживаемые Impala типы данных, форматы файлов и кодеки сжатия.

Использование одной и той же таблицы в Hive и Impala

Как описано выше, вы можете работать с одной и той же таблицей в Hive и Impala. Например, создадим таблицу в Hive (см. Работа с таблицами Hive). Командная строка Hive доступна на хостах, где установлен компонент *Hive HiveServer2*.

```
$ sudo -u hive hive
```

```
CREATE TABLE employees (
employee_id INT,
full_name STRING,
hours INT,
rate  INT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

INSERT INTO TABLE
    employees
VALUES
     (1, 'James Wilson', 8, 500),
    (2, 'Mary Burton', 6, 450),
    (3, 'Patricia Robinson', 7, 430);
```

После этого можно использовать impala-shell для выборки данных:

```
$ impala-shell
```

```
SELECT * FROM employees;
```

Результат:

```
Query: select * from employees
Query submitted at: 2023-09-05 14:32:38 (Coordinator: http://ees-1adh.ru-central1.internal:25000)
Query progress can be monitored at: http://ees-1adh.ru-central1.internal:25000/query_plan?
query_id=dd477956d9bde7ca:61358fec00000000
+-------------+-------------------+-------+------+
| employee_id | full_name         | hours | rate |
+-------------+-------------------+-------+------+
| 1           | James Wilson      | 8     | 500  |
| 2           | Mary Burton       | 6     | 450  |
| 3           | Patricia Robinson | 7     | 430  |
+-------------+-------------------+-------+------+
Fetched 3 row(s) in 0.21s
```

Добавим строки в таблицу в impala-shell:

```
insert into employees values ((4,"Robert Gray", 8, 480), (5,"Elizabeth Tucker", 7, 470));
```

Используйте командную строку Hive для отображения данных:

```
$ sudo -u hive hive
```

```
SELECT * FROM employees;
```

Результат:

```
OK
1   James Wilson    8   500
2   Mary Burton 6   450
3   Patricia Robinson   7   430
4   Robert Gray 8   480
5   Elizabeth Tucker    7   470
Time taken: 0.23 seconds, Fetched: 5 row(s)
```

Строки, добавленные в Impala, отобразились.

Первоначальный фокус на функциях запросов и производительности означает, что Impala может считывать больше типов данных с помощью оператора `SELECT`, чем записывать с помощью оператора `INSERT`. Чтобы запросить данные с использованием форматов файлов Avro, RCFile или SequenceFile, вы должны загрузить данные с помощью Hive.

Сбор статистики по таблицам и столбцам для оптимизации запросов

Impala может более эффективно оптимизировать сложные запросы или запросы к нескольким таблицам, если у нее есть статистика о размере данных и их распределении. Например, оптимизация `JOIN`-запросов требует статистики, чтобы определить, какая таблица больше. Это можно определить путем сравнения количества строк и среднего размера строк каждой таблицы.

Оптимизатор запросов Impala может использовать статистику таблиц и статистику столбцов. Первоначально эта информация собиралась с помощью оператора `ANALYZE TABLE` в Hive. В Impala вместо этого используйте оператор COMPUTE STATS. `COMPUTE STATS` требует меньше настроек, более надежен и не требует переключения между impala-shell и командной строкой Hive.

Статистика таблицы

Планировщик запросов Impala использует статистику, собранную для таблиц целиком и для их партиций. Эта информация включает в себя физические характеристики: количество строк, количество файлов данных, общий размер файлов данных и формат файла. Для партиционированных таблиц статистика собирается для каждой партиции и для всей таблицы.

Метаданные статистики хранятся в базе данных Metastore и могут обновляться с помощью Impala или Hive. Если значение недоступно, выводится `-1`. Некоторые значения, например количество и общий размер файлов данных, всегда остаются актуальными, поскольку их можно легко вычислить в процессе сбора метаданных HDFS.

Пример ниже использует оператор SHOW TABLE STATS, чтобы отобразить статистику таблицы. Для партиционированных таблиц также можно использовать оператор `SHOW PARTITIONS`, который отображает ту же информацию, что и `SHOW TABLE STATS`.

```
show table stats test_table1;
```

Результат:

```
+-------+--------+---------+--------------+-------------------+---------+-------------------+------------------------------------
--------+
| #Rows | #Files |  Size   | Bytes Cached | Cache Replication |  Format | Incremental stats | Location
|
+-------+--------+---------+--------------+-------------------+---------+-------------------+------------------------------------
--------+
| -1    |   52   | 12.62GB | NOT CACHED   | NOT CACHED        | PARQUET | false             |
hdfs://adh/apps/hive/warehouse/test_table1 |
```

Вывод отображает количество и размер файлов. Количество строк неизвестно и заменено `-1`, поскольку это требует потенциально затратного сканирования всей таблицы.

Выполним оператор `COMPUTE STATS` и проверим статистику таблицы:

```
compute stats test_table1;

show table stats test_table1;
```

Результат:

```
+-------------+--------+---------+--------------+-------------------+---------+-------------------+------------------------------
--------------+
|    #Rows    | #Files |  Size   | Bytes Cached | Cache Replication |  Format | Incremental stats | Location
|
+-------------+--------+---------+--------------+-------------------+---------+-------------------+------------------------------
--------------+
| 541666666   |   52   | 12.62GB | NOT CACHED   | NOT CACHED        | PARQUET | false             |
hdfs://adh/apps/hive/warehouse/test_table1 |
```

Оператор `COMPUTE STATS` добавляет недостающие значения статистики.

В выводе результата вы можете увидеть колонку `Incremental stats`. Значение в этой колонке показывает, используется ли `Incremental stats` для сбора статистики. Функциональность `Incremental stats` была добавлена для партиционированных таблиц. Когда Impala вычисляет инкрементную статистику для партиционированной таблицы, по умолчанию, обрабатываются только те партиции, которые еще не имеют инкрементной статистики. Таким образом, статистика поддерживается в актуальном состоянии, и повторная обработка всей таблицы не проводится. Для каждой таблицы необходимо использовать один тип вычисления статистики: или `COMPUTE STATS`, или `COMPUTE INCREMENTAL STATS`. Типы нельзя комбинировать или чередовать. Если необходимо переключиться с `COMPUTE STATS` на `COMPUTE INCREMENTAL STATS` или наоборот для существующей таблицы, удалите всю статистику, выполнив команду `DROP STATS`, перед тем как менять тип статистики. За дополнительной информацией обратитесь к разделу ***Generating Table and Column Statistics*** статьи Table and Column Statistics.

Статистика столбцов

Планировщик запросов Impala может использовать статистику по отдельным столбцам, если эти метаданные доступны в базе данных Metastore. Такая статистика наиболее полезна, когда нужно сравнить столбцы таблиц в запросах JOIN, чтобы оценить, сколько строк запрос получит из каждой таблицы. Эта статистика также важна для подзапросов, использующих операторы `EXISTS` и `IN`. Эти операторы обрабатываются Impala по тому же принципу, что и запросы `JOIN`.

Следующий пример отображает статистику столбцов для непартиционированной таблицы в формате Parquet:

```
show column stats test_table2;
```

Результат:

```
+--------+---------+------------------+--------+----------+----------+--------+---------+
| Column | Type    | #Distinct Values | #Nulls | Max Size | Avg Size | #Trues | #Falses |
+--------+---------+------------------+--------+----------+----------+--------+---------+
| field1 | INT     | -1               | -1     |  4       | 4.0      |  -1    |  -1     |
| field2 | BOOLEAN | -1               | -1     |  1       | 1.0      |  -1    |  -1     |
| field3 | STRING  | -1               | -1     | -1       | -1       |  -1    |  -1     |
+--------+---------+------------------+--------+----------+----------+--------+---------+
```

В выводе команды максимальные и средние размер значений числовых типов и типов фиксированного размера отображены, поскольку они являются константами. Количество `Distinct Values` не определено, так как для этого требуется потенциально дорогостоящее сканирование всей таблицы, и значения отображаются как `-1`. То же самое относится к максимальным и средним размерам типов переменного размера, таких как `STRING`. Выполнение оператора `COMPUTE STATS` добавляет недостающие значения и обновляет статистику.

```
compute stats test_table2;

show column stats test_table2;
```

Результат:

```
+--------+---------+------------------+--------+----------+----------+--------+---------+
| Column | Type    | #Distinct Values | #Nulls | Max Size | Avg Size | #Trues | #Falses |
+--------+---------+------------------+--------+----------+----------+--------+---------+
| field1 | INT     | 6                | 0      | 4        | 4.0      | -1     | -1      |
| field2 | BOOLEAN | 2                | 0      | 1        | 1.0      | 14     | 7       |
| field3 | STRING  | 6                | 0      | 4        | 4.0      | -1     | -1      |
+--------+---------+------------------+--------+----------+----------+--------+---------+
```

Значения `Trues` и `Falses` вычисляются только для полей типа `BOOLEAN`.

Для эффективного использования статистики по столбцам Impala также необходима статистика по таблицам. Когда вы используете оператор Impala `COMPUTE STATS`, статистика по таблице и всем столбцам собирается одновременно. Если вы запустите оператор Hive `ANALYZE TABLE COMPUTE STATISTICS FOR COLUMNS`, будет собрана статистика по столбцам. Impala может использовать статистику столбцов только в том случае, если таблица непартиционированная. Impala не может использовать статистику столбцов, созданную Hive, для партиционированной таблицы.

Impala и HBase

Евгения Кузина

Содержание

Назад к содержанию

- Пример использования таблицы HBase
- Обзор производительности
- Случаи использования
- Ограничения интеграции Impala и HBase

Impala позволяет выполнять SQL-запросы к таблицам HBase. С точки зрения пользователя Impala, имеющего опыт работы с РСУБД, HBase — это хранилище ключ/значение, где значение состоит из нескольких полей. Ключ сопоставляется с одним столбцом таблицы Impala, а остальные поля значения сопоставляются с другими столбцами таблицы Impala. Поскольку Impala и Hive используют одну и ту же базу данных Metastore, после создания таблицы в Hive вы можете выполнять запросы к ней через Impala.

Пример использования таблицы HBase

Пример ниже показывает, как работать с таблицами HBase. В этом примере используется таблица HBase `people_ages` из статьи Массовая загрузка с использованием встроенных задач MapReduce.

1. Чтобы использовать таблицу HBase в Impala, необходимо создать внешнюю таблицу в командной строке Hive.
   Укажите следующие опции:
   - Выражение `STORED BY` с обработчиком хранилища `org.apache.hadoop.hive.hbase.HBaseStorageHandler`.
   - Выражение `TBLPROPERTIES` для сопоставления созданной таблицы с таблицей, существующей в HBase. Свойство `hbase.mapred.output.outputtable` необходимо указать, если планируется добавлять данные в таблицу.
   - Столбец, соответствующий ключу строки HBase, должен иметь тип `STRING`.

   ```
   $ sudo -u hive hive
   ```

   ```
   CREATE EXTERNAL TABLE hbase_people_ages (name string, age int)
   STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
   WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,basic:age")
   TBLPROPERTIES ("hbase.table.name" = "people_ages", "hbase.mapred.output.outputtable" = "people_ages");
   ```

2. После создания новой таблицы через Hive выполните оператор `INVALIDATE METADATA` в impala-shell, чтобы Impala получила информацию о новой таблице.

```
$ impala-shell
```

```
INVALIDATE METADATA hbase_people_ages;
```

3. Вставьте новую строку в таблицу:

```
INSERT INTO hbase_people_ages VALUES ('Gary Morgan', 35), ('Jim Brian', 48);
```

4. Выполните выборку данных:

```
SELECT * from hbase_people_ages WHERE name LIKE 'Gar%';
```

   Результат:

```
+-------------------+-----+
| name              | age |
+-------------------+-----+
| Garcia Nathaniel | 61  |
| Gardner Helena   | 54  |
| Garner Andre     | 37  |
| Garner Juan      | 28  |
| Garner Mark      | 47  |
| Garrett Isabelle | 52  |
| Garrett Steven   | 24  |
| Garrett Willie   | 42  |
| Gary Morgan      | 35  |
| Garza Manuel     | 30  |
| Garza Paul       | 51  |
+-------------------+-----+
Fetched 11 row(s) in 5.74s
```

Для эффективных запросов используйте выражение `WHERE`, чтобы найти одно значение ключа или диапазон значений ключа. Старайтесь избегать запросов, выполняющих полное сканирование таблицы. Это эффективно для обычных таблиц Impala, но неэффективно в HBase.

ПРИМЕЧАНИЕ

Синтаксис `INSERT … VALUES` Impala подходит для таблиц HBase, поскольку вставка одной строки является эффективной операцией для таблицы HBase. Когда используется синтаксис `INSERT … SELECT`, HBase сохраняет только последнюю версию строки для каждого уникального ключа. Если оператор `INSERT … SELECT` копирует несколько строк, содержащих одно и то же значение ключа, запрос возвращает только одну строку с каждым значением ключевого столбца. Хотя в Impala нет оператора `UPDATE`, вы можете получить тот же результат, выполнив оператор `INSERT` с тем же значением ключа.

Обзор производительности

Impala использует клиентский API HBase через собственный интерфейс Java (JNI) для запроса данных, хранящихся в HBase. Эта операция не читает HFiles напрямую. Дополнительная нагрузка соединения делает важным выбор того, какие данные хранить в HBase. Также необходимо создавать запросы, которые могут эффективно извлекать данные HBase. Для этого необходимо учитывать следующие рекомендации:

- Используйте таблицу HBase для запросов, возвращающих одну строку или небольшой диапазон строк, а не для запросов, выполняющих полное сканирование всей таблицы. Если запрос ссылается на таблицу HBase и не содержит выражение `WHERE`, это явный признак неэффективности запроса.
- HBase может обеспечить приемлемую производительность для хранения небольших многомерных таблиц. Таблица должна быть достаточно небольшой, чтобы полное сканирование таблицы для каждого запроса было эффективным. HDFS также подходит для многомерных таблиц, которым не требуется поддержка запросов на обновление, удаление или вставку небольшого количества строк.
- Предикаты запроса применяются к ключам строк в качестве начального и конечного значения, тем самым ограничивая область конкретного поиска. Если поле ключа строки не сопоставлено со столбцом типа `STRING`, операции сравнения не работают.
- Если предикаты используются для неключевых столбцов, Impala возвращает больше строк, чем HBase, который может использовать фильтр `SingleColumnValueFilter`. Когда предикат ключа строки соответствует одной строке, HBase находит и возвращает эту строку. Если используется предикат для неключевого поля и ему соответствует только одна строка, HBase сканирует всю таблицу, чтобы найти эту строку.
- По соображениям производительности важно объявить ключевой столбец как `STRING` в операторе `CREATE EXTERNAL TABLE`. Вы также можете определить другие столбцы с типами `INT` или `BOOLEAN` как `STRING`. В этом случае Impala сможет более эффективно оптимизировать работу с этими данными. За дополнительной информацией обратитесь к разделу ***Interpreting EXPLAIN Output for HBase Queries*** статьи Using Impala to Query HBase Tables.

Случаи использования

Ниже приведены типичные случаи использования Impala для выполнения запросов к таблицам HBase:

- Хранение быстро увеличивающихся счетчиков. Например, количество просмотров веб-страницы, количество пользовательских подключений или голосов за публикацию в социальных сетях. HBase эффективен для сбора данных такого типа: механизм хранения только для добавления эффективен для записи каждого изменения на диск, а запрос всегда возвращает самое последнее значение.
- Хранение таблиц с большим количеством столбцов (wide tables). Такие таблицы включают множество столбцов (тысячи), которые содержат характеристики сущности, например, пользователя онлайн-сервиса. Очень часто в этих таблицах большинство значений столбцов `NULL`, `0`, `FALSE`, пустые строки, другие пустые значения или значения-заполнители. Например, пользователь может никогда не использовать некоторые возможности сайта, не заполнить специальное поле в профиле или не посещать определенную часть сайта. Типичный запрос к таблице этого типа — получение одной строки со всей информацией об объекте, а не вычисление суммы, среднего значения или фильтрация миллионов строк, как в таблицах Impala.

Ограничения интеграции Impala и HBase

Интеграция Impala с HBase имеет следующие ограничения:

- Если вы выполняете команду `DROP TABLE` для таблицы, управляемой Impala, которая сопоставлена с таблицей HBase, базовая таблица в HBase не удаляется.
- Оператор `INSERT OVERWRITE` недоступен для таблиц HBase. Чтобы добавить новые данные или изменить существующую строку, вставьте новую строку. Для изменения существующей строки используйте то же значение ключа.
- Если вы выполняете оператор `CREATE TABLE LIKE` для таблицы, сопоставленной с таблицей HBase, новая таблица также получает имя базовой таблицы HBase. Таким образом, новая таблица является псевдонимом старой, а не новой таблицей с идентичной структурой столбцов.
- Копирование данных в таблицу HBase с использованием синтаксиса `INSERT … SELECT` Impala может привести к появлению меньшего количества новых строк, чем в наборе результатов запроса `SELECT`. Если набор результатов содержит несколько строк с одинаковым значением для ключевого столбца, каждая строка перезаписывает предыдущую с тем же значением ключа. Поскольку порядок вставленных строк непредсказуем, не следует полагаться на этот метод для сохранения последней версии определенного значения ключа.
- Поскольку сложные типы данных (`ARRAY`, `STRUCT` и `MAP`) в настоящее время поддерживаются только в таблицах Parquet, нельзя использовать эти типы в таблицах HBase, обрабатываемых через Impala.
- Оператор `LOAD DATA` нельзя использовать с таблицами HBase.
- Выражение `TABLESAMPLE` оператора `SELECT` не может быть применено к таблицам HBase.

Аутентификация LDAP в Impala

Сергей Тихомиров

Содержание

Назад к содержанию

Активация LDAP

Проверка

Impala позволяет пользователям проходить аутентификацию в impala-shell с помощью LDAP для подключения к Impala Daemon (impalad).

Активация LDAP

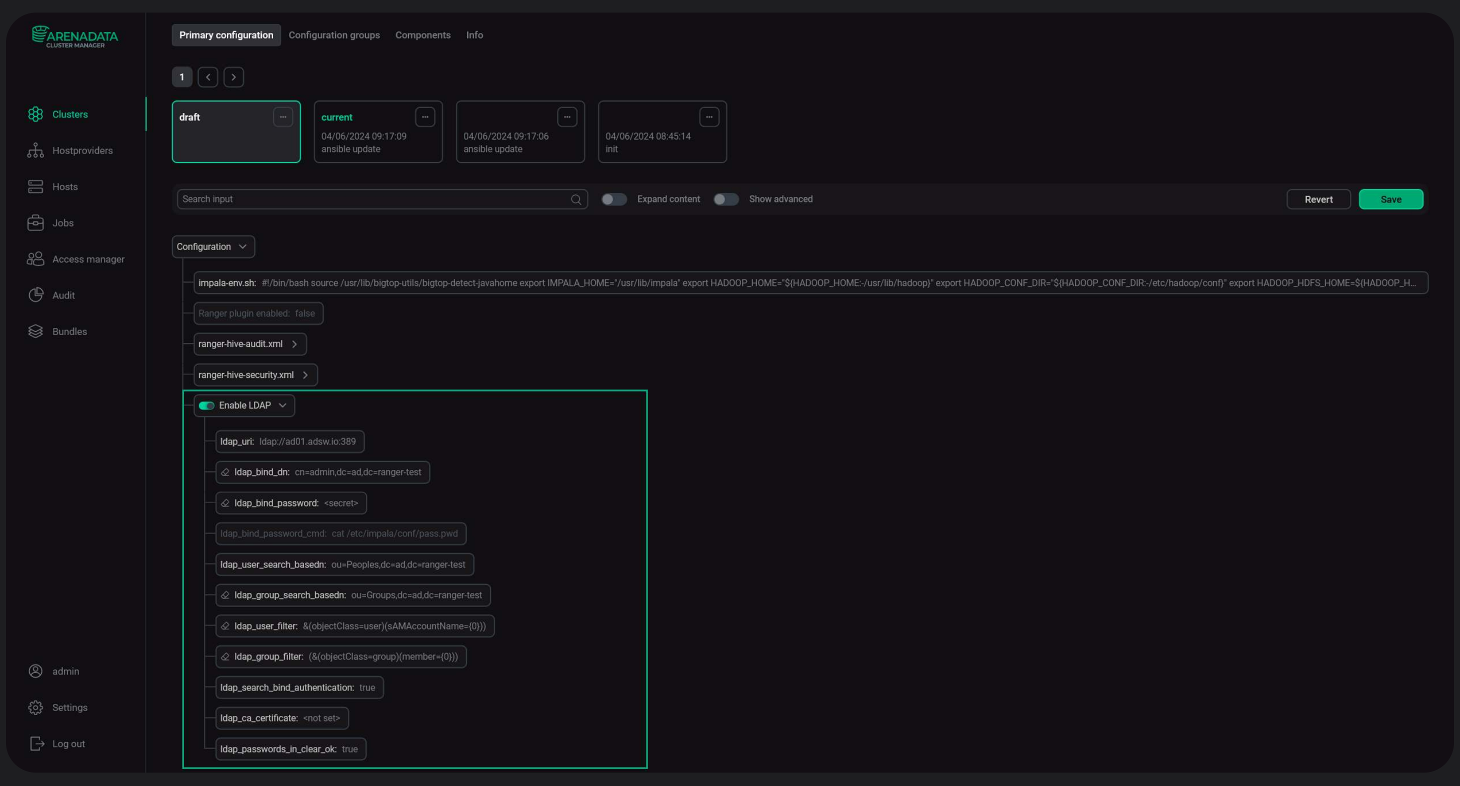
Для активации LDAP-аутентификации следуйте шагам ниже:

1. На странице ***Clusters*** выберите ваш кластер ADH и перейдите на вкладку ***Services*** в меню кластера.
2. Выберите сервис ***Impala*** и активируйте группу параметров ***Enable LDAP***.
3. Заполните параметры LDAP для Active Directory или FreeIPA и перезапустите Impala.

ВАЖНО

Параметры ***ldap_baseDN***, ***ldap_domain*** и ***ldap_bind_pattern*** являются взаимоисключающими.

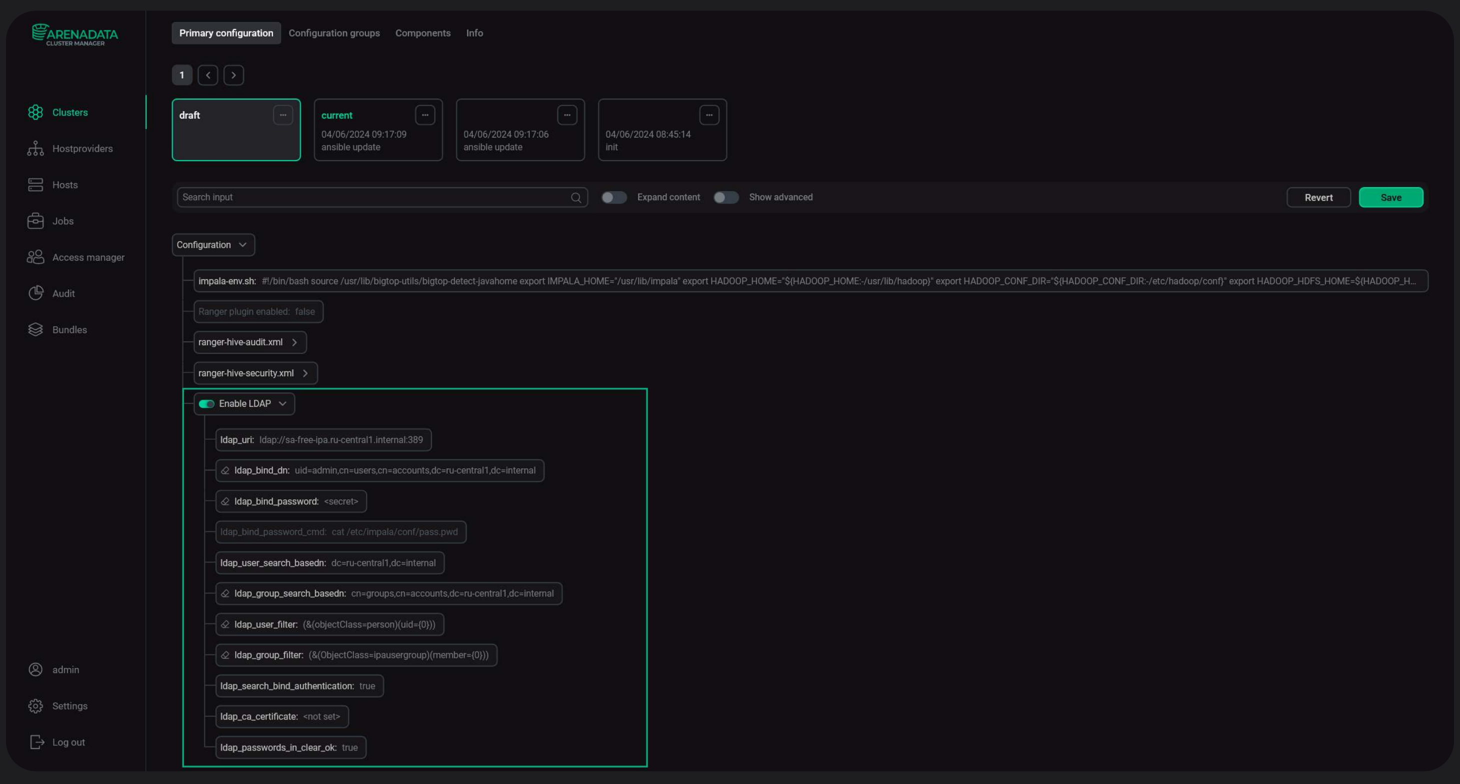
Active Directory



Enable LDAP
ldap_uri: ldap://ad01.adsw.io:389
ldap_bind_dn: cn=admin,dc=ad,dc=ranger-test
ldap_bind_password: <secret>
ldap_bind_password_cmd: cat /etc/impala/conf/pass.pwd
ldap_user_search_basedn: ou=Peoples,dc=ad,dc=ranger-test
ldap_group_search_basedn: ou=Groups,dc=ad,dc=ranger-test
ldap_user_filter: &(objectClass=user)(sAMAccountName={0}))
ldap_group_filter: (&(objectClass=group)(member={0}))
ldap_search_bind_authentication: true
ldap_ca_certificate: <not set>
ldap_passwords_in_clear_ok: true

Конфигурация LDAP с AD

FreeIPA



Enable LDAP
ldap_uri: ldap://sa-free-ipa.ru-central1.internal:389
ldap_bind_dn: uid=admin,cn=users,cn=accounts,dc=ru-central1,dc=internal
ldap_bind_password: <secret>
ldap_bind_password_cmd: cat /etc/impala/conf/pass.pwd
ldap_user_search_basedn: dc=ru-central1,dc=internal
ldap_group_search_basedn: cn=groups,cn=accounts,dc=ru-central1,dc=internal
ldap_user_filter: (&(objectClass=person)(uid={0}))
ldap_group_filter: (&(ObjectClass=ipausergroup)(member={0}))
ldap_search_bind_authentication: true
ldap_ca_certificate: <not set>
ldap_passwords_in_clear_ok: true

Конфигурация LDAP с FreeIPA

Проверка

Для запуска impala-shell требуется конфигурационный файл или параметры, переданные в командной строке. Чтобы не передавать вручную, поля `user` и `ldap_password_cmd` можно добавить в файл конфигурации:

```
[impala]
use_ldap=true
auth_creds_ok_in_clear=true
user=<username>
ldap_password_cmd=<pwd_print_command>
```

где:

- `<username>` — короткое имя пользователя;
- `<pwd_print_command>` — команда вывода пароля. Например, если вы хотите хранить пароль в *impalarc*, используйте `ldap_password_cmd=echo -n myPassword`. Если же вы храните (или планируете хранить) пароль в файле, то используйте `ldap_password_cmd=cat /etc/impala/conf/user-pass.pwd`, при этом расширение файла не имеет значения.

Помимо "ручного" способа, вы также можете изменить файлы конфигурации в ADCM:

1. В меню сервиса ***Impala*** перейдите на вкладку ***Components*** и нажмите ***Impala Client***.
2. Измените файлы ***impala-shell-env.sh*** и ***impalarc***. Первый отвечает за установку нужных системных переменных, а во втором хранятся настройки.

ПРИМЕЧАНИЕ

По умолчанию бандл не сохраняет пользовательский пароль, указанный в *impalarc*. Решение о том, хранить ли пользовательский пароль, и если хранить, то где, принимает администратор. Бандл сохраняет в файл только пароль администратора для поиска и аутентификации LDAP.

Чтобы применить параметры из конфигурационного файла, используйте для запуска команду ниже:

```
$ impala-shell --config_file=<path_to_config_file>
```

Необязательно указывать конфигурационный файл в команде, так как по умолчанию используется файл, указанный в *impala-shell-env.sh*. Чтобы использовать кастомный конфигурационный файл, измените параметр ***impala-shell-env.sh*** в ADCM, присвоив переменной `IMPALA_SHELL_GLOBAL_CONFIG_FILE` значение пути к вашему файлу. О запуске impala-shell с определенной конфигурацией читайте в статье Использование impala-shell для подключения к Impala.

Работа с таблицами Iceberg в Impala

Константин Алпашкин

Содержание

Назад к содержанию

- Каталоги Iceberg
- Команды DDL
  - Создание таблицы
  - CREATE TABLE AS SELECT
  - Удаление таблицы
  - Таблицы Iceberg V2
  - Эволюция схемы
  - Создание таблицы с партициями
- Команды DML
  - Запись данных
  - Удаление данных
  - Обновление данных
- Time travel
- Roll back
- Конвертация типов данных Iceberg

Apache Iceberg — это открытый высокопроизводительный формат для больших аналитических таблиц. Сервис ADH Impala поддерживает данный формат, позволяя работать с таблицами Iceberg с помощью SQL и выполнять аналитические операции.

Каталоги Iceberg

В архитектуре Iceberg каталог (catalog) — это именованное хранилище таблиц и их метаданных. Для работы с Iceberg-таблицами в Impala доступны следующие типы каталогов:

- **HiveCatalog**. Является дефолтным каталогом в Impala. Использует Hive Metastore для хранения метаданных таблиц. Данный каталог используется во всех примерах, приведенных в статье.
- **HadoopCatalog**. Каталог с указанием пути, предполагающий наличие в HDFS директории, где хранятся отдельные таблицы Iceberg.
- **HadoopTables**. Каталог, предполагающий наличие HDFS-директории, где хранятся данные нескольких таблиц Iceberg.

Также можно использовать кастомные каталоги для существующих таблиц. Однако автоматическое обновление метаданных не работает для таблиц в кастомном каталоге. Необходимо вручную вызывать `REFRESH` для таких таблиц, когда они изменяются вне Impala.

Команды DDL

Создание таблицы

Чтобы создать таблицу Iceberg, добавьте предложение `STORED AS ICEBERG` к стандартному оператору Impala `CREATE TABLE`.

Пример:

```
CREATE TABLE default.impala_ice_test (
    txn_id int,
    acc_id int,
    txn_value double,
    txn_date date)
STORED AS ICEBERG;
```

Выполните команду `DESCRIBE FORMATTED` для получения информации о только что созданной таблице:

```
DESCRIBE FORMATTED default.impala_ice_test;
```

Пример вывода

```
Query: describe formatted impala_ice_test
+------------------------------+-----------------------------------------------------+----------------------------------------
------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------
------------------------------------+
| name                         | type                                                 | comment
|
+------------------------------+-----------------------------------------------------+----------------------------------------
------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------
------------------------------------+
| # col_name                   | data_type                                            | comment
|
|                              | NULL                                                 | NULL
|
| txn_id                       | int                                                  | NULL
|
| acc_id                       | int                                                  | NULL
|
| txn_value                    | double                                               | NULL
|
| txn_date                     | date                                                 | NULL
|
|                              | NULL                                                 | NULL
|
| # Detailed Table Information | NULL                                                 | NULL
|
| Database:                    | default                                              | NULL
|
| OwnerType:                   | USER                                                 | NULL
|
| Owner:                       | admin                                                | NULL
|
| CreateTime:                  | Thu Jul 04 10:50:39 UTC 2024                         | NULL
|
| LastAccessTime:              | Sun Jan 25 08:15:21 UTC 1970                         | NULL
|
| Retention:                   | 2147483647                                           | NULL
|
| Location:                    | hdfs://adh/apps/hive/warehouse/impala_ice_test       | NULL
|
| Erasure Coding Policy:       | NONE                                                 | NULL
|
| Table Type:                  | EXTERNAL_TABLE                                       | NULL
|
| Table Parameters:            | NULL                                                 | NULL
|
|                              | EXTERNAL                                             | TRUE
|
|                              | OBJCAPABILITIES                                      | EXTREAD,EXTWRITE
|
|                              | accessType                                           | 8
|
|                              | current-schema                                       | {\"type\":\"struct\",\"schema-
id\":0,\"fields\":[{\"id\":1,\"name\":\"txn_id\",\"required\":false,\"type\":\"int\"},
{\"id\":2,\"name\":\"acc_id\",\"required\":false,\"type\":\"int\"},
{\"id\":3,\"name\":\"txn_value\",\"required\":false,\"type\":\"double\"},
{\"id\":4,\"name\":\"txn_date\",\"required\":false,\"type\":\"date\"}]} |
|                              | engine.hive.enabled                                 | true
|
|                              | external.table.purge                                | TRUE
|
|                              | metadata_location                                    |
hdfs://adh/apps/hive/warehouse/impala_ice_test/metadata/00000-c3406504-fe59-4464-8b1f-01971fc3b949.metadata.json
|
|                              | numFiles                                             | 1
|
|                              | snapshot-count                                       | 0
|
|                              | storage_handler                                      | org.apache.iceberg.mr.hive.
HiveIcebergStorageHandler
|
|                              | table_type                                           | ICEBERG
|
|                              | totalSize                                            | 1914
|
|                              | transient_lastDdlTime                                | 1720090239
|
|                              | uuid                                                 | 5495460c-53fe-4fb7-9120-0d5dad7fd022
|
|                              | write.format.default                                | parquet
|
|                              | NULL                                                 | NULL
|
| # Storage Information        | NULL                                                 | NULL
|
| SerDe Library:               | org.apache.iceberg.mr.hive.HiveIcebergSerDe        | NULL
|
| InputFormat:                 | org.apache.iceberg.mr.hive.HiveIcebergInputFormat  | NULL
|
| OutputFormat:                | org.apache.iceberg.mr.hive.HiveIcebergOutputFormat | NULL
|
| Compressed:                  | No                                                   | NULL
|
| Sort Columns:                | []                                                   | NULL
|
|                              | NULL                                                 | NULL
|
| # Constraints                | NULL                                                 | NULL
|
+------------------------------+-----------------------------------------------------+----------------------------------------
------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------
------------------------------------+
```

Параметр `table_type` в выводе указывает на то, что таблица была создана как Iceberg-таблица.

По умолчанию Impala создает таблицы Iceberg, используя формат данных Parquet. Чтобы использовать ORC или AVRO, установите свойство таблицы `write.format.default`, как показано ниже:

```
CREATE TABLE default.impala_ice_test_orc (
    txn_id int,
    acc_id int,
    txn_value double,
    txn_date date)
STORED AS ICEBERG
TBLPROPERTIES('write.format.default'='ORC');
```

РЕКОМЕНДАЦИЯ

Больше информации о свойствах таблиц Iceberg можно найти в документации Impala.

CREATE TABLE AS SELECT

Вы можете использовать синтаксис `CREATE TABLE AS <tbl_name>` (CTAS) для создания таблицы Iceberg на основе уже существующей таблицы.

Например:

```
CREATE TABLE default.impala_ice_test_ctas
STORED AS ICEBERG
AS SELECT txn_value, txn_date FROM default.impala_ice_test;
```

Пример вывода DESCRIBE FORMATTED

```
DESCRIBE FORMATTED default.impala_ice_test_ctas;
+------------------------------+------------------------------------------------------+---------------------------------------
------------------------------------------------------------------------------------------------------------------------------
----------------------------+
| name                         | type                                                  | comment
|
+------------------------------+------------------------------------------------------+---------------------------------------
------------------------------------------------------------------------------------------------------------------------------
----------------------------+
| # col_name                   | data_type                                             | comment
|
|                              | NULL                                                  | NULL
|
| txn_value                    | double                                                | NULL
|
| txn_date                     | date                                                  | NULL
|
|                              | NULL                                                  | NULL
|
| # Detailed Table Information | NULL                                                  | NULL
|
| Database:                    | default                                               | NULL
|
| OwnerType:                   | USER                                                  | NULL
|
| Owner:                       | admin                                                 | NULL
|
| CreateTime:                  | Thu Jul 04 11:07:31 UTC 2024                          | NULL
|
| LastAccessTime:              | Sun Jan 25 08:32:12 UTC 1970                          | NULL
|
| Retention:                   | 2147483647                                            | NULL
|
| Location:                    | hdfs://adh/apps/hive/warehouse/impala_ice_test_ctas | NULL
|
| Erasure Coding Policy:       | NONE                                                  | NULL
|
| Table Type:                  | EXTERNAL_TABLE                                        | NULL
|
| Table Parameters:            | NULL                                                  | NULL
|
|                              | EXTERNAL                                              | TRUE
|
|                              | OBJCAPABILITIES                                       | EXTREAD,EXTWRITE
|
|                              | accessType                                            | 8
|
|                              | current-schema                                        | {\"type\":\"struct\",\"schema-
id\":0,\"fields\":[{\"id\":1,\"name\":\"txn_value\",\"required\":false,\"type\":\"double\"},
{\"id\":2,\"name\":\"txn_date\",\"required\":false,\"type\":\"date\"}]} |
|                              | engine.hive.enabled                                  | true
|
|                              | external.table.purge                                 | TRUE
|
|                              | metadata_location                                     |
hdfs://adh/apps/hive/warehouse/impala_ice_test_ctas/metadata/00000-c3871ac4-ca86-4c6d-be6c-116f3bf7cdf9.metadata.json
|
|                              | numFiles                                              | 1
|
|                              | snapshot-count                                        | 0
|
|                              | storage_handler                                       | org.apache.iceberg.mr.hive.
HiveIcebergStorageHandler
|
|                              | table_type                                            | ICEBERG
|
|                              | totalSize                                             | 1531
|
|                              | transient_lastDdlTime                                 | 1720091251
|
|                              | uuid                                                  | 5b792d0b-4e67-4a2b-957c-86e41970c977
|
|                              | write.format.default                                 | parquet
|
|                              | NULL                                                  | NULL
|
| # Storage Information        | NULL                                                  | NULL
|
| SerDe Library:               | org.apache.iceberg.mr.hive.HiveIcebergSerDe         | NULL
|
| InputFormat:                 | org.apache.iceberg.mr.hive.HiveIcebergInputFormat   | NULL
|
| OutputFormat:                | org.apache.iceberg.mr.hive.HiveIcebergOutputFormat  | NULL
|
| Compressed:                  | No                                                    | NULL
|
| Sort Columns:                | []                                                    | NULL
|
|                              | NULL                                                  | NULL
|
| # Constraints                | NULL                                                  | NULL
|
+------------------------------+------------------------------------------------------+---------------------------------------
------------------------------------------------------------------------------------------------------------------------------
----------------------------+
```

Удаление таблицы

Вы можете удалить таблицу с помощью команды `DROP TABLE <tbl_name>`. Пример:

```
DROP TABLE default.impala_ice_test;
```

Если установлено свойство таблицы `external.table.purge=true`, команда также удаляет файлы данных из HDFS. По умолчанию свойство `external.table.purge` равно `true`, если таблица была создана с помощью команды `CREATE TABLE` в Impala. Если таблица создана с помощью `CREATE EXTERNAL TABLE` (таблица уже существует и только регистрируется в каталоге), данное свойство устанавливается как `false`, поэтому `DROP TABLE` не удаляет данные таблицы, а лишь удаляет таблицу из каталога.

Таблицы Iceberg V2

Формат таблиц Iceberg V2 поддерживает обновление на уровне строк для операций `DELETE` и `UPDATE`, используя подход "merge-on-read". При использовании этого формата, вместо того чтобы переписывать существующие файлы данных, Iceberg создает файлы удаления (delete files), которые хранят информацию об удаленных записях. Эти файлы содержат путь к файлу и позицию удаленных строк (position deletes).

Чтобы создать таблицу Iceberg, использующую формат Iceberg V2, укажите свойство таблицы `format-version=2` при создании таблицы.

Пример:

```
CREATE TABLE default.impala_ice_test_v2 (
    txn_id int,
    acc_id int,
    txn_value double,
    txn_date date)
STORED AS ICEBERG
TBLPROPERTIES('format-version'='2');
```

Вы также можете обновить существующую таблицу Iceberg V1 до версии Iceberg V2 с помощью команды `ALTER TABLE`. Пример показан ниже.

```
ALTER TABLE default.impala_ice_test
SET TBLPROPERTIES('format-version'='2');
```

Эволюция схемы

Iceberg поддерживает эволюцию схемы, что позволяет переупорядочивать, удалять или изменять столбцы без ущерба для читаемости старых файлов данных (до изменения схемы).

ПРИМЕЧАНИЕ

Impala не поддерживает эволюцию схемы для таблиц в формате AVRO. Дополнительная информация доступна в документации Iceberg.

Ниже приведены команды изменения схемы Iceberg-таблиц, поддерживаемые в Impala.

ALTER TABLE … RENAME TO

Следующая команда переименовывает таблицу Iceberg.

```
ALTER TABLE default.impala_ice_test
RENAME TO default.impala_ice_test_new;
```

ALTER TABLE … ADD COLUMN

Вы можете добавить столбец в существующую таблицу Iceberg, как показано ниже.

```
ALTER TABLE default.impala_ice_test
ADD COLUMN `comment` string;
```

Результат выполнения команды:

```
+----------------------------+
| summary                    |
+----------------------------+
| Column(s) have been added. |
+----------------------------+
```

ALTER TABLE … DROP COLUMN

Вы можете удалить столбец из существующей таблицы Iceberg, как показано ниже.

```
ALTER TABLE default.impala_ice_test
DROP COLUMN `comment`;
```

Результат выполнения команды:

```
+--------------------------+
| summary                  |
+--------------------------+
| Column has been dropped. |
+--------------------------+
```

ALTER TABLE … CHANGE COLUMN

Вы можете изменить имя столбца и его тип, если новый тип совместим со старым.

```
ALTER TABLE default.impala_ice_test
CHANGE txn_id txn_id BIGINT;
```

Вывод:

```
+--------------------------+
| summary                  |
+--------------------------+
| Column has been altered. |
+--------------------------+
```

Создание таблицы с партициями

В Impala вы можете создавать Iceberg-таблицы с партициями, используя традиционное партиционирование по значению (value-based partitioning). Синтаксис показан ниже:

```
CREATE TABLE default.impala_ice_test_partitioned (
    txn_id int,
    txn_value double,
    txn_date date)
PARTITIONED BY (acc_id int)
STORED AS ICEBERG;
```

Также можно создать таблицу с партициями, используя одно или несколько выражений преобразования (transform expressions). Пример показан ниже.

```
CREATE TABLE default.impala_ice_test_partitioned_transform (
    txn_id int,
    acc_id int,
    txn_value double,
    txn_date date)
PARTITIONED BY SPEC (
    BUCKET(3, acc_id), ①
    day(txn_date)) ②
STORED AS ICEBERG;
```

① Первая часть выражения преобразования заставляет Iceberg распределять вставленные данные между 3 бакетами. Для каждого значения `acc_id`, которое должно быть записано, Iceberg вычисляет хеш и получает остаток от деления на 3 (`hash(acc_id) mod 3`). Полученное значение определяет бакет, в который будут записаны данные.

② Затем в рамках каждого бакета Iceberg создает партиции по дате.

Для таблицы, созданной с помощью вышеуказанной команды, Impala создает следующую структуру директорий в HDFS.

```
[admin@ka-adh-2 ~]$ hdfs dfs -ls -R /apps/hive/warehouse
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01 /apps/hive/warehouse/impala_ice_test_partitioned_transform
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01 /apps/hive/warehouse/impala_ice_test_partitioned_transform/data
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=0
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=0/txn_date_day=2023-01-03
-rw-r--r--   3 impala hadoop       1153 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=0/txn_date_day=2023-01-03/314da83d47eee296-
145c34e600000000_1592985449_data.0.parq
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=1
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=1/txn_date_day=2023-01-02
-rw-r--r--   3 impala hadoop       1153 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=1/txn_date_day=2023-01-02/314da83d47eee296-
145c34e600000000_584411471_data.0.parq
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=2
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=2/txn_date_day=2023-01-01
-rw-r--r--   3 impala hadoop       1153 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/data/acc_id_bucket=2/txn_date_day=2023-01-01/314da83d47eee296-
145c34e600000000_1148089692_data.0.parq
drwxrwxr-x   - impala hadoop          0 2024-07-05 14:01 /apps/hive/warehouse/impala_ice_test_partitioned_transform/metadata
-rw-r--r--   3 impala hadoop       2385 2024-07-05 14:00
/apps/hive/warehouse/impala_ice_test_partitioned_transform/metadata/00000-b04f03f7-f770-44a1-825b-8e84e4392f41.metadata.json
-rw-r--r--   3 impala hadoop       3564 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/metadata/00001-13512042-67f5-4327-933c-978d41be5df1.metadata.json
-rw-r--r--   3 impala hadoop       6539 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/metadata/24a8910e-e616-4d4b-ace5-46987f83fe9a-m0.avro
-rw-r--r--   3 impala hadoop       3804 2024-07-05 14:01
/apps/hive/warehouse/impala_ice_test_partitioned_transform/metadata/snap-4359281668985942129-1-24a8910e-e616-4d4b-ace5-
46987f83fe9a.avro
```

Для просмотра партиций можно использовать команду `SHOW PARTITIONS`. Пример вывода показан ниже.

```
+---------------------------------------------+----------------+-----------------+
| Partition                                   | Number Of Rows | Number Of Files |
+---------------------------------------------+----------------+-----------------+
| {"acc_id_bucket":"0","txn_date_day":"19360"} | 1              | 1               |
| {"acc_id_bucket":"0","txn_date_day":"19361"} | 1              | 1               |
| {"acc_id_bucket":"0","txn_date_day":"19723"} | 1              | 1               |
| {"acc_id_bucket":"0","txn_date_day":"19725"} | 1              | 1               |
| {"acc_id_bucket":"1","txn_date_day":"19359"} | 1              | 1               |
| {"acc_id_bucket":"1","txn_date_day":"19360"} | 1              | 1               |
| {"acc_id_bucket":"2","txn_date_day":"19358"} | 1              | 1               |
| {"acc_id_bucket":"2","txn_date_day":"19361"} | 1              | 1               |
| {"acc_id_bucket":"2","txn_date_day":"19392"} | 1              | 1               |
+---------------------------------------------+----------------+-----------------+
```

Iceberg поддерживает эволюцию партиций (partition evolution), что позволяет изменять партиционирование таблицы без необходимости перезаписи существующих данных. Партиционирование таблицы можно изменить с помощью оператора `ALTER TABLE SET PARTITION SPEC`, как показано в примере ниже.

```
ALTER TABLE default.impala_ice_test_partitioned
SET PARTITION SPEC (DAY(txn_date));
```

Вывод:

```
+-------------------------+
| summary                 |
+-------------------------+
| Updated partition spec. |
+-------------------------+
```

Команды DML

Запись данных

В Impala вы можете записывать данные в таблицу Iceberg с помощью команд `INSERT INTO` и `INSERT OVERWRITE`.

Пример:

```
INSERT INTO default.impala_ice_test VALUES
(1, 1001, 150.50, '2023-01-04'),
(2, 1002, 200.50, '2023-01-03'),
(3, 1003, 50.00, '2024-01-03');
```

С помощью `INSERT OVERWRITE` можно заменить данные в таблице на результат запроса. Например:

```
INSERT OVERWRITE TABLE default.impala_ice_test_tmp
SELECT * from default.impala_ice_test;
```

ПРИМЕЧАНИЕ

Операция `INSERT OVERWRITE` не поддерживается для таблиц, использующих преобразование партиций типа `BUCKET`.

Удаление данных

Вы можете удалить данные из таблиц Iceberg V2 с помощью команды `DELETE`. Пример:

```
DELETE FROM default.impala_ice_test
WHERE txn_id = 1;
```

ПРИМЕЧАНИЕ

Операция `DELETE` поддерживается только для таблиц Iceberg V2. Попытка удаления строк из таблицы V1 приводит к ошибке.

Обновление данных

В Impala вы можете обновлять таблицы Iceberg V2 как показано в следующем примере.

```
UPDATE default.impala_ice_test
SET txn_value = txn_value + 100;
```

ПРИМЕЧАНИЕ

Операция `UPDATE` поддерживается только для таблиц Iceberg V2. Попытка обновления данных в таблице V1 приводит к ошибке.

Вы также можете использовать оператор `UPDATE FROM` для обновления целевой таблицы Iceberg на основе некой исходной таблицы (исходная таблица необязательно должна являться Iceberg-таблицей).

Пример:

```
UPDATE default.impala_ice_test
SET default.impala_ice_test.acc_id = src.acc_id,
    default.impala_ice_test.txn_date = src.txn_date,
FROM default.impala_ice_test, default.impala_ice_test_src src
WHERE default.impala_ice_test.txn_id = src.txn_internal_id;
```

Если в предложении `WHERE` есть несколько совпадений, Impala выдаст ошибку. Существует несколько ограничений на использование команды `UPDATE FROM`, например, поддержка единственного формата Parquet, проблемы с обновлением столбцов с партициями и т. д. Больше информации об ограничениях вы можете найти в документации Impala.

Time travel

Iceberg поддерживают функцию time travel, которая позволяет запрашивать данные из определенного снепшота таблицы, созданного в прошлом, и на который можно ссылаться по идентификатору или временной метке. Для выполнения time-travel запросов используйте предложение `FOR SYSTEM_TIME AS OF <timestamp>` и `FOR SYSTEM_VERSION AS OF <snapshot-id>`. Ниже приведены несколько примеров использования.

В следующем примере запрашиваются данные из ближайшего снепшота, который старше указанной временной метки:

```
SELECT * FROM default.impala_ice_test
FOR SYSTEM_TIME AS OF '2024-07-02 12:00:00';
```

Вывод:

```
+--------+--------+-----------+------------+
| txn_id | acc_id | txn_value | txn_date   |
+--------+--------+-----------+------------+
| 2      | 1002   | 63.5      | 2024-02-04 |
| 1      | 1001   | 110.0     | 2023-01-01 |
+--------+--------+-----------+------------+
```

В следующем запросе нужный снепшот определяется путем вычитания отрезка времени из текущего момента.

```
SELECT * FROM default.impala_ice_test
FOR SYSTEM_TIME AS OF now() - interval 1 minute;
```

Вывод:

```
+--------+--------+-----------+------------+
| txn_id | acc_id | txn_value | txn_date   |
+--------+--------+-----------+------------+
| 2      | 1002   | 63.5      | 2024-02-04 |
+--------+--------+-----------+------------+
```

Следующий запрос получает данные из снепшота с указанием его ID:

```
SELECT * FROM default.impala_ice_test
FOR SYSTEM_VERSION AS OF 7308000224696874146;
```

Результат:

```
+--------+--------+-----------+------------+
| txn_id | acc_id | txn_value | txn_date   |
+--------+--------+-----------+------------+
| 1      | 1001   | 10.0      | 2023-01-01 |
+--------+--------+-----------+------------+
```

Чтобы получить информацию о снепшотах, доступных для конкретной таблицы, используйте команду `DESCRIBE HISTORY`, как показано в следующем примере:

```
DESCRIBE HISTORY default.impala_ice_test;
```

Результат выполнения команды:

```
+-------------------------------+---------------------+---------------------+---------------------+
| creation_time                 | snapshot_id         | parent_id           | is_current_ancestor |
+-------------------------------+---------------------+---------------------+---------------------+
| 2024-04-07 21:41:56.959000000 | 7308000224696874146 | NULL                | TRUE                |
| 2024-04-07 21:42:19.537000000 | 3739750708916322181 | 7308000224696874146 | TRUE                |
| 2024-04-07 22:04:58.577000000 | 3634103131974849338 | 3739750708916322181 | TRUE                |
+-------------------------------+---------------------+---------------------+---------------------+
```

Roll back

При каждом изменении содержимого Iceberg-таблицы Iceberg создает снепшот для этой таблицы. С помощью снепшотов вы можете откатить состояние таблицы к некоторой версии в прошлом, обозначенной идентификатором снепшота. При откате таблицы создается новый снепшот с тем же идентификатором, но с новой временной меткой создания. Используя оператор `ALTER TABLE <tbl_name> EXECUTE ROLLBACK`, вы можете откатить таблицу к предыдущим версиям, как показано ниже.

Например, предположим, что существует таблица Iceberg со следующим содержимым:

```
+--------+--------+-----------+------------+
| txn_id | acc_id | txn_value | txn_date   |
+--------+--------+-----------+------------+
| 2      | 1002   | 63.5      | 2024-02-04 |
| 1      | 1001   | 110.0     | 2023-01-01 |
+--------+--------+-----------+------------+
```

Следующая команда откатывает состояние таблицы до снепшота, обозначенного ID:

```
ALTER TABLE default.impala_ice_test
EXECUTE ROLLBACK(7308000224696874146);
```

После выполнения этой команды выборка из таблицы с помощью `SELECT` возвращает данные, которые хранились в таблице в момент создания указанного снепшота. Например:

```
+--------+--------+-----------+------------+
| txn_id | acc_id | txn_value | txn_date   |
+--------+--------+-----------+------------+
| 1      | 1001   | 10.0      | 2023-01-01 |
+--------+--------+-----------+------------+
```

Следующая команда откатывает таблицу до версии снепшота, который был создан раньше указанной временной метки:

```
ALTER TABLE default.impala_ice_test
EXECUTE ROLLBACK('2024-07-05 12:00:00');
```

По умолчанию Iceberg накапливает снепшоты, пока они не будут удалены пользователем. Вы можете удалить ненужные снепшоты с помощью команды `ALTER TABLE … EXECUTE expire_snapshots(<timestamp>)`.

Примеры:

```
ALTER TABLE default.impala_ice_test
EXECUTE expire_snapshots('2024-07-05 12:00:00');
```

```
ALTER TABLE default.impala_ice_test
EXECUTE expire_snapshots(now() - interval 10 days);
```

Конвертация типов данных Iceberg

В следующей таблице описано сопоставление типов данных Iceberg и типов SQL в Impala.

Тип данных Iceberg	Тип SQL в Impala
boolean	BOOLEAN
int	INTEGER
long	BIGINT
float	FLOAT
double	DOUBLE
decimal(P, S)	DECIMAL(P, S)
date	DATE
time	—
timestamp	TIMESTAMP
timestamptz	Поддержка чтения с использованием TIMESTAMP
string	STRING
uuid	—
fixed(L)	—
binary	—
struct	STRUCT (только чтение)
list	ARRAY (только чтение)
map	MAP (только чтение)

Управление сервисом Impala через ADCM

Евгения Кузина

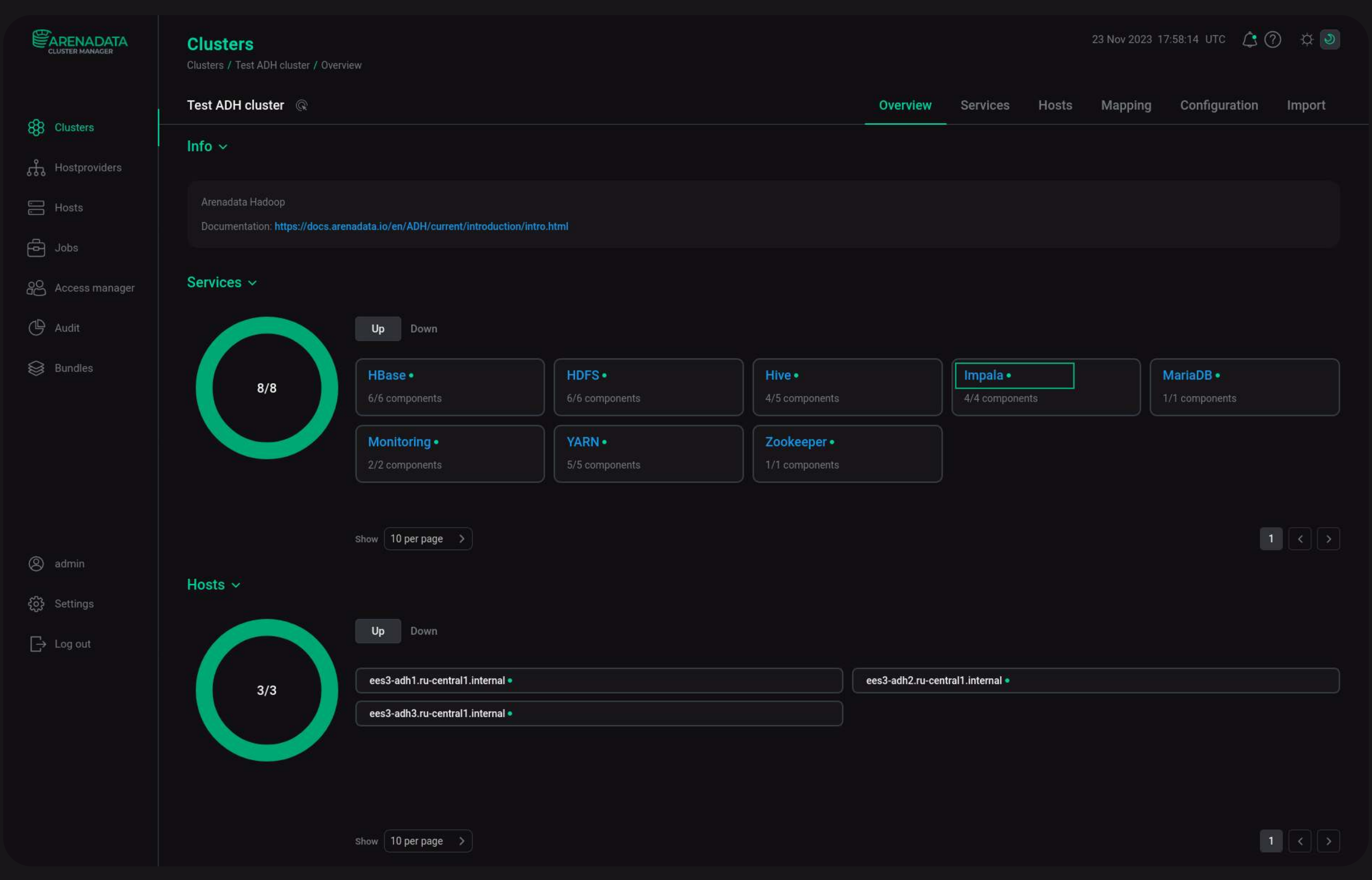
Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Impala. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Impala, перечислены в таблице ниже.

Действия с сервисом Impala

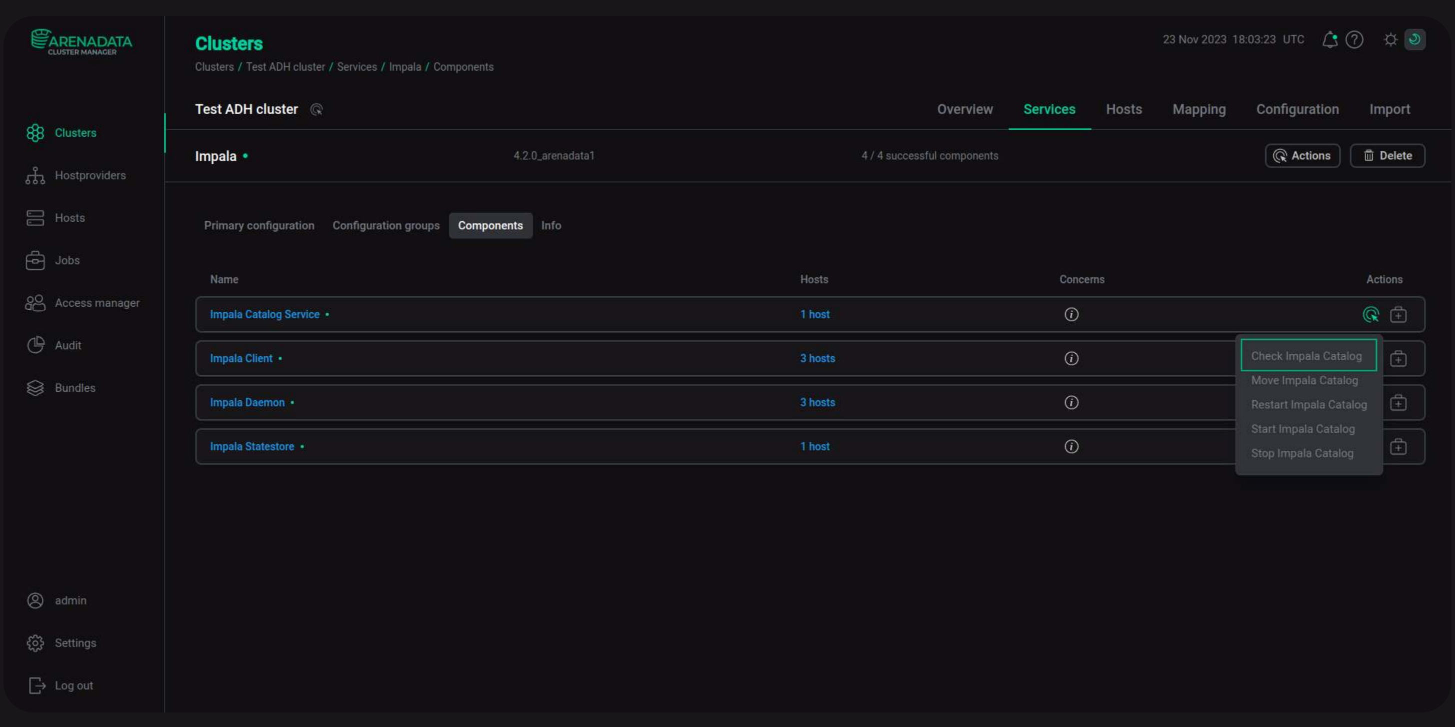
Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Impala
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются

У компонентов Impala также есть действия, которые работают с соответствующими демонами. Чтобы выполнить эти действия, кликните сервис ***Impala*** на вкладке кластера ***Services***. Также можно нажать ***Impala*** рядом с диаграммой ***Services*** на вкладке кластера ***Overview***.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / Test ADH cluster / Overview
23 Nov 2023 17:58:14 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Info
Arenadata Hadoop
Documentation: https://docs.arenadata.io/en/ADH/current/introduction/intro.html
Services
8/8
Up
Down
HBase
6/6 components
HDFS
6/6 components
Hive
4/5 components
Impala
4/4 components
MariaDB
1/1 components
Monitoring
2/2 components
YARN
5/5 components
Zookeeper
1/1 components
Show
10 per page
Hosts
3/3
ees3-adh1.ru-central1.internal
ees3-adh2.ru-central1.internal
ees3-adh3.ru-central1.internal

Вкладка кластера "Overview" с диаграммой "Services"

Откроется вкладка ***Components*** сервиса Impala. В колонке ***Actions*** кликните иконку для нужного компонента и выберите действие из выпадающего списка.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / Test ADH cluster / Services / Impala / Components
23 Nov 2023 18:03:23 UTC
Test ADH cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Impala
4.2.0_arenadata1
4 / 4 successful components
Actions
Delete
Primary configuration
Configuration groups
Components
Info
Name
Hosts
Concerns
Actions
Impala Catalog Service
1 host
Impala Client
3 hosts
Impala Daemon
3 hosts
Impala Statestore
1 host
Check Impala Catalog
Move Impala Catalog
Restart Impala Catalog
Start Impala Catalog
Stop Impala Catalog
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out

Действия с компонентами Impala

Действия, доступные для каждого компонента, перечислены в таблицах ниже.

Действия с компонентом Impala Daemon

Действие	Описание
Add/Remove Daemon(s)	Позволяет удалять или добавлять компонент Impala Daemon на хосты кластера. При выполнении этого действия отображается окно с компонентами и хостами, которое позволяет перераспределить компонент Impala Daemon между хостами, как описано в статье Добавление компонентов. Компонент Impala Daemon может быть установлен на нескольких хостах кластера
Check Impala Daemon	Проверяет, правильно ли работают все демоны Impala Daemon в кластере
Restart Impala Daemon	Перезапускает все демоны Impala в кластере
Start Impala Daemon	Запускает все демоны Impala в кластере
Stop Impala Daemon	Останавливает все демоны Impala в кластере

Действия с компонентом Impala Statestore

Действие	Описание
Check Impala Statestore	Проверяет, правильно ли работает демон Impala Statestore
Move Impala Statestore	Позволяет переместить компонент Impala Statestore с одного хоста на другой в том же кластере. При выполнении этого действия отображается окно с компонентами и хостами, которое позволяет удалить компонент Impala Statestore с одного хоста и добавить его на другой, как описано в статье Добавление компонентов. Компонент Impala Statestore может быть установлен только на одном хосте в кластере
Restart Impala Statestore	Перезапускает демон Impala Statestore
Start Impala Statestore	Запускает демон Impala Statestore
Stop Impala Statestore	Останавливает демон Impala Statestore

Действия с компонентом Impala Catalog Service

Действие	Описание
Check Impala Catalog	Проверяет, правильно ли работает демон Impala Catalog
Move Impala Catalog	Позволяет переместить компонент Impala Catalog Service с одного хоста на другой в том же кластере. При выполнении этого действия отображается окно с компонентами и хостами, которое позволяет удалить компонент Impala Catalog Service с одного хоста и добавить его на другой, как описано в статье Добавление компонентов. Компонент Impala Catalog Service может быть установлен только на одном хосте в кластере
Restart Impala Catalog	Перезапускает демон Impala Catalog Service
Start Impala Catalog	Запускает демон Impala Catalog Service
Stop Impala Catalog	Останавливает демон Impala Catalog Service

Действия с компонентом Impala Client

Действие	Описание
Add/Remove Client(s)	Позволяет удалять или добавлять компонент Impala Client на хосты кластера. При выполнении этого действия отображается окно с компонентами и хостами, которое позволяет перераспределить компонент Impala Client между хостами, как описано в статье Добавление компонентов. Компонент Impala Client может быть установлен на нескольких хостах кластера

Обзор Kyuubi UI

Елена Костюченко

Содержание

Назад к содержанию

Overview

Management

- Session
- Operation
- Engine
- Server

Swagger

У Kyuubi есть встроенный веб-сервер, который предоставляет доступ к REST API Kyuubi, информации о состоянии сервера, сессиях, операциях и движках Kyuubi.

> **ПРИМЕЧАНИЕ**
>
> Веб-интерфейс Kyuubi является экспериментальным и по умолчанию недоступен. Чтобы включить его, установите для параметра `kyuubi.frontend.protocols` значение `THRIFT_BINARY,REST` вручную или через ADCM.

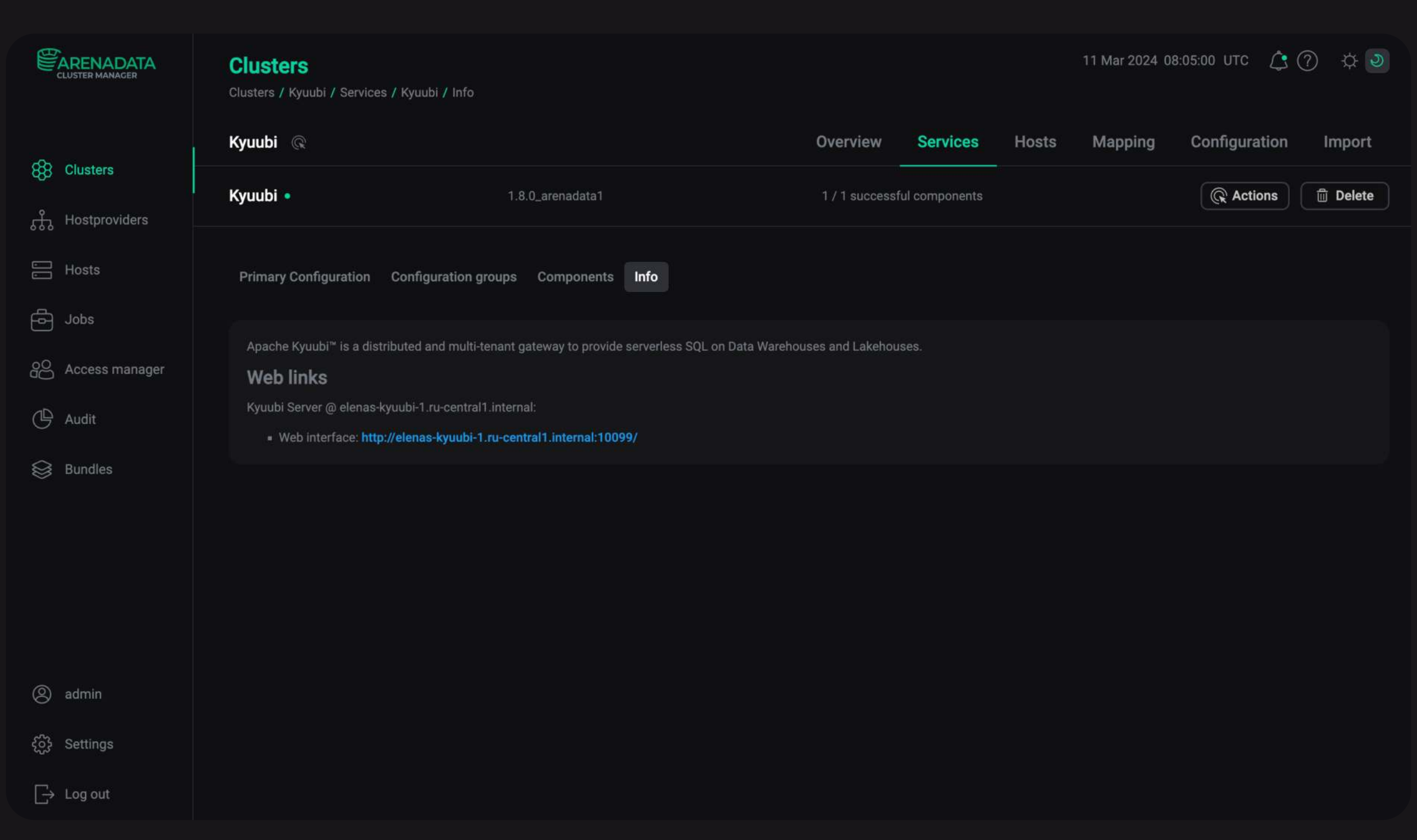
Веб-интерфейс доступен по следующему URL: *http://<server-hostname>:10099/*, где `<server-hostname>` — это IP-адрес или FQDN хоста с установленным сервером Kyuubi.

Для безопасных кластеров используйте `https://` вместо `http://`.

Вы также можете найти ссылку на пользовательский интерфейс Kyuubi в ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на ***Kyuubi***.
3. Выберите вкладку ***Info***.

Чтобы изменить порт веб-интерфейса по умолчанию, укажите новое значение в параметре `kyubi.frontend.rest.bind.port` в файле конфигурации *kyuubi-defaults.conf* вручную или через ADCM.



ARENADATA CLUSTER MANAGER
Clusters
Clusters / Kyuubi / Services / Kyuubi / Info
11 Mar 2024 08:05:00 UTC
Kyuubi
Overview
Services
Hosts
Mapping
Configuration
Import
Kyuubi
1.8.0_arenadata1
1 / 1 successful components
Actions
Delete
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Primary Configuration
Configuration groups
Components
Info

Apache Kyuubi™ is a distributed and multi-tenant gateway to provide serverless SQL on Data Warehouses and Lakehouses.

Web links

Kyuubi Server @ elenas-kyuubi-1.ru-central1.internal:

- Web interface: http://elenas-kyuubi-1.ru-central1.internal:10099/

Ссылка на веб-интерфейс сервиса Kyuubi в ADCM

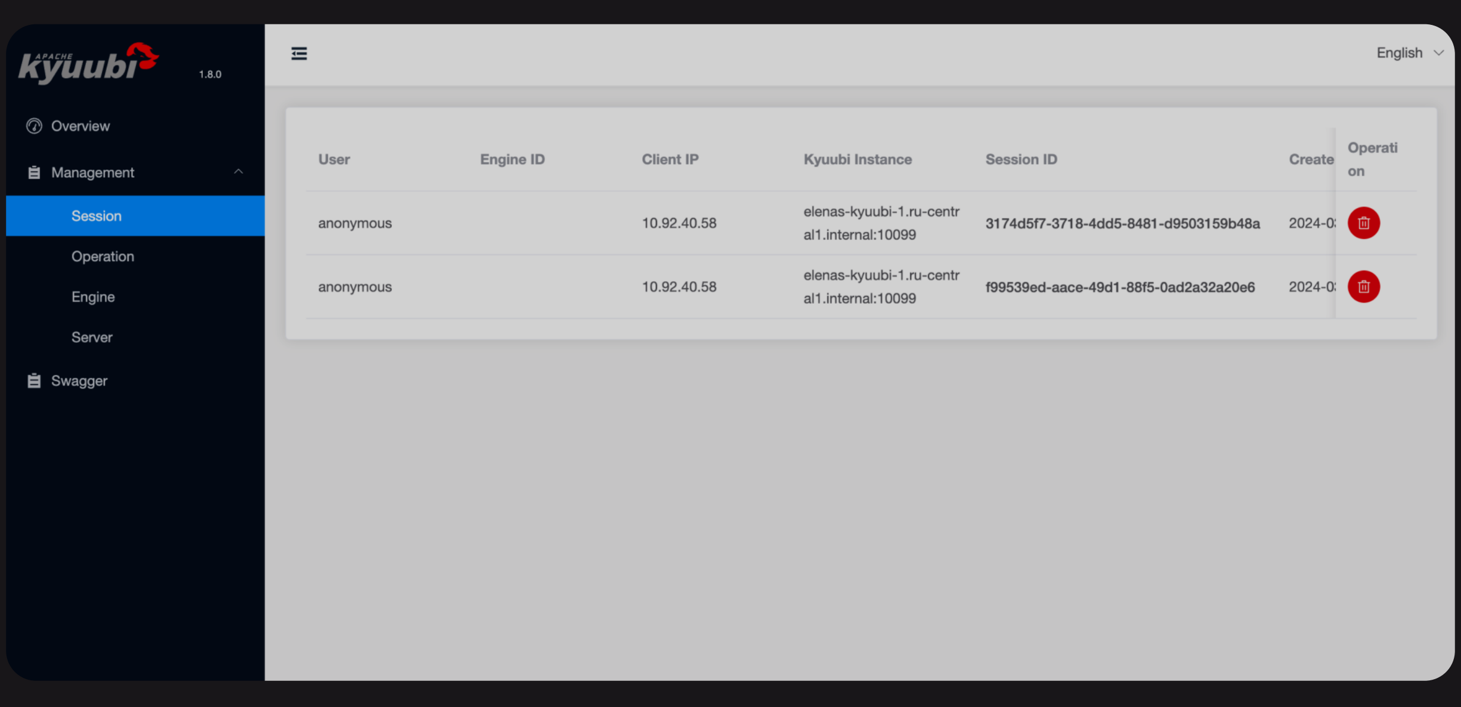
Overview

В текущей версии интерфейса страница ***Overview*** не содержит информации.

Management

В разделе ***Management*** отображается информация о сессиях, операциях, движках и серверах Kyuubi.

Session



APACHE kyuubi
1.8.0
Overview
Management
Session
Operation
Engine
Server
Swagger
English

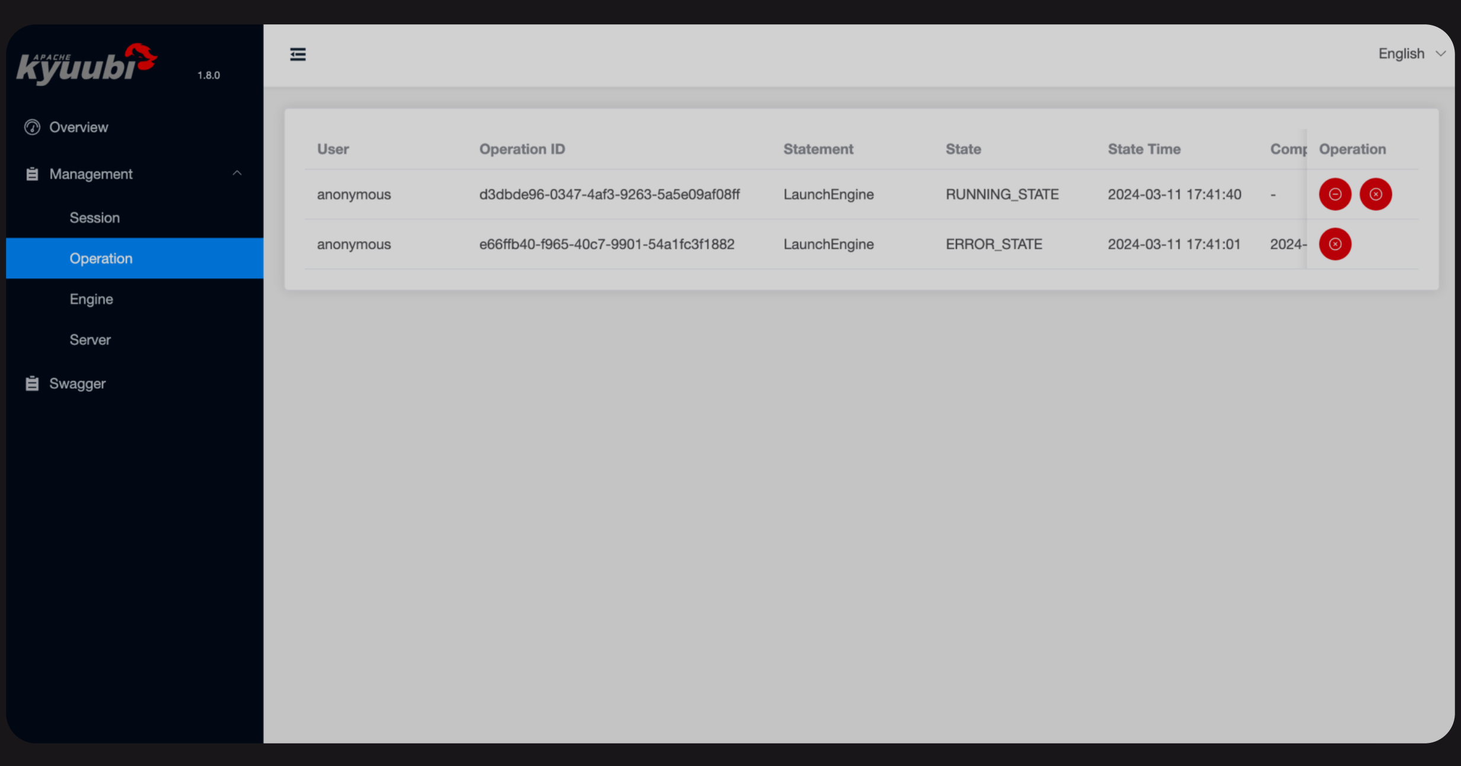
User	Engine ID	Client IP	Kyuubi Instance	Session ID	Create	Operation
anonymous		10.92.40.58	elenas-kyuubi-1.ru-central1.internal:10099	3174d5f7-3718-4dd5-8481-d9503159b48a	2024-0:	
anonymous		10.92.40.58	elenas-kyuubi-1.ru-central1.internal:10099	f99539ed-aace-49d1-88f5-0ad2a32a20e6	2024-0:	

Страница Session

На странице ***Session*** отображается таблица с информацией о сессиях пользователей:

- ***User*** — имя пользователя, создавшего сессию;
- ***Engine ID*** — идентификатор движка, используемого пользователем;
- ***Client IP*** — IP-адрес клиента, через который был создан запрос;
- ***Kyuubi Instance*** — FQDN и порт сервера Kyuubi, на котором зарегистрирована сессия;
- ***Session ID*** — идентификатор сессии, созданной пользователем;
- ***Create Time*** — время создания сессии;
- ***Operation*** — действия, доступные для этой сессии через веб-интерфейс.

Operation



APACHE kyuubi
1.8.0
Overview
Management
Session
Operation
Engine
Server
Swagger
English

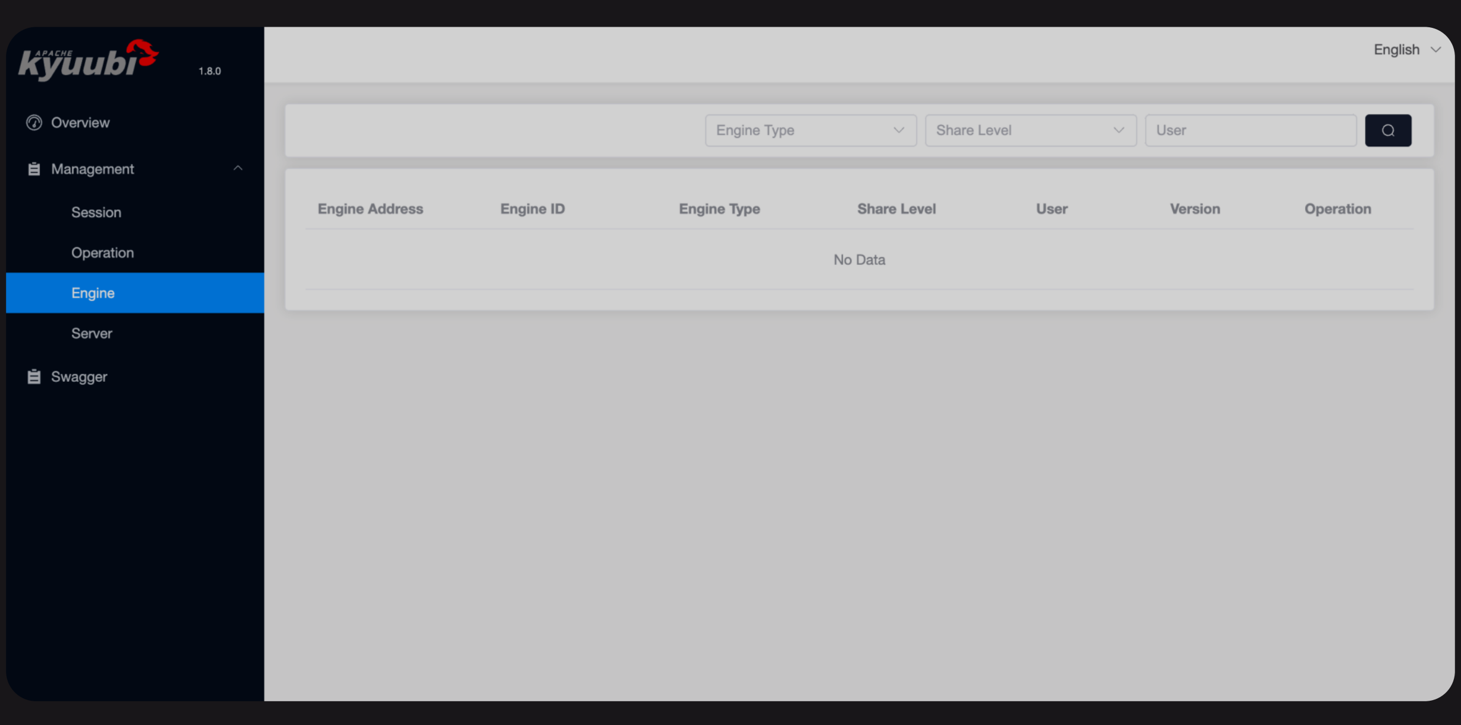
User	Operation ID	Statement	State	State Time	Comp	Operation
anonymous	d3dbde96-0347-4af3-9263-5a5e09af08ff	LaunchEngine	RUNNING_STATE	2024-03-11 17:41:40	-	
anonymous	e66ffb40-f965-40c7-9901-54a1fc3f1882	LaunchEngine	ERROR_STATE	2024-03-11 17:41:01	2024-	

Страница Operation

На странице ***Operation*** отображается таблица с подробной информацией об операциях, выполняемых пользователями:

- ***User*** — имя пользователя, создавшего сессию;
- ***Operation ID*** — идентификатор операции;
- ***Statement*** — SQL-запрос, выполняемый в операции;
- ***State*** — текущее состояние операции;
- ***State Time*** — время, в течение которого операция находится в текущем состоянии;
- ***Completed Time*** — время окончания выполнения операции;
- ***Duration*** — время, за которое операция была выполнена;
- ***Operation*** — действия, доступные для этой операции через веб-интерфейс.

Engine



APACHE kyuubi
1.8.0
Overview
Management
Session
Operation
Engine
Server
Swagger
English
Engine Type
Share Level
User

Engine Address	Engine ID	Engine Type	Share Level	User	Version	Operation
No Data						

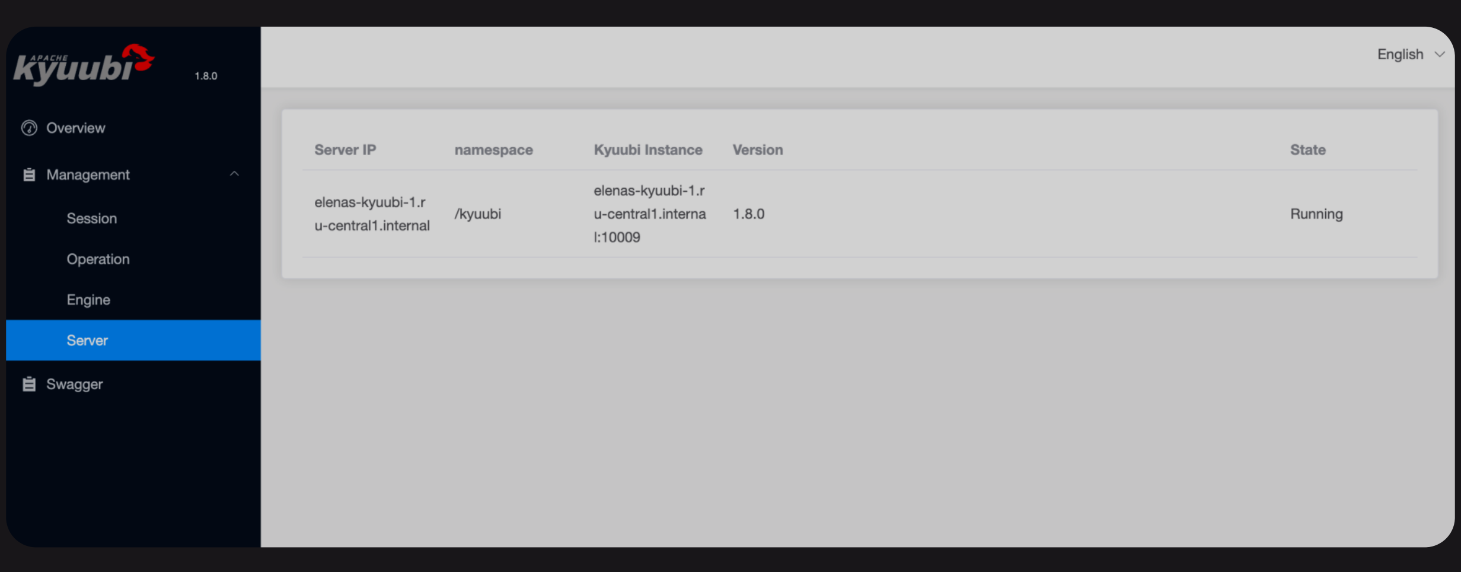
Страница Engine

На странице ***Engine*** отображается таблица с подробной информацией о движках Kyuubi:

- ***Engine Address*** — IP-адрес сервера движка;
- ***Engine ID*** — идентификатор движка;
- ***Engine Type*** — тип движка (текущая версия веб-интерфейса отображает только движки SPARK-SQL);
- ***Share Level*** — тип общего доступа движка;
- ***User*** — имя пользователя, создавшего движок;
- ***Version*** — версия сервера Kyuubi, создавшего движок;
- ***Operation*** — действия, доступные для этого движка через веб-интерфейс.

> **РЕКОМЕНДАЦИЯ**
>
> По умолчанию на странице отображаются не все существующие движки. Чтобы выбрать какие движки показать, используйте фильтр вверху страницы.

Server



APACHE kyuubi
1.8.0
Overview
Management
Session
Operation
Engine
Server
Swagger
English

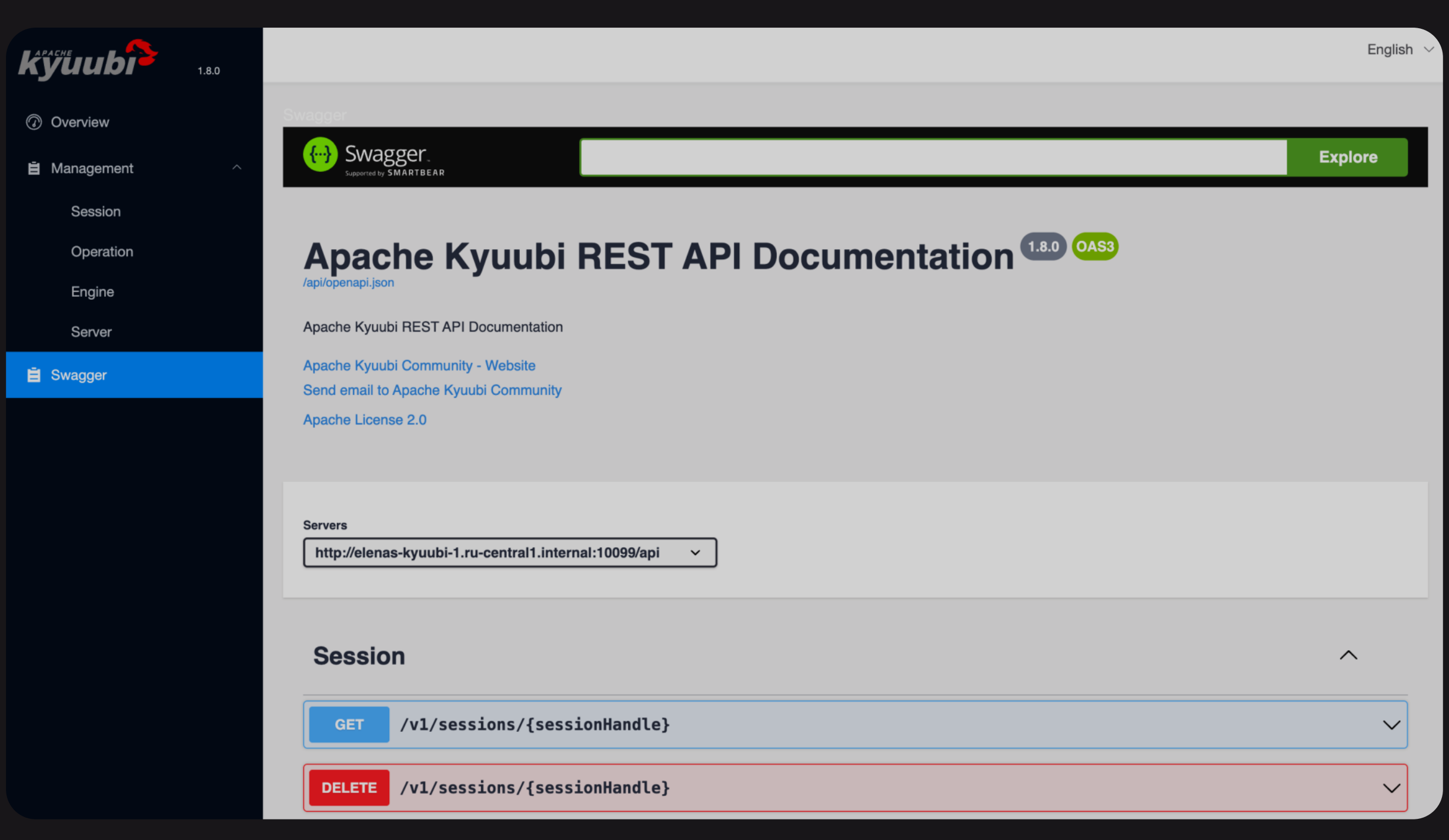
Server IP	namespace	Kyuubi Instance	Version	State
elenas-kyuubi-1.ru-central1.internal	/kyuubi	elenas-kyuubi-1.ru-central1.internal:10009	1.8.0	Running

Страница Server

На странице ***Server*** отображается таблица с подробной информацией о серверах Kyuubi:

- ***Server IP*** — IP-адрес сервера Kyuubi;
- ***namespace*** — пространство имен, к которому принадлежит сервер;
- ***Kyuubi Instance*** — FQDN и порт сервера;
- ***Version*** — версия сервера Kyuubi;
- ***State*** — текущее состояние сервера (например, `Running`).

Swagger



APACHE kyuubi
1.8.0
Overview
Management
Session
Operation
Engine
Server
Swagger
English
Swagger
Supported by SMARTBEAR
Explore
1.8.0
OAS3
/api/openapi.json
Apache Kyuubi Community - Website
Send email to Apache Kyuubi Community
Apache License 2.0
Servers
Session
GET
DELETE

Apache Kyuubi REST API Documentation

Apache Kyuubi REST API Documentation

Servers: http://elenas-kyuubi-1.ru-central1.internal:10099/api

GET /v1/sessions/{sessionHandle}

DELETE /v1/sessions/{sessionHandle}

Страница Swagger

На странице ***Swagger*** отображается интерактивная справка по REST API Kyuubi. Она позволяет отправлять запросы Kyuubi через веб-интерфейс.

Чтобы отправить запрос, нажмите на нужный метод, выберите ***Try it out***, введите параметры, если необходимо, и нажмите ***Execute***.

Аутентификация LDAP в Kyuubi

Сергей Тихомиров

Содержание

Назад к содержанию

Активация LDAP

Проверка

Kyuubi поддерживает LDAP-аутентификацию для клиентов (например, BeeLine shell) и JDBC/ODBC-драйверов.

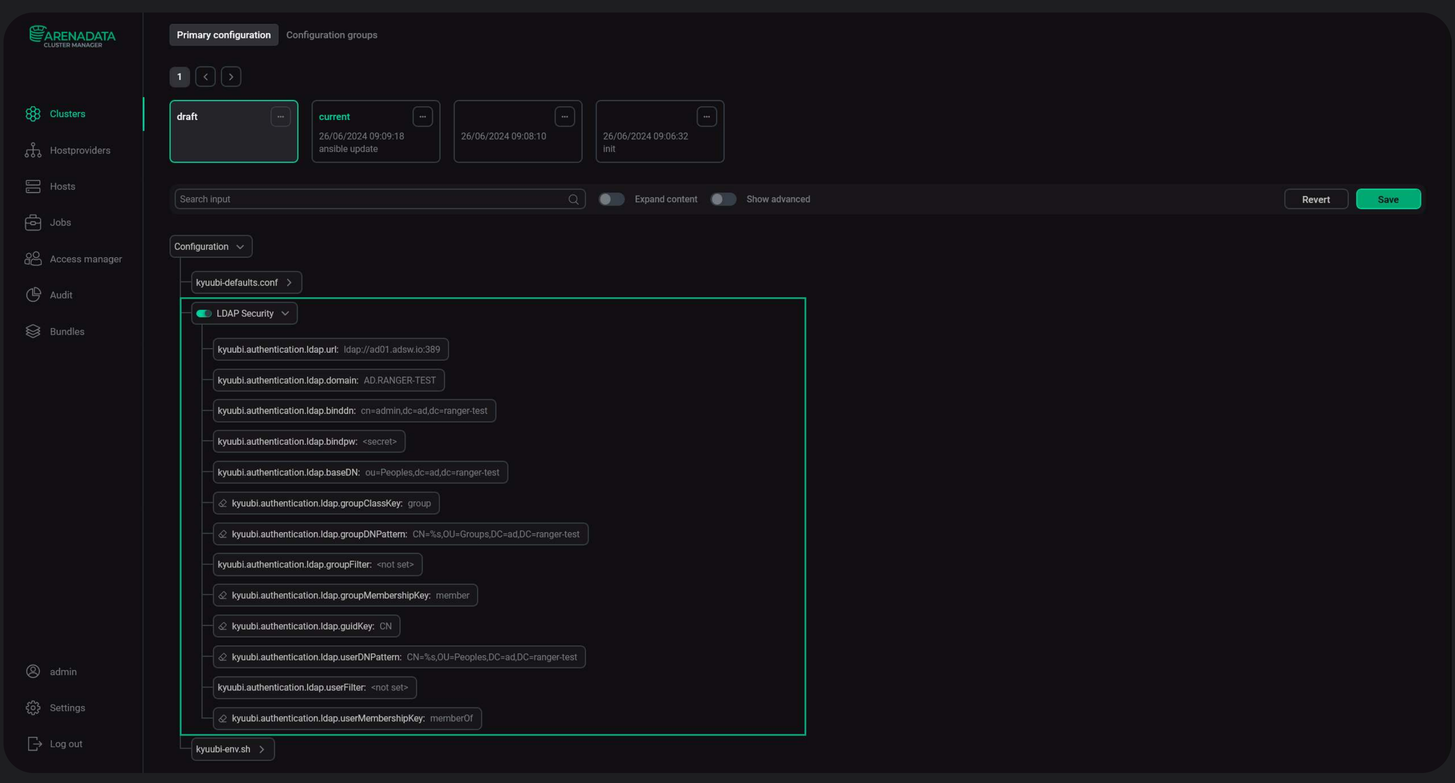
> **ПРИМЕЧАНИЕ**
>
> При активации Kerberos в кластере происходит автоматическое включение LDAP-аутентификации для Kyuubi.

Активация LDAP

Для активации LDAP-аутентификации следуйте шагам ниже:

1. На странице ***Clusters*** выберите ваш кластер ADH и перейдите на вкладку ***Services*** в меню кластера.
2. Выберите сервис ***Kyuubi*** и перейдите на вкладку ***Components***.
3. В настройках компонента ***Kyuubi Server*** активируйте группу параметров ***LDAP Security***.
4. Заполните параметры LDAP для Active Directory или FreeIPA и перезапустите Kyuubi.

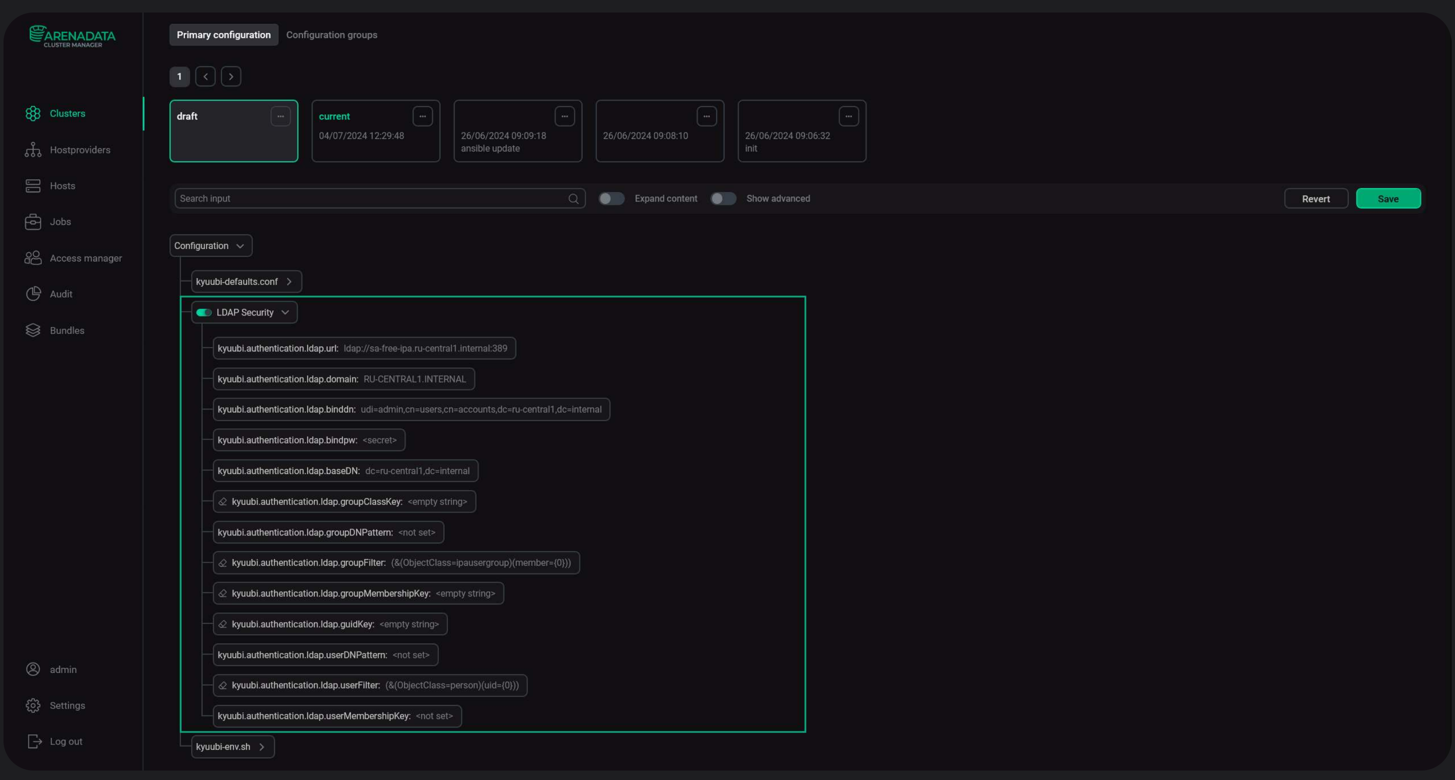
Active Directory


ARENADATA
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Primary configuration
Configuration groups
1
draft
current
26/06/2024 09:09:18
ansible update
26/06/2024 09:08:10
26/06/2024 09:06:32
init
Search input
Expand content
Show advanced
Revert
Save
Configuration
kyuubi-defaults.conf
LDAP Security
kyuubi-env.sh

```
kyuubi.authentication.ldap.url: ldap://ad01.adsw.io:389
kyuubi.authentication.ldap.domain: AD.RANGER-TEST
kyuubi.authentication.ldap.binddn: cn=admin,dc=ad,dc=ranger-test
kyuubi.authentication.ldap.bindpw: <secret>
kyuubi.authentication.ldap.baseDN: ou=Peoples,dc=ad,dc=ranger-test
kyuubi.authentication.ldap.groupClassKey: group
kyuubi.authentication.ldap.groupDNPattern: CN=%s,OU=Groups,DC=ad,DC=ranger-test
kyuubi.authentication.ldap.groupFilter: <not set>
kyuubi.authentication.ldap.groupMembershipKey: member
kyuubi.authentication.ldap.guidKey: CN
kyuubi.authentication.ldap.userDNPattern: CN=%s,OU=Peoples,DC=ad,DC=ranger-test
kyuubi.authentication.ldap.userFilter: <not set>
kyuubi.authentication.ldap.userMembershipKey: memberOf
```

Пример конфигурации LDAP с AD

FreeIPA


ARENADATA
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Primary configuration
Configuration groups
1
draft
current
04/07/2024 12:29:48
26/06/2024 09:09:18
ansible update
26/06/2024 09:08:10
26/06/2024 09:06:32
init
Search input
Expand content
Show advanced
Revert
Save
Configuration
kyuubi-defaults.conf
LDAP Security
kyuubi-env.sh

```
kyuubi.authentication.ldap.url: ldap://sa-free-ipa.ru-central1.internal:389
kyuubi.authentication.ldap.domain: RU-CENTRAL1.INTERNAL
kyuubi.authentication.ldap.binddn: udi=admin,cn=users,cn=accounts,dc=ru-central1,dc=internal
kyuubi.authentication.ldap.bindpw: <secret>
kyuubi.authentication.ldap.baseDN: dc=ru-central1,dc=internal
kyuubi.authentication.ldap.groupClassKey: <empty string>
kyuubi.authentication.ldap.groupDNPattern: <not set>
kyuubi.authentication.ldap.groupFilter: (&(ObjectClass=ipausergroup)(member={0}))
kyuubi.authentication.ldap.groupMembershipKey: <empty string>
kyuubi.authentication.ldap.guidKey: <empty string>
kyuubi.authentication.ldap.userDNPattern: <not set>
kyuubi.authentication.ldap.userFilter: (&(ObjectClass=person)(uid={0}))
kyuubi.authentication.ldap.userMembershipKey: <not set>
```

Пример конфигурации LDAP с FreeIPA

Проверка

Для проверки работоспособности LDAP-аутентификации запустите команду BeeLine shell, подобную представленной ниже:

```
$ beeline -u 'jdbc:hive2://stikhomirov-adh-1.ru-central1.internal:2181,stikhomirov-adh-2.ru-central1.internal:2181,stikhomirov-adh-3.ru-central1.internal:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/67/kyuubi?kyuubi.engine.type=HIVE_SQL' -n <name> -p <password>
```

где:

- `<name>` — имя пользователя в LDAP.
- `<password>` — пароль для указанного пользователя.

При корректной работе ожидаемый вывод этой команды:

```
Connecting to jdbc:hive2://stikhomirov-adh-1.ru-central1.internal:2181,stikhomirov-adh-2.ru-central1.internal:2181,stikhomirov-adh-3.ru-central1.internal:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/67/kyuubi?kyuubi.engine.type=HIVE_SQL
24/03/06 14:21:03 [main]: INFO jdbc.HiveConnection: Connected to stikhomirov-adh-1.ru-central1.internal:10009
Connected to: Apache Hive (version 3.1.3)
Driver: Hive JDBC (version 3.1.3)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 3.1.3 by Apache Hive
```

Управление сервисом Kyuubi через ADCM

Сергей Тихомиров

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Kyuubi. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Kyuubi, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Kyuubi
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются

ВАЖНО

Для использования любых движков (например, Flink SQL/Hive/Spark SQL) на хосте с Kyuubi необходимо разместить клиент соответствующего движку сервиса. Для общих конфигураций необходимы клиенты HDFS и YARN.

У сервиса Kyuubi есть всего один компонент — ***Kyuubi Server***, у которого есть свой список действий.

Действия компонента Kyuubi Server

Действие	Описание
Add/Remove Kyuubi Server(s)	Добавляет/удаляет компонент(ы) Kyuubi Server. При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете распределить компоненты Kyuubi Server между хостами
Check	Выполняет проверку работоспособности компонента Kyuubi Server
Restart	Перезапускает компонент. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт компонента выполняется с учетом всех параметров, указанных в настройках этого компонента в ADCM. В противном случае настройки компонента в ADCM игнорируются
Start	Запускает компонент. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск компонента выполняется с учетом всех параметров, указанных в настройках этого компонента в ADCM. В противном случае настройки компонента в ADCM игнорируются
Stop	Останавливает компонент

Начало работы с MapReduce

Альберт Багдасарян

Содержание

Назад к содержанию

Код Mapper
Код Reducer
Код Driver

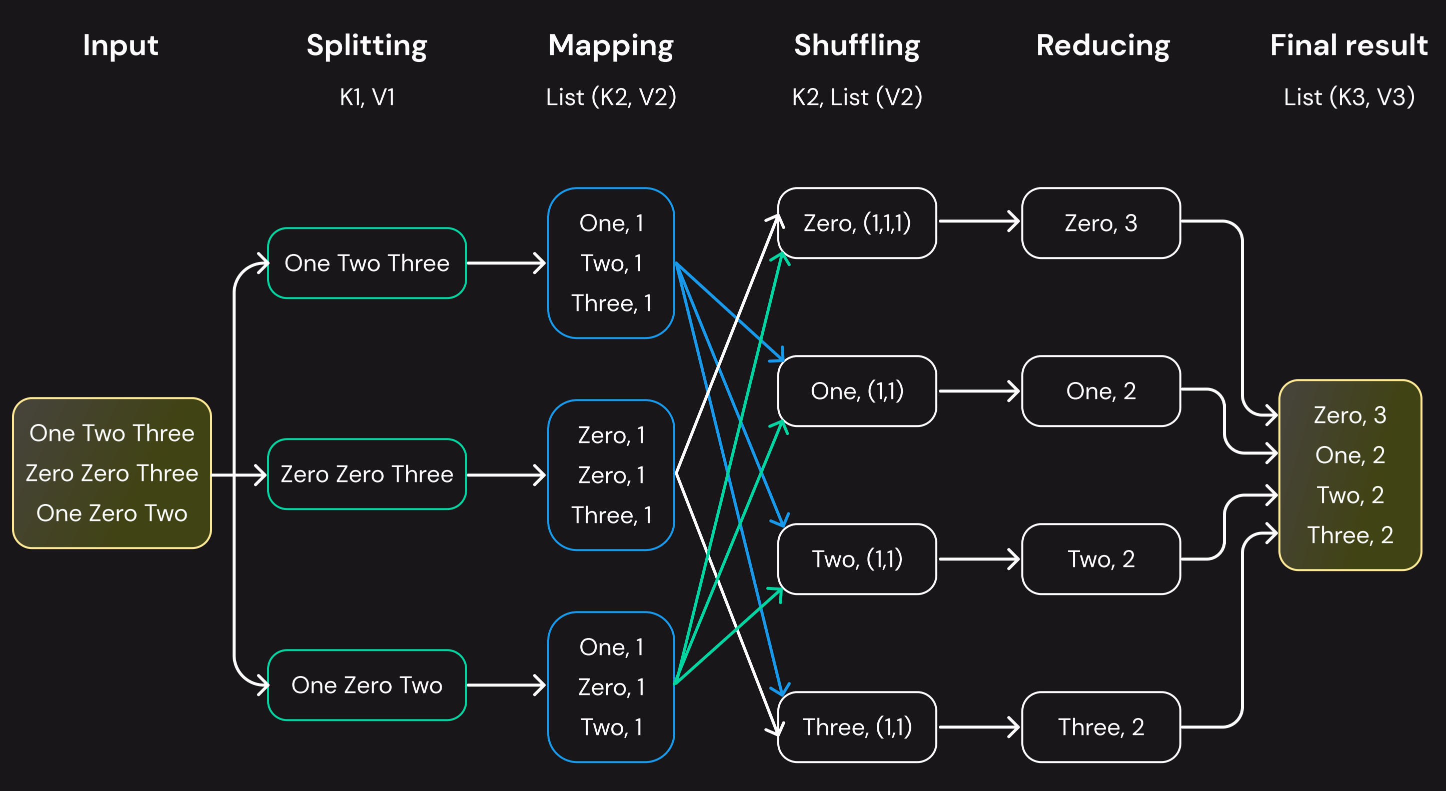
Код приложения для MapReduce состоит из трех частей:

- Mapper
- Reducer
- Driver

Эти части, как правило, базируются на классах с такими же названиями:

- **Mapper**. Этот класс отвечает за выполнение первой стадии обработки данных. Объект `InputSplit` является логическим представлением входных данных. Он описывает единицу работы, которую выполняет одна задача на стадии Map в задании для MapReduce. `RecordReader` обрабатывает каждую отдельную запись и генерирует соответствующую пару ключ-значение. MapReduce сохраняет эти промежуточные данные на локальном диске узла.
- **Reducer**. Этот класс отвечает за выполнение второй стадии обработки данных. MapReduce поставляет промежуточные данные, сгенерированные процессами mapper, на вход процессам reducer, которые формируют окончательные данные и сохраняют их в HDFS.
- **Driver**. Данный класс отвечает за настройку задания для MapReduce, которое будет запущено в Hadoop.

Рассматриваемый пример будет подсчитывать слова в текстовом файле *sample.txt* с использованием MapReduce. Мы определим уникальные слова и сколько раз каждое слово встречается в тексте. Обработка данных проходит несколько стадий, как представлено на следующей схеме.



Последовательность обработки данных

1. Как представлено на схеме, входной текст разделен на три части, распределенные по узлам DataNodes.
2. Каждый экземпляр mapper разбивает полученный текст на токены (слова) и присваивает каждому из них значение `1`.
3. Образуется столько пар ключ-значение (где ключ — это слово, а значение — 1), сколько всего слов в строке без учета уникальности. Например, из первой строки "One Two Three" получим три пары: ("One", 1), ("Two", 1) и ("Three", 1). Если какое-то слово встречается несколько раз, то образуется такое же количество одинаковых пар. Данный процесс работает одинаково на всех узлах.
4. После фазы map выполняется процесс сортировки и распределения (shuffling) данных так, что все пары с одинаковым ключом будут направлены на вход одного процесса агрегации (reducer).
5. После распределения каждый reducer получит уникальный ключ и список значений, где каждое значение равно 1, например: ("One", [1,1]), ("Zero", [1,1,1]) и так далее.
6. Агрегатор складывает все значения в списке. Например, в приведенной схеме агрегатор получает список [1,1] по ключу "One". Он подсчитывает количество единиц в списке и формирует окончательный результат – ("One", 2).
7. В конечном итоге MapReduce собирает все пары и записывает их в один файл.

Код Mapper

Код mapper выполняет следующие операции:

1. Создает класс `Map`, расширяющий класс `Mapper` из библиотеки MapReduce.
2. Определяет типы входных и выходных ключей и значений в угловых скобках в определении класса. Как вход, так и выход метода обработки данных внутри класса состоят из пар ключ-данные.

   Входные данные:

   - Ключ равен смещению соответствующей ему строки текста во входном файле — тип `LongWritable` (изменяемое длинное целое число).
   - Значением является строка, на которую указывает ключ — тип `Text`.

   Выходные данные:

   - Ключом является слово, выделенное как токен — тип `Text`.
   - Значение, закрепленное в коде, равно 1 — тип `IntWritable` (изменяемое целое число).
     Пример выходных данных: ("Dear", 1), ("Bear", 1) и так далее.

   Мы представили код, написанный на Java, который разделяет текст на токены и назначает каждому значение 1.

Код Reducer

Код агрегации (reducer) выполняет следующие операции:

1. Создает класс `Reduce`, расширяющий класс `Reducer` аналогично тому, как это сделано в Mapper.
2. Задает типы данных для входных и выходных ключей и значений аналогично тому, как это сделано для Mapper. На входе и выходе основного метода обработки данных используются пары ключ-значение.

   Входные данные:

   - Ключом является уникальное слово, выделенное в результате сортировки и распределения данных — тип `Text`.
   - Значением является список целых чисел, соответствующих ключу — тип `IntWritable`.
     Пример входных данных: ("Bear", [1, 1]), ("Car", [1, 1, 1]) и так далее.

   Выходные данные:

   - Ключом является одно из уникальных слов, обнаруженных во входном текстовом файле — тип `Text`.
   - Значением является частота вхождения слова в тексте — тип `IntWritable`.
     Пример выходных данных: ("Bear", 2), ("Car", 3) и так далее.

Рассмотренный код агрегирует значения, представленные в списке каждого ключа, и выдает окончательный результат. Процесс reducer выполняется для каждого уникального слова, а ограничение на общее количество таких параллельных процессов определено в *mapred-site.xml*.

Код Driver

Код driver выполняет следующие операции:

1. Настраивает задание (job) для MapReduce в Hadoop.
2. Дает название заданию, определяет типы входных и выходных данных для обработчиков mapper и reducer.
3. Определяет названия классов для mapper и reducer.
4. Задает пути к папкам для нахождения входных данных и записи выходных данных.

Метод `setInputFormatClass()` определяет способ чтения входных данных или единицу работы. Здесь мы выбрали тип `TextInputFormat`, в результате чего mapper считывает данные из входного файла построчно.

Метод `main()` является точкой входа. Этот метод активизирует объект `Configuration` для задания.

Подключение к MapReduce через CLI

Елена Костюченко

Содержание

Назад к содержанию

Обзор CLI

Примеры использования

Обзор CLI

MapReduce предоставляет интерфейс командной строки, реализованный в виде скрипта *bin/mapred*. Чтобы взаимодействовать с MapReduce через CLI, подключитесь к хосту кластера с компонентом MapReduce через SSH и запустите нужную команду MapReduce CLI.

Для проверки работы MapReduce CLI можно запросить версию с помощью команды:

```
$ mapred version
```

Пример вывода:

```
Hadoop 3.2.4
Source code repository git@ssh.gitlab.adsw.io:arenadata/infrastructure/code/ci/prj_adh.git -r
3cb85f40e394dcfb50fe77310908cce385381ba2
Compiled by jenkins on 2024-03-20T17:53Z
Compiled with protoc 2.5.0
From source with checksum ee031c16fe785bbb35252c749418712
This command was run using /usr/lib/hadoop/hadoop-common-3.2.4.jar
```

Запуск команды `mapred` без аргументов выводит список всех команд MapReduce.

Команды имеют следующий синтаксис:

```
$ mapred [SHELL_OPTIONS] COMMAND [GENERIC_OPTIONS] [COMMAND_OPTIONS]
```

Здесь:

- `SHELL_OPTIONS` — опции Hadoop shell, описанные в разделе Shell Options справки команд Hadoop.
- `GENERIC_OPTIONS` — стандартные опции, поддерживаемые несколькими командами. Дополнительную информацию можно получить в разделе Generic Options справки команд Hadoop.
- `COMMAND_OPTIONS` — опции, специфичные для конкретной команды, описанные в справочных материалах.

Примеры использования

Обычно задачи (jobs) для MapReduce запускается с помощью команд yarn jar или hadoop jar, но вы можете использовать MapReduce CLI, например, для получения информации о конкретных задачах и для контроля их выполнения.

Чтобы увидеть список всех задач, выполните:

```
$ mapred job -list all
```

Пример вывода:

```
Total jobs:13
                  JobId              JobName         State           StartTime      UserName           Queue      Priority
UsedContainers  RsvdContainers      UsedMem         RsvdMem         NeededMem         AM info
 job_1713528166269_0001      QuasiMonteCarlo     SUCCEEDED       1713529720429          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1713528166269_0001/
 job_1713770455913_0001      QuasiMonteCarlo     SUCCEEDED       1713770623388          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1713770455913_0001/
 job_1713770455913_0002      QuasiMonteCarlo     SUCCEEDED       1713771341014          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1713770455913_0002/
 job_1713964485222_0001      QuasiMonteCarlo          PREP       1713965480860          yarn         default       DEFAULT
1               0        2048M              0M             2048M      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1713964485222_0001/
 job_1701175565199_0001      QuasiMonteCarlo     SUCCEEDED       1701175708248          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1701175565199_0001/
 job_1705311887839_0001      QuasiMonteCarlo     SUCCEEDED       1705319626157          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1705311887839_0001/
 job_1705311887839_0002      QuasiMonteCarlo     SUCCEEDED       1705319754517          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1705311887839_0002/
 job_1705573015955_0002      QuasiMonteCarlo     SUCCEEDED       1705658611315          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1705573015955_0002/
 job_1704872668642_0003      QuasiMonteCarlo     SUCCEEDED       1704884498654          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1704872668642_0003/
 job_1712415098046_0001      QuasiMonteCarlo     SUCCEEDED       1712416353017          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1712415098046_0001/
 job_1704872668642_0004      QuasiMonteCarlo     SUCCEEDED       1704884603925          yarn         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1704872668642_0004/
 job_1704872668642_0001               distcp     SUCCEEDED       1704884156346          hdfs         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1704872668642_0001/
 job_1704872668642_0002               distcp     SUCCEEDED       1704884247607          hdfs         default       DEFAULT
N/A             N/A          N/A             N/A               N/A      http://elenas-adh2.ru-central1.
internal:8088/proxy/application_1704872668642_0002/
```

Чтобы увидеть состояние конкретной задачи, выполните:

```
$ mapred job -status <job-ID>
```

Здесь `job-ID` — это идентификатор задачи, по которой запрашивается информация.

Пример вывода:

```
Job: job_1713964485222_0001
Job File: hdfs://adh/user/yarn/.staging/job_1713964485222_0001/job.xml
Job Tracking URL : http://elenas-adh2.ru-central1.internal:8088/proxy/application_1713964485222_0001/
Uber job : false
Number of maps: 16
Number of reduces: 1
map() completion: 0.625
reduce() completion: 0.0
Job state: RUNNING
retired: false
reason for failure:
Counters: 33
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=2306730
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=2536
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=40
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=0
        Job Counters
                Launched map tasks=11
                Data-local map tasks=11
                Total time spent by all maps in occupied slots (ms)=19885
                Total time spent by all map tasks (ms)=19885
                Total vcore-milliseconds taken by all map tasks=19885
                Total megabyte-milliseconds taken by all map tasks=20362240
        Map-Reduce Framework
                Map input records=10
                Map output records=20
                Map output bytes=180
                Map output materialized bytes=280
                Input split bytes=1356
                Combine input records=0
                Spilled Records=20
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=406
                CPU time spent (ms)=3600
                Physical memory (bytes) snapshot=3162824704
                Virtual memory (bytes) snapshot=28058554368
                Total committed heap usage (bytes)=2188902400
                Peak Map Physical memory (bytes)=346021888
                Peak Map Virtual memory (bytes)=2810654720
        File Input Format Counters
                Bytes Read=1180
```

Использование REST API MapReduce

Елена Костюченко

Содержание

Назад к содержанию

REST API Application Master

REST API History Server

Компоненты MapReduce предоставляют REST API для отправки запросов к Application Master и History Server.

Используя MapReduce REST API, можно получить информацию о запущенных и завершенных приложениях, заданиях и задачах. Эту информацию также можно найти в веб-интерфейсе YARN.

Для выполнения некоторых запросов может быть нужен специальный доступ. Вы можете настроить доступы для пользователей вашего кластера в настройках ACL.

REST API Application Master

Application Master принимает запросы, отправленные через прокси. В качестве прокси можно использовать Resource Manager или настроить отдельный хост.

URL запроса может быть в одном из следующих форматов (вывод от этого не изменится):

http://<адрес-прокси>:<порт>/proxy/<ID-приложения>/ws/v1/mapreduce

http://<адрес-прокси>:<порт>/proxy/<ID-приложения>/ws/v1/mapreduce/info

Здесь:

- `<адрес-прокси>` — IP-адрес прокси-сервера;
- `<порт>` — порт прокси-сервера;
- `<ID-приложения>` — идентификатор приложения.

Пример запроса к Application Master:

```
$ curl -L http://127.0.0.1:8088/proxy/application_1713528166269_0001/ws/v1/mapreduce/info -H "Accept: application/xml"
```

Эта команда запрашивает информацию о запущенном приложении с идентификатором `application_1713528166269_0001` в формате XML. Вы можете запросить ответ в формате JSON, передав в заголовке `Accept: application/json`.

Ответ Application Master:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><info><appId>application_1713528166269_0001</appId>
<name>QuasiMonteCarlo</name><user>yarn</user><startedOn>1713529723410</startedOn><elapsedTime>38797</elapsedTime></info>
```

Полная спецификация API доступна в статье MapReduce Application Master REST API’s.

REST API History Server

REST API History Server предоставляет информацию о завершившихся приложениях.

Пример команды, которая возвращает список заданий:

```
$ curl -X GET http://<адрес-history-server>:8042/ws/v1/history/mapreduce/jobs -H "Accept: application/json"
```

Здесь `<адрес-history-server>` — это IP-адрес или полное доменное имя хоста, на котором размещен History Server.

Эта команда запрашивает информацию о выполненных задачах (jobs) в формате JSON. Вы можете запросить ответ в формате XML, передав в заголовке `Accept: application/xml`.

Существуют дополнительные параметры, которые можно использовать для фильтрации вывода:

- `user` — имя пользователя, запустившего приложение;
- `state` — состояние задачи;
- `queue` — имя очереди;
- `limit` — количество объектов в выводе команды;
- `startedTimeBegin` — задачи, которые начали выполняться после указанного времени (в миллисекундах);
- `startedTimeEnd` — задачи, которые начали выполняться до указанного времени (в миллисекундах);
- `finishedTimeBegin` — задачи, которые завершились после указанного времени (в миллисекундах);
- `finishedTimeEnd` — задачи, которые завершились до указанного времени (в миллисекундах).

Пример команды, запрашивающей список задач, которые начались в промежутке времени между `1713770628450` и `1713770680980`:

```
$ curl -X GET http://127.0.0.1:8088:19888/ws/v1/history/mapreduce/jobs?startedTimeBegin=1713770628450&startedTimeEnd=1713770680980
-H "Accept: application/json"
```

Ответ сервера:

```
{"jobs":{"job":
[{"submitTime":1713770623388,"startTime":1713770628457,"finishTime":1713770680975,"id":"job_1713770455913_0001","name":"QuasiMont
eCarlo","queue":"default","user":"yarn","state":"SUCCEEDED","mapsTotal":16,"mapsCompleted":16,"reducesTotal":1,"reducesCompleted"
:1},
{"submitTime":1713771341013,"startTime":1713771345766,"finishTime":1713771398188,"id":"job_1713770455913_0002","name":"QuasiMonte
Carlo","queue":"default","user":"yarn","state":"SUCCEEDED","mapsTotal":16,"mapsCompleted":16,"reducesTotal":1,"reducesCompleted":
1}]}}
```

Полная спецификация API доступна в статье MapReduce History Server REST API’s.

Логирование в MapReduce

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
- Поиск по логам
- Уровни логирования
- Логи контейнеров
- Архивация логов

Обзор

MapReduce записывает выводы команд и события процессов в текстовые логи. Эта информация может быть полезна при диагностике технических проблем.

Каждый компонент MapReduce хранит логи локально, на хосте, на котором он установлен. Логи имеют расширение *.log* и расположены в директории *var/log/hadoop-mapreduce/*. Здесь же находятся файлы *.out*, содержащие информацию о запуске компонентов.

Просмотреть логи MapReduce и других компонентов YARN в браузере можно через веб-интерфейсы YARN.

Имена файлов с логами имеют следующий формат: `hadoop-mapred-<компонент>-<хост>.log`.

Здесь:

- `<компонент>` — имя компонента, например, History Server;
- `<хост>` — FQDN хоста компонента.

MapReduce и HDFS используют один файл log4j.properties для управления логированием.

Поиск по логам

Чтобы найти в логах интересующую вас информацию, например, сообщения об ошибках, подключитесь к хосту с компонентом, логи которого вы хотите просмотреть, и воспользуйтесь командой `grep`.

Например:

```
$ cat /var/log/hadoop-mapreduce/hadoop-mapred-historyserver-elenas-adh2.ru-central1.internal.log | grep -i -A3 -B1 error
```

Эта команда сканирует лог компонента History Server, расположенного на хосте `elenas-adh2.ru-central1.internal`, и ищет сообщения, содержащие слово `error`. Опция `-i` позволяет игнорировать различия в регистре, а опции `-A3 -B1` добавляют к найденной строке с ошибкой одну строчку до нее и три строчки после.

Пример вывода команды:

```
2024-04-20 08:53:57,337 INFO org.apache.hadoop.mapreduce.v2.hs.JobHistory: Starting scan to move intermediate done files
2024-04-20 08:55:59,664 ERROR org.apache.hadoop.mapreduce.v2.hs.JobHistoryServer: RECEIVED SIGNAL 15: SIGTERM
2024-04-20 08:55:59,668 INFO org.apache.hadoop.metrics2.impl.MetricsSystemImpl: Stopping JobHistoryServer metrics system...
2024-04-20 08:55:59,668 INFO org.apache.hadoop.metrics2.impl.MetricsSystemImpl: JobHistoryServer metrics system stopped.
2024-04-20 08:55:59,669 INFO org.apache.hadoop.metrics2.impl.MetricsSystemImpl: JobHistoryServer metrics system shutdown complete.
```

Уровни логирования

MapReduce использует для создания логов библиотеку Log4j2, которая поддерживает следующие уровни логирования (от наименее информативного к наиболее информативному):

1. `FATAL` — сообщает, что операция не может быть выполнена и будет завершена.
2. `ERROR` — уведомляет, что программа работает неправильно или прекратила работу.
3. `WARN` — предупреждает о потенциальных проблемах. Это означает, что программа работает не по стандартному сценарию и в будущем могут возникнуть проблемы.
4. `INFO` — передает информацию о жизненном цикле или состоянии программы.
5. `DEBUG` — выводит отладочную информацию о внутренних состояниях программы.
6. `TRACE` — выводит сообщения, отслеживающие ход выполнения программы.

Logger Log4j также принимает значения уровня логирования: `OFF` — для отключения логирования, и `ALL` — для разрешения всех типов сообщений.

Включение одного уровня логирования включит этот уровень и все уровни до него. Например, если установлен уровень логирования `WARN`, то в логи будут попадать только события `WARN`, `ERROR` и `FATAL`, но не `INFO`, `DEBUG` и `TRACE`.

Логи контейнеров

Приложения, запущенные в YARN, генерируют собственные логи. Чтобы посмотреть логи приложения, используйте команду logs.

Например:

```
$ yarn logs -applicationId application_1714649647710_0001
```

Архивация логов

Логи MapReduce можно архивировать с помощью команды archive-logs. Этот процесс аналогичен функции агрегирования логов в YARN.

Пример команды для создания архива из десяти log-файлов:

```
$ mapred archive-logs -minNumberLogFiles 10
```

Значение по умолчанию для `-minNumberLogFiles` — 20. Это значит, что запуск команды `mapred archive-logs` создаст архив только в том случае, если существует 20 или более логов MapReduce.

Настройка сортировки в MapReduce

Елена Костюченко

Назад к содержанию

Hadoop позволяет настроить использование кастомных реализаций методов сортировки (sort) и перемешивания (shuffle) в MapReduce.

Это позволяет, например, использовать протокол RDMA вместо HTTP для перемешивания данных или использовать другие алгоритмы сортировки, которые позволяют агрегировать хеш и выполнять запросы Limit-N.

Дополнительную информацию о MapReduce можно получить в статьях Обзор работы MapReduce и Архитектура MapReduce.

ВНИМАНИЕ

Возможность использования кастомных решений для перемешивания и сортировки — экспериментальная и нестабильная. Это значит, что предоставляемые API могут измениться и нарушить совместимость в будущих версиях Hadoop.

Перемешивание выполняется сервисами NodeManager Auxiliary service и shuffle consumer plugin.

Чтобы заменить auxiliary service по умолчанию, необходимо отредактировать свойства конфигурации NodeManager, перечисленные в таблице ниже.

Параметры NodeManager Auxiliary service

Параметр	Описание	Значение по умолчанию
yarn.nodemanager.aux-services	Название нового auxiliary service	...,mapreduce_shuffle
yarn.nodemanager.aux-services.mapreduce_shuffle.class	Имя класса auxiliary service	org.apache.hadoop.mapred.ShuffleHandler
yarn.nodemanager.aux-services.<имя_класса>.classpath	Путь к локальной директории с JAR-файлами auxiliary service и его зависимостей. Путь может вести к конкретному JAR-файлу, использовать паттерн `{local_dir_to_jar}/*` для загрузки всех JAR-файлов из директории или содержать несколько значений, разделенных `:`	—
yarn.nodemanager.aux-services.<имя_класса>.remote-classpath	Параметр, который следует использовать вместо `yarn.nodemanager.aux-services.%s.classpath`, если файлы JAR расположены в удаленной файловой системе	—

Чтобы отредактировать эти параметры на всех хостах с помощью ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите в ***Services*** и нажмите на ***YARN***.
3. Включите параметр ***Show Advanced*** и найдите ***yarn-site.xml***.
4. Откройте раскрывающийся список параметров, выберите ***yarn.nodemanager.aux-services*** и добавьте новый параметр.
5. Выберите ***yarn.nodemanager.aux-services.mapreduce_shuffle.class*** и введите новое значение.
6. Найдите параметр ***Custom yarn-site.xml***, выберите ***Add property*** и введите имя нового параметра в зависимости от расположения JAR-файлов auxiliary service: `yarn.nodemanager.aux-services.<имя_класса>.classpath` или `yarn.nodemanager.aux-services.<имя_класса>.remote-classpath`.
7. Подтвердите изменения в конфигурации YARN, нажав ***Save***.
8. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Чтобы заменить shuffle consumer plugin по умолчанию, установите новое значение параметра `mapreduce.job.reduce.shuffle.consumer.plugin.class`.

Если новое значение задано в переменной окружения на одном хосте, настройки shuffle consumer plugin изменятся для отдельной задачи MapReduce (job). Чтобы использовать один shuffle consumer plugin для всех задач, отредактируйте значение этого параметра в файле конфигурации *mapred-site.xml* с помощью ADCM.

ПРИМЕЧАНИЕ

Если использовать кастомные auxiliary service и shuffle consumer plugin, необходимо добавить новый сервисный ключ в параметр `yarn.nodemanager.aux-services.<…>`. Например, для сервиса с названием `mapred.shufflex` необходимо добавить параметр `yarn.nodemanager.aux-services.mapreduce_shufflex.class`.

Чтобы изменить метод сортировки по умолчанию, установите новое значение для свойства `mapreduce.job.map.output.collector.class` коллектора вывода (output collector).

В значении параметра может быть указан разделенный запятыми список коллекторов вывода. В этом случае задача map будет пытаться создать экземпляр каждого из них по очереди, пока одна из реализаций не будет успешно инициализирована. Это может быть полезно в случаях, когда отдельная реализация коллектора совместима только с определенными типами ключей или значений.

Если новое значение задано в переменной окружения на одном хосте, настройки коллектора изменятся для отдельной задачи MapReduce (job). Чтобы использовать новый коллектор вывода для всех задач, отредактируйте значение этого параметра в файле конфигурации *mapred-site.xml* с помощью ADCM.

Чтобы отредактировать *mapred-site.xml* на всех хостах через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите в ***Services*** и нажмите на ***YARN***.
3. Включите параметр ***Show Advanced*** и найдите ***Custom mapred-site.xml***.
4. Откройте раскрывающийся список параметров и выберите ***Add property***.
5. Введите имя параметра и желаемое значение.
6. Подтвердите изменения в конфигурации YARN, нажав ***Save***.
7. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Encrypted shuffle в MapReduce

Елена Костюченко

Содержание

Назад к содержанию

Обзор

Конфигурация MapReduce

Конфигурация YARN

Настройки keystore и truststore

Активация

Обзор

Hadoop позволяет добавить шифрование данных на этапе перемешивания (shuffle) в MapReduce и YARN. Эта функция использует HTTPS с дополнительной аутентификацией клиента, также известной как двунаправленный HTTPS или HTTPS с клиентскими сертификатами.

ВНИМАНИЕ

Использование encrypted shuffle существенно влияет на производительность кластера. Рекомендуется добавить дополнительные ресурсы для поддержки шифрования данных.

Функция encrypted shuffle включает в себя несколько дополнительных настроек безопасности кластера:

- Параметры конфигурации для переключения между HTTP и HTTPS.
- Параметры конфигурации, определяющие свойства хранилища ключей (keystore) и хранилища доверенных сертификатов (truststore): их расположение, тип, пароли.
- Параметр перезагрузки хранилищ доверенных сертификатов в кластере при добавлении или удалении ноды.

Чтобы включить encrypted shuffle для MapReduce и YARN, необходимо обновить их файлы конфигурации и добавить информацию о сертификате SSL в настройках хранилища ключей и хранилища доверенных сертификатов.

Файлы конфигурации MapReduce и YARN (*core-site.xml* и *mapred-site.xml*) можно отредактировать с помощью ADCM. Настройки хранилищ ключей и сертификатов необходимо обновить вручную.

Клиентские сертификаты

Файл хранилища ключей сертификата клиента, содержащий закрытый ключ (private key), должен быть доступен всем пользователям, создающим задачи в кластере. Это означает, что любой пользователь может получить доступ к файлам хранилища ключей и использовать клиентские сертификаты для установки соединения с shuffle-сервером.

Лучшей защитой данных является механизм JobToken, используемый в Hadoop. Каждое задание использует свой уникальный JobToken для получения только тех данных, к которым разрешен доступ. Без действительного JobToken невозможно получить доступ ко всем данным на сервере.

Если использование клиентского сертификата необходимо, убедитесь, что браузеры, подключающиеся к веб-интерфейсам, используют сертификаты с соответствующей подписью. Если сертификаты подписаны центром сертификации (Certificate Authority, CA), убедитесь, что полная цепочка сертификатов включена в хранилище ключей сервера.

Конфигурация MapReduce

Чтобы настроить encrypted shuffle для MapReduce через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите в ***Services*** и нажмите на ***HDFS***.
3. Включите параметр ***Show Advanced*** и найдите ***core-site.xml***.
4. Откройте раскрывающийся список параметров, выберите и отредактируйте необходимые свойства из таблицы ниже.
5. Подтвердите изменения в конфигурации кластера, нажав ***Save***.

Свойства encrypted shuffle

Параметр	Описание	Значение по умолчанию
hadoop.ssl.require.client.cert	Логическое значение, указывающее, используются ли клиентские сертификаты	false
hadoop.ssl.hostname.verifier	Используемый тип проверки имени хоста. Принимает следующие значения: `DEFAULT`, `STRICT`, `STRICT_IE6`, `DEFAULT_AND_LOCALHOST` или `ALLOW_ALL`	DEFAULT
hadoop.ssl.keystores.factory.class	Используемая реализация KeyStoresFactory. В настоящее время доступна только реализация по умолчанию. Для нее используются свойства, расположенные в файлах *ssl-server.xml* и *ssl-client.xml*	org.apache.hadoop.security.ssl.FileBasedKeyStoresFactory
hadoop.ssl.server.conf	Файл ресурсов с информацией о хранилище ключей SSL-сервера	ssl-server.xml
hadoop.ssl.client.conf	Файл ресурсов с информацией о хранилище ключей SSL-клиента	ssl-client.xml
hadoop.ssl.enabled.protocols	Поддерживаемые протоколы SSL. Этот параметр используется только HTTP-сервером DataNode	TLSv1.2

Параметр `hadoop.ssl.hostname.verifier` поддерживает следующие типы проверки:

- `DEFAULT` — проверяется соответствие имени хоста первому общему имени (CN) или любому из альтернативных имен (SAN). Имена могут использовать маску (wildcard), например, имя хоста **.example.com* будет соответствовать всем субдоменам, включая *beta.test.example.com*.
- `DEFAULT_AND_LOCALHOST` — проверка работает так же, как `DEFAULT`, но позволяет использовать все имена хостов типа: `localhost`, `localhost.example` или `127.0.0.1`.
- `STRICT` — проверка работает так же, как `DEFAULT`, но допускает использование маски (wildcard) только для доменов того же уровня. Например, имя хоста **.example.com* будет соответствовать только субдоменам того же уровня, например *test.example.com*, но не *beta.test.example.com*.
- `STRICT_IE6` — проверка работает так же, как `STRICT`, но разрешает имена хостов, которые соответствуют любому из общих имен (CN) в сертификате сервера X.509.
- `ALLOW_ALL` — отключает механизм проверки имени хоста.

Конфигурация YARN

Настроенная конфигурация для MapReduce позволяет также включить encrypted shuffle для YARN.

Чтобы настроить encrypted shuffle для YARN через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите в ***Services*** и нажмите на ***YARN***.
3. Включите параметр ***Show advanced***.
4. Найдите параметр ***mapreduce.shuffle.ssl.enabled*** и установите для него значение `true`.
5. Подтвердите изменения в конфигурации YARN, нажав ***Save***.

Настройки keystore и truststore

Для включения encrypted shuffle необходимо изменить настройки хранилища ключей и хранилища доверенных сертификатов, используемые сервисом MapReduce shuffle.

Эти настройки расположены в файлах *ssl-server.xml* и *ssl-client.xml* в директории *etc/hadoop/conf/* на хостах кластера.

После редактирования свойств убедитесь, что пользователь `mapred`:

- является владельцем файлов *ssl-server.xml* и *ssl-client.xml*;
- имеет эксклюзивный доступ на чтение файла конфигурации SSL-сервера;
- имеет разрешения по умолчанию для файла конфигурации клиента SSL.

Свойства SSL-клиента и сервера

Параметр клиента	Параметр сервера	Описание	Значение по умолчанию
ssl.client.keystore.type	ssl.server.keystore.type	Тип файла хранилища ключей	jks
ssl.client.keystore.location	ssl.server.keystore.location	Местоположение файла хранилища ключей. Пользователь `mapred` должен быть владельцем этого файла и иметь эксклюзивный доступ к нему на чтение	—
ssl.client.keystore.password	ssl.server.keystore.password	Пароль к файлу хранилища ключей	—
ssl.client.truststore.type	ssl.server.truststore.type	Тип файла хранилища доверенных сертификатов	jks
ssl.client.truststore.location	ssl.server.truststore.location	Расположение файла хранилища доверенных сертификатов. Пользователь `mapred` должен быть владельцем этого файла и иметь эксклюзивный доступ к нему на чтение	—
ssl.client.truststore.password	ssl.server.truststore.password	Пароль к файлу хранилища доверенных сертификатов	—
ssl.client.truststore.reload.interval	ssl.server.truststore.reload.interval	Определяет, как часто хранилища доверенных сертификатов перезагружают свою конфигурацию (в миллисекундах)	10000

Если вы скопируете новый файл хранилища доверенных сертификатов поверх старого, система заменит старые сертификаты новыми.

Эта функция полезна для добавления или удаления нод из кластера или доверенных клиентов. В таких случаях сертификат клиента или NodeManager добавляется (или удаляется) из всех файлов хранилища доверенных сертификатов в системе. Новая конфигурация будет реализована без необходимости перезапуска NodeManager.

Файл ssl-client.xml по умолчанию

```
<?xml version="1.0"?>
<configuration>
        <property>
                <name>ssl.client.keystore.type</name>
                <value>jks</value>
        </property>
        <property>
                <name>ssl.client.truststore.reload.interval</name>
                <value>10000</value>
        </property>
        <property>
                <name>ssl.client.truststore.type</name>
                <value>jks</value>
        </property>
</configuration>
```

Файл ssl-server.xml по умолчанию

```
<?xml version="1.0"?>
<configuration>
        <property>
                <name>ssl.server.keystore.type</name>
                <value>jks</value>
        </property>
        <property>
                <name>ssl.server.truststore.reload.interval</name>
                <value>10000</value>
        </property>
        <property>
                <name>ssl.server.truststore.type</name>
                <value>jks</value>
        </property>
</configuration>
```

Активация

После того, как все необходимые параметры конфигурации заданы, активируйте encrypted shuffle, перезапустив все NodeManager через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите в ***Services***.
3. Раскройте меню ***Actions*** для ***YARN*** и выберите ***Restart***
4. Убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Управление сервисом Monitoring через ADCM

Константин Алпашкин

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Monitoring. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Monitoring, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Monitoring
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Reconfigure and restart	Обновляет конфигурацию и выполняет рестарт Monitoring-сервиса
Update scrape config files	Обновляет содержимое scrape-файлов Prometheus, в которых указаны эндпойнты ADH-сервисов для получения метрик. Данное действие (либо перезапуск всего Monitoring-сервиса) следует выполнять при добавлении новых сервисов ADH или перераспределении сервисных компонентов

Подключение к Ozone через CLI

Сергей Тихомиров

Содержание

Назад к содержанию

Обзор
Примеры
- Работа с томами
- Работа с корзинами
- Управление списками контроля доступа

Обзор

Ozone предоставляет CLI-инструмент, позволяющий получить доступ к некоторым функциям, которые нельзя выполнить другими способами. Обычно такие команды решают одноразовые административные задачи во время настройки, например, создание тома (volume) или работа с ACL.

Общий вид команды Ozone shell представлен ниже:

```
$ ozone sh <object> <action> [params]
```

где:

- `<object>` — объект Ozone, например, корзина (bucket), ключ, префикс, снепшот, пользователь, токен или том.
- `<action>` — действие, которое можно произвести с объектом. Например, для создания тома соответствующим действием является `create`.
- `[params]` — дополнительные параметры, которые необходимы для выполнения некоторых действий.

CLI-инструмент предоставляет руководство как для объектов, так и для действий. Чтобы увидеть список действий, доступных для объекта, выполните команду:

```
$ ozone sh <object> --help
```

А чтобы увидеть параметры, необходимые для выполнения действия, запустите команду:

```
$ ozone sh <object> <action> --help
```

ПРИМЕЧАНИЕ

На данный момент нет онлайн-документации для объектов и действий, доступных в Ozone shell. Для получения списка доступных опций используйте руководство внутри самого инструмента.

Примеры

Работа с томами

CLI включает в себя некоторые уникальные функции, среди которых — создание томов с квотами. Чтобы воспользоваться данной функцией, запустите следующую команду:

```
$ ozone sh volume create <volume_name> --quota=<quota_in_bytes>
```

Чтобы проверить, что том создался правильно с нужной квотой, выведите список всех томов:

```
$ ozone sh volume ls
```

Эта команда возвращает информацию обо всех томах:

```
[ {
  "metadata" : { },
  "name" : "vol1",
  "admin" : "sergei",
  "owner" : "sergei",
  "quotaInBytes" : 1073741824,
  "quotaInNamespace" : -1,
  "usedNamespace" : 0,
  "creationTime" : "2025-01-13T13:20:43.173Z",
  "modificationTime" : "2025-01-13T13:20:43.173Z",
  "acls" : [ {
    "type" : "USER",
    "name" : "sergei",
    "aclScope" : "ACCESS",
    "aclList" : [ "ALL" ]
  }, {
    "type" : "GROUP",
    "name" : "sergei",
    "aclScope" : "ACCESS",
    "aclList" : [ "ALL" ]
  } ],
  "refCount" : 0
} ]
```

Работа с корзинами

Ниже приведен пример создания корзины с ключами шифрования:

```
$ ozone sh bucket create -k <key_name> <URI>
```

где:

- `<key_name>` — название ключа шифрования, созданного при помощи команды `hadoop key create`.
- `<URI>` — название корзины, включающее название тома (например, `/vol1/bucket1`).

Также с помощью CLI можно управлять настройками репликации корзины:

```
$ ozone sh bucket set-replication-config -t=<type> -r=<replication> <value>
```

где:

- `<type>` — тип репликации. Поддерживаемые типы: `RATIS` и `EC`.
- `<replication>` — определение репликации. Валидные значения зависят от типа репликации. Для RATIS подойдут `ONE` или `THREE`. В то время как для EC должен соблюдаться формат `CODEC-DATA-PARITY-CHUNK_SIZE`, например `rs-3-2-1024k`.
- `<value>` — URI корзины.

Ниже приведен пример команды и изменений в конфигурации корзины:

```
$ ozone sh bucket set-replication-config -t=RATIS -r=ONE vol1/bucket1
```

```
"replicationConfig" : {
  "replicationFactor" : "ONE",
  "requiredNodes" : 1,
  "replicationType" : "RATIS"
}
```

Управление списками контроля доступа

CLI-инструмент позволяет добавлять и удалять списки контроля доступа (ACL) для томов и корзин. Ниже приведен пример команды, добавляющей ACL для тома:

```
$ ozone sh volume addacl -s=<URI> -a=<type>:<entity>:<permissions> <value>
```

где:

- `<store>` — тип хранилища: `OZONE` или `S3`.
- `<type>` — вид системной сущности: `user` или `group`.
- `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
- `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
  - `r` — чтение данных из ресурса.
  - `w` — запись данных в ресурс.
  - `c` — создание объектов в ресурсе.
  - `d` — удаление объектов из ресурса.
  - `l` — перечисление объектов, хранящихся в ресурсе.
  - `x` — чтение списка контроля доступа к ресурсу.
  - `y` — изменение списка контроля доступа к ресурсу.
  - `a` — все вышеперечисленное.
  - `n` — никаких прав.
- `<value>` — URI тома/корзины.

Пусть есть следующий том до добавления нового ACL:

```
[ {
  "metadata" : { },
  "name" : "vol1",
  "admin" : "s_tikhomirov_krb1",
  "owner" : "s_tikhomirov_krb1",
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "usedNamespace" : 0,
  "creationTime" : "2025-01-22T21:23:28.796Z",
  "modificationTime" : "2025-01-22T21:23:28.796Z",
  "acls" : [ {
    "type" : "USER",
    "name" : "s_tikhomirov_krb1",
    "aclScope" : "ACCESS",
    "aclList" : [ "ALL" ]
  } ],
  "refCount" : 0
} ]
```

В нем присутствует ACL только для пользователя, создавшего этот том. Вызов команды `addacl` с опцией `-a=user:sergei:rwcdlxy` добавляет следующий ACL:

```
[ {
  "metadata" : { },
  "name" : "vol1",
  "admin" : "s_tikhomirov_krb1",
  "owner" : "s_tikhomirov_krb1",
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "usedNamespace" : 0,
  "creationTime" : "2025-01-22T21:23:28.796Z",
  "modificationTime" : "2025-01-22T21:53:56.260Z",
  "acls" : [ {
    "type" : "USER",
    "name" : "s_tikhomirov_krb1",
    "aclScope" : "ACCESS",
    "aclList" : [ "ALL" ]
  }, {
    "type" : "USER",
    "name" : "sergei",
    "aclScope" : "ACCESS",
    "aclList" : [ "READ", "WRITE", "CREATE", "LIST", "DELETE", "READ_ACL", "WRITE_ACL" ]
  } ],
  "refCount" : 0
} ]
```

Удалить список контроля доступа можно с помощью команды `removeacl`:

```
$ ozone sh volume removeacl -s=OZONE -a=user:sergei:rwcdlxy vol1
```

Обзор веб-интерфейсов Ozone

Владимир Адаменков

Содержание

Назад к содержанию

Веб-интерфейс Ozone Datanode
  - Overview
  - Metrics
  - Configuration
  - Documentation
  - Common tools

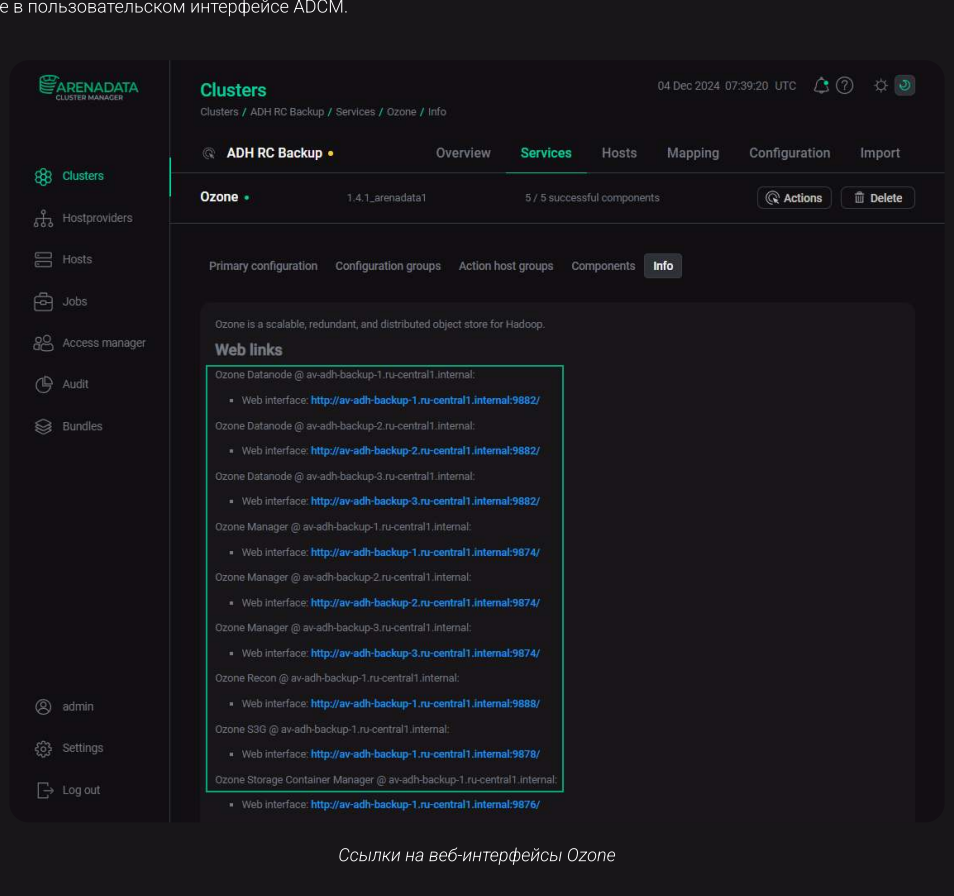
Веб-интерфейс Ozone Manager
  - Overview
  - Metrics
  - Configuration
  - Documentation
  - Common tools

Ozone Recon UI
  - Overview
  - Volumes
  - Buckets
  - Datanodes
  - Pipelines
  - Containers
  - Insights
  - Disk Usage

Веб-интерфейс Ozone S3G

Веб-интерфейс Ozone Storage Container Manager
  - Overview
  - Metrics
  - Configuration
  - Documentation
  - Common tools

Каждый компонент Ozone имеет собственный веб-интерфейс, отображающий диагностическую информацию и информацию о состоянии данных, хранящихся в Ozone. Если компонент распределен по нескольким хостам, то у каждого из этих хостов будет веб-интерфейс для этого компонента. Ссылки на эти веб-интерфейсы можно найти на вкладке ***Info*** сервиса Ozone в пользовательском интерфейсе ADCM.



Ozone is a scalable, redundant, and distributed object store for Hadoop.

Web links

- Ozone Datanode @ av-adh-backup-1.ru-central1.internal:
  - Web interface: http://av-adh-backup-1.ru-central1.internal:9882/
- Ozone Datanode @ av-adh-backup-2.ru-central1.internal:
  - Web interface: http://av-adh-backup-2.ru-central1.internal:9882/
- Ozone Datanode @ av-adh-backup-3.ru-central1.internal:
  - Web interface: http://av-adh-backup-3.ru-central1.internal:9882/
- Ozone Manager @ av-adh-backup-1.ru-central1.internal:
  - Web interface: http://av-adh-backup-1.ru-central1.internal:9874/
- Ozone Manager @ av-adh-backup-2.ru-central1.internal:
  - Web interface: http://av-adh-backup-2.ru-central1.internal:9874/
- Ozone Manager @ av-adh-backup-3.ru-central1.internal:
  - Web interface: http://av-adh-backup-3.ru-central1.internal:9874/
- Ozone Recon @ av-adh-backup-1.ru-central1.internal:
  - Web interface: http://av-adh-backup-1.ru-central1.internal:9888/
- Ozone S3G @ av-adh-backup-1.ru-central1.internal:
  - Web interface: http://av-adh-backup-1.ru-central1.internal:9878/
- Ozone Storage Container Manager @ av-adh-backup-1.ru-central1.internal:
  - Web interface: http://av-adh-backup-1.ru-central1.internal:9876/

Ссылки на веб-интерфейсы Ozone

URL веб-интерфейса компонента Ozone имеет следующую форму: `http://<node-hostname>:<component-port>/`.

Пример: `http://adh-host1.ru-central1.internal:9876/`.

Для защищённых кластеров используйте `https://` вместо `http://`.

Порты веб-интерфейсов, используемые по умолчанию, перечислены в таблице ниже.

Порты веб-интерфейсов компонентов Ozone

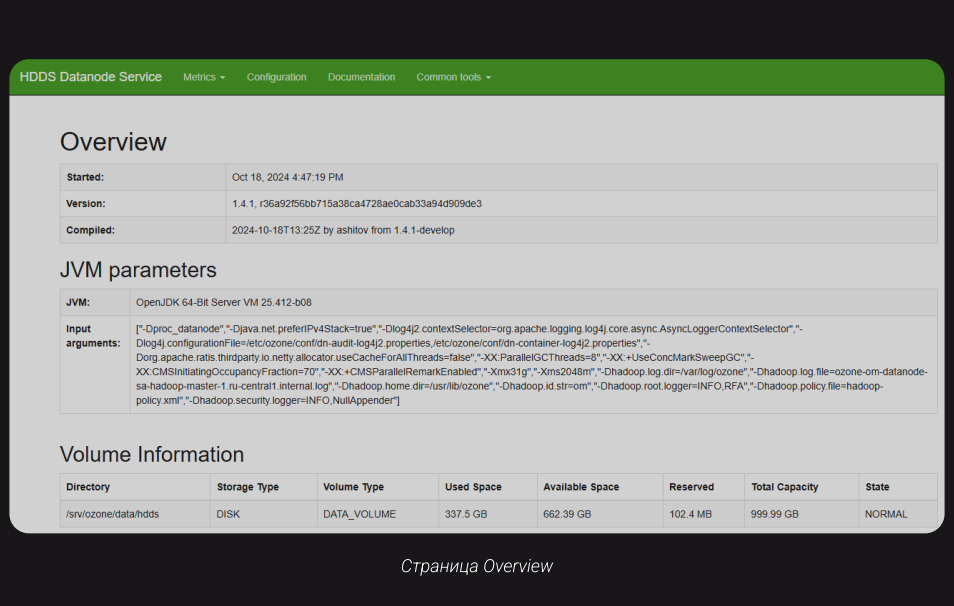
Компонент	Порт
Ozone Datanode	9880
Ozone Manager	9874
Ozone Recon	9888
Ozone S3G	9878
Ozone Storage Container Manager	9876

Веб-интерфейс Ozone Datanode

Overview

Начальной страницей веб-интерфейса Ozone Datanode является страница ***Overview***. Она содержит следующую информацию:

- Информация о сервисе:
  - время запуска;
  - версия и контрольная сумма;
  - время компиляции, пользователь и исходная ветка.
- Информация о виртуальной машине Java: версия и строка входных аргументов.
- Информация о томе:
  - путь к директории тома;
  - тип хранилища;
  - тип тома;
  - статистика использования дискового пространства;
  - состояние тома.



Overview

Started:	Oct 18, 2024 4:47:19 PM
Version:	1.4.1, r36a92f56bb715a38ca4728ae0cab33a94d909de3
Compiled:	2024-10-18T13:25Z by ashitov from 1.4.1-develop

JVM parameters

JVM:	OpenJDK 64-Bit Server VM 25.412-b08
Input arguments:	["-Dproc_datanode","-Djava.net.preferIPv4Stack=true","-Dlog4j2.contextSelector=org.apache.logging.log4j.core.async.AsyncLoggerContextSelector","-Dlog4j.configurationFile=/etc/ozone/conf/dn-audit-log4j2.properties,/etc/ozone/conf/dn-container-log4j2.properties","-Dorg.apache.ratis.thirdparty.io.netty.allocator.useCacheForAllThreads=false","-XX:ParallelGCThreads=8","-XX:+UseConcMarkSweepGC","-XX:CMSInitiatingOccupancyFraction=70","-XX:+CMSParallelRemarkEnabled","-Xmx31g","-Xms2048m","-Dhadoop.log.dir=/var/log/ozone","-Dhadoop.log.file=ozone-om-datanode-sa-hadoop-master-1.ru-central1.internal.log","-Dhadoop.home.dir=/usr/lib/ozone","-Dhadoop.id.str=om","-Dhadoop.root.logger=INFO,RFA","-Dhadoop.policy.file=hadoop-policy.xml","-Dhadoop.security.logger=INFO,NullAppender"]

Volume Information

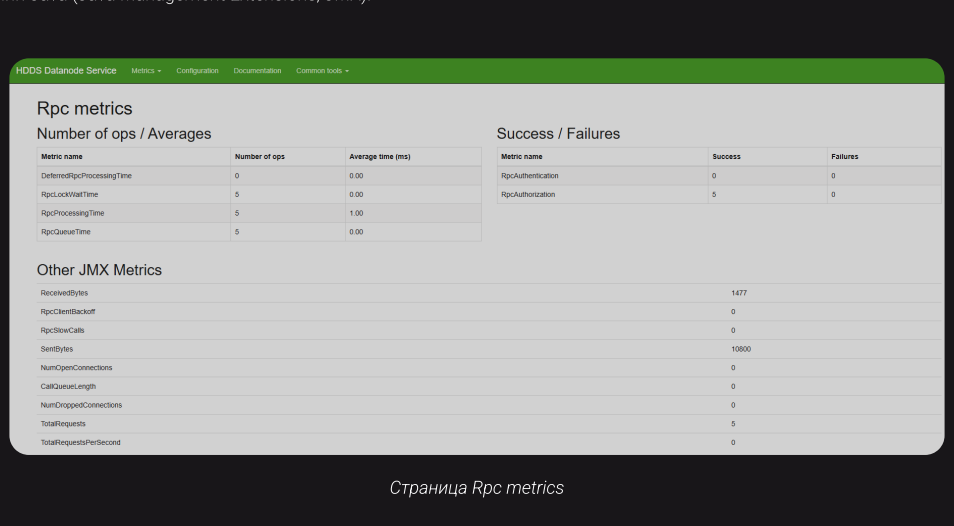
Directory	Storage Type	Volume Type	Used Space	Available Space	Reserved	Total Capacity	State
/srv/ozone/data/hdds	DISK	DATA_VOLUME	337.5 GB	662.39 GB	102.4 MB	999.99 GB	NORMAL

Страница Overview

Metrics

Группа ***Metrics*** содержит страницу ***Rpc metrics***. Эта страница содержит следующую информацию:

- Количество выполненных операций и среднее время выполнения каждой операции.
- Статистика успешных и неудачных событий по метрикам RPC-аутентификации и авторизации.
- Другие метрики расширений Java (Java Management Extensions, JMX).



Rpc metrics

Number of ops / Averages

Metric name	Number of ops	Average time (ms)
DeferredRpcProcessingTime	0	0.00
RpcLockWaitTime	5	0.00
RpcProcessingTime	5	1.00
RpcQueueTime	5	0.00

Success / Failures

Metric name	Success	Failures
RpcAuthentication	0	0
RpcAuthorization	5	0

Other JMX Metrics

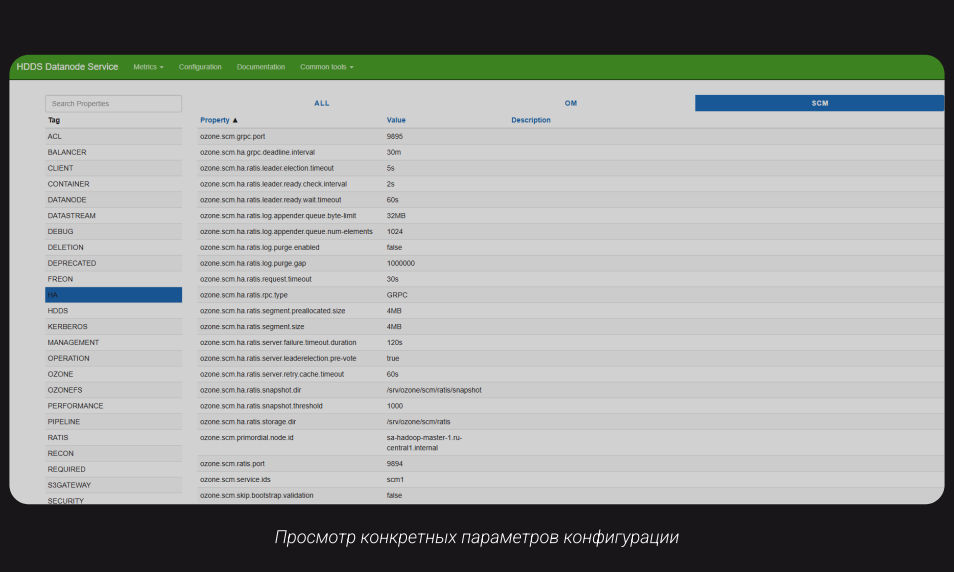
ReceivedBytes	1477
RpcClientBackoff	0
RpcSlowCalls	0
SentBytes	10800
NumOpenConnections	0
CallQueueLength	0
NumDroppedConnections	0
TotalRequests	5
TotalRequestsPerSecond	0

Страница Rpc metrics

Configuration

Страница ***Configuration*** содержит список всех параметров конфигурации, относящихся к работе сервиса Ozone. С помощью поля ***Search Properties*** можно искать параметры по их имени или значению. Также можно фильтровать список параметров, выделяя один или несколько тегов, перечисленных в левой части экрана. Для просмотра параметров Ozone Manager нажмите ***OM*** в верхней части экрана, а для просмотра параметров Storage Container Manager нажмите ***SCM***. Кроме того, список параметров можно сортировать по их именам и значениям.

Например, для просмотра значений параметров конфигурации, относящихся к режиму высокой доступности в компоненте Storage Container Manager, выделите тег ***HA*** и выберите вкладку ***SCM***.

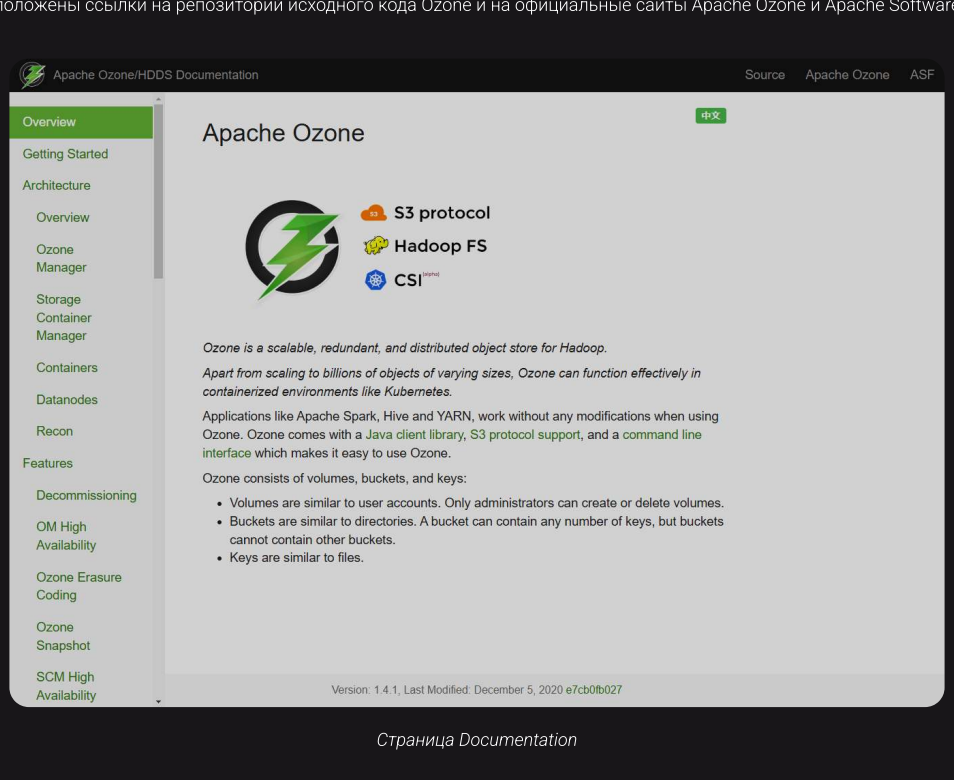


Property ▲	Value	Description
ozone.scm.grpc.port	9895	
ozone.scm.ha.grpc.deadline.interval	30m	
ozone.scm.ha.ratis.leader.election.timeout	5s	
ozone.scm.ha.ratis.leader.ready.check.interval	2s	
ozone.scm.ha.ratis.leader.ready.wait.timeout	60s	
ozone.scm.ha.ratis.log.appender.queue.byte-limit	32MB	
ozone.scm.ha.ratis.log.appender.queue.num-elements	1024	
ozone.scm.ha.ratis.log.purge.enabled	false	
ozone.scm.ha.ratis.log.purge.gap	1000000	
ozone.scm.ha.ratis.request.timeout	30s	
ozone.scm.ha.ratis.rpc.type	GRPC	
ozone.scm.ha.ratis.segment.preallocated.size	4MB	
ozone.scm.ha.ratis.segment.size	4MB	
ozone.scm.ha.ratis.server.failure.timeout.duration	120s	
ozone.scm.ha.ratis.server.leaderelection.pre-vote	true	
ozone.scm.ha.ratis.server.retry.cache.timeout	60s	
ozone.scm.ha.ratis.snapshot.dir	/srv/ozone/scm/ratis/snapshot	
ozone.scm.ha.ratis.snapshot.threshold	1000	
ozone.scm.ha.ratis.storage.dir	/srv/ozone/scm/ratis	
ozone.scm.primordial.node.id	sa-hadoop-master-1.ru-central1.internal	
ozone.scm.ratis.port	9894	
ozone.scm.service.ids	scm1	
ozone.scm.skip.bootstrap.validation	false	

Просмотр конкретных параметров конфигурации

Documentation

Страница ***Documentation*** содержит руководство по Apache Ozone. Оно доступно на английском и китайском языках — переключатель языков расположен в правом верхнем углу. Также там расположены ссылки на репозиторий исходного кода Ozone и на официальные сайты Apache Ozone и Apache Software Foudation.



Apache Ozone

Ozone is a scalable, redundant, and distributed object store for Hadoop.

Apart from scaling to billions of objects of varying sizes, Ozone can function effectively in containerized environments like Kubernetes.

Applications like Apache Spark, Hive and YARN, work without any modifications when using Ozone. Ozone comes with a Java client library, S3 protocol support, and a command line interface which makes it easy to use Ozone.

Ozone consists of volumes, buckets, and keys:

- Volumes are similar to user accounts. Only administrators can create or delete volumes.
- Buckets are similar to directories. A bucket can contain any number of keys, but buckets cannot contain other buckets.
- Keys are similar to files.

Version: 1.4.1, Last Modified: December 5, 2020 e7cb0fb027

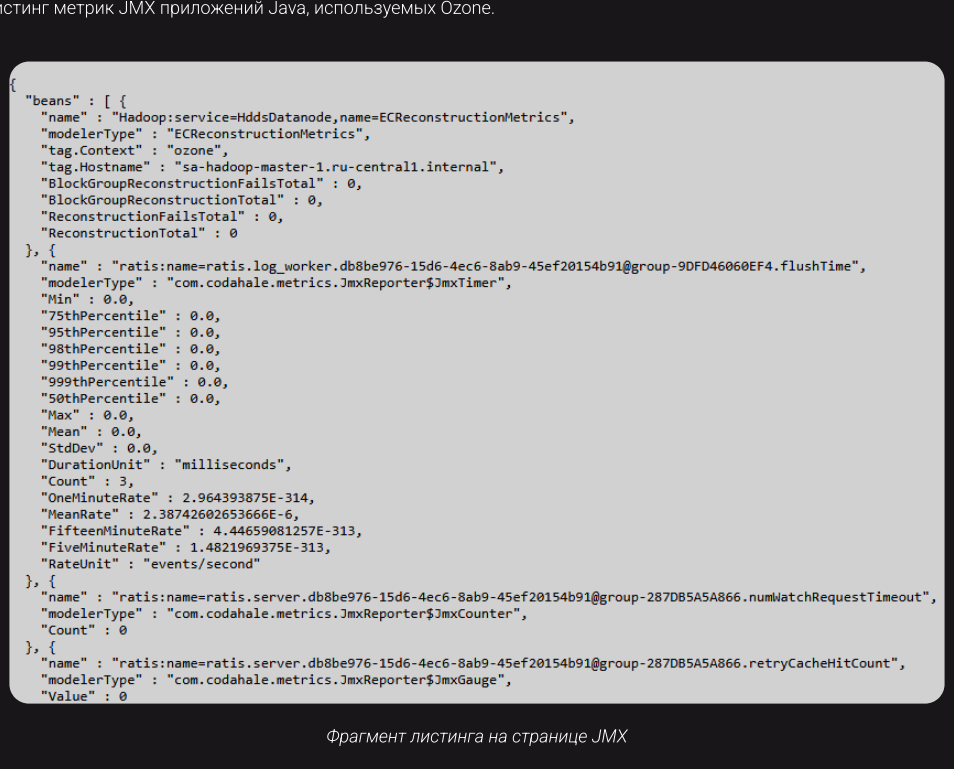
Страница Documentation

Common tools

Группа ***Common tools*** содержит страницы ***JMX***, ***Config***, ***Stacks*** и ***Log levels***.

JMX

Страница ***JMX*** отображает листинг метрик JMX приложений Java, используемых Ozone.

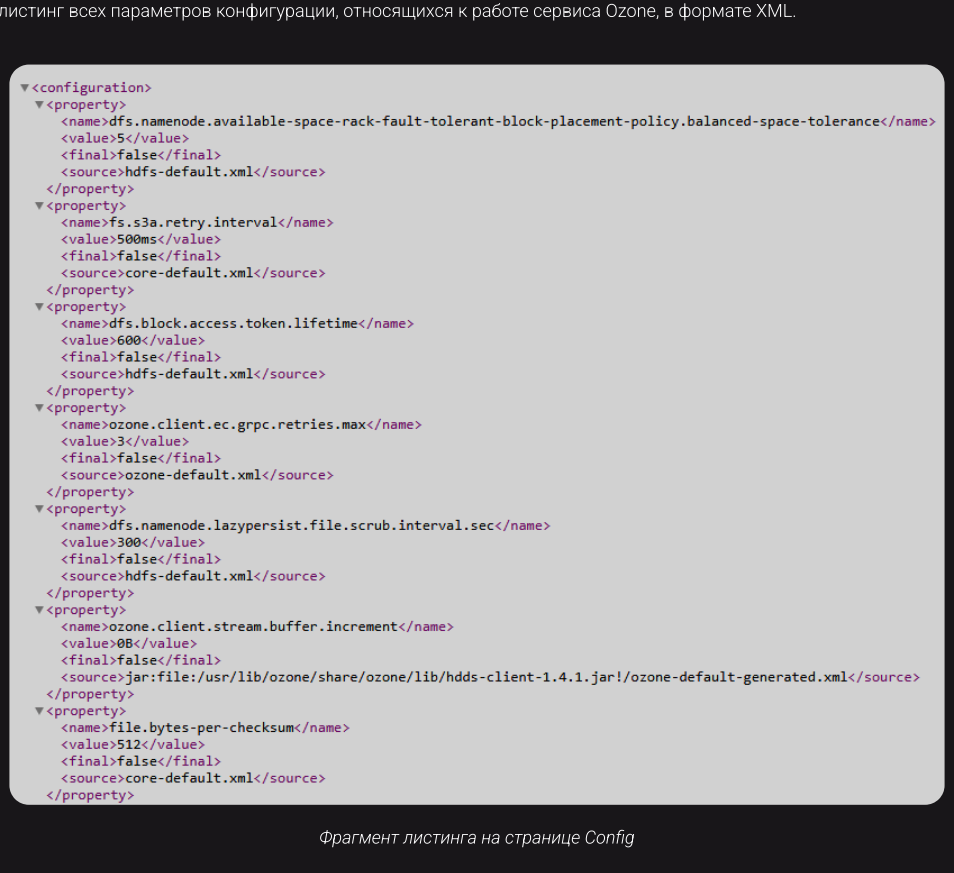


```
{
  "beans" : [ {
    "name" : "Hadoop:service=HddsDatanode,name=ECReconstructionMetrics",
    "modelerType" : "ECReconstructionMetrics",
    "tag.Context" : "ozone",
    "tag.Hostname" : "sa-hadoop-master-1.ru-central1.internal",
    "BlockGroupReconstructionFailsTotal" : 0,
    "BlockGroupReconstructionTotal" : 0,
    "ReconstructionFailsTotal" : 0,
    "ReconstructionTotal" : 0
  }, {
    "name" : "ratis:name=ratis.log_worker.db8be976-15d6-4ec6-8ab9-45ef20154b91@group-9DFD46060EF4.flushTime",
    "modelerType" : "com.codahale.metrics.JmxReporter$JmxTimer",
    "Min" : 0.0,
    "75thPercentile" : 0.0,
    "95thPercentile" : 0.0,
    "98thPercentile" : 0.0,
    "99thPercentile" : 0.0,
    "999thPercentile" : 0.0,
    "50thPercentile" : 0.0,
    "Max" : 0.0,
    "Mean" : 0.0,
    "StdDev" : 0.0,
    "DurationUnit" : "milliseconds",
    "Count" : 3,
    "OneMinuteRate" : 2.964393875E-314,
    "MeanRate" : 2.38742602653666E-6,
    "FifteenMinuteRate" : 4.44659001257E-313,
    "FiveMinuteRate" : 1.4821969375E-313,
    "RateUnit" : "events/second"
  }, {
    "name" : "ratis:name=ratis.server.db8be976-15d6-4ec6-8ab9-45ef20154b91@group-287DB5A5A866.numWatchRequestTimeout",
    "modelerType" : "com.codahale.metrics.JmxReporter$JmxCounter",
    "Count" : 0
  }, {
    "name" : "ratis:name=ratis.server.db8be976-15d6-4ec6-8ab9-45ef20154b91@group-287DB5A5A866.retryCacheHitCount",
    "modelerType" : "com.codahale.metrics.JmxReporter$JmxGauge",
    "Value" : 0
```

Фрагмент листинга на странице JMX

Config

Страница ***Config*** отображает листинг всех параметров конфигурации, относящихся к работе сервиса Ozone, в формате XML.

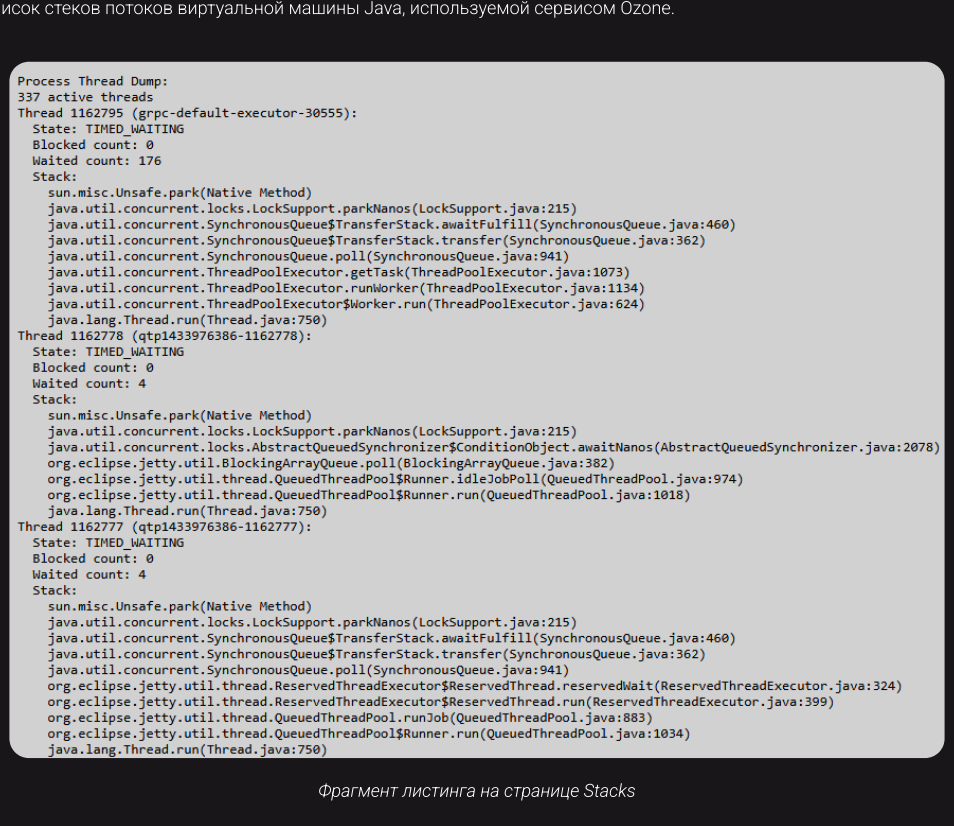


```
▼<configuration>
  ▼<property>
     <name>dfs.namenode.available-space-rack-fault-tolerant-block-placement-policy.balanced-space-tolerance</name>
     <value>5</value>
     <final>false</final>
     <source>hdfs-default.xml</source>
   </property>
  ▼<property>
     <name>fs.s3a.retry.interval</name>
     <value>500ms</value>
     <final>false</final>
     <source>core-default.xml</source>
   </property>
  ▼<property>
     <name>dfs.block.access.token.lifetime</name>
     <value>600</value>
     <final>false</final>
     <source>hdfs-default.xml</source>
   </property>
  ▼<property>
     <name>ozone.client.ec.grpc.retries.max</name>
     <value>3</value>
     <final>false</final>
     <source>ozone-default.xml</source>
   </property>
  ▼<property>
     <name>dfs.namenode.lazypersist.file.scrub.interval.sec</name>
     <value>300</value>
     <final>false</final>
     <source>hdfs-default.xml</source>
   </property>
  ▼<property>
     <name>ozone.client.stream.buffer.increment</name>
     <value>0B</value>
     <final>false</final>
     <source>jar:file:/usr/lib/ozone/share/ozone/lib/hdds-client-1.4.1.jar!/ozone-default-generated.xml</source>
   </property>
  ▼<property>
     <name>file.bytes-per-checksum</name>
     <value>512</value>
     <final>false</final>
     <source>core-default.xml</source>
   </property>
```

Фрагмент листинга на странице Config

Stacks

Страница ***Stacks*** содержит список стеков потоков виртуальной машины Java, используемой сервисом Ozone.



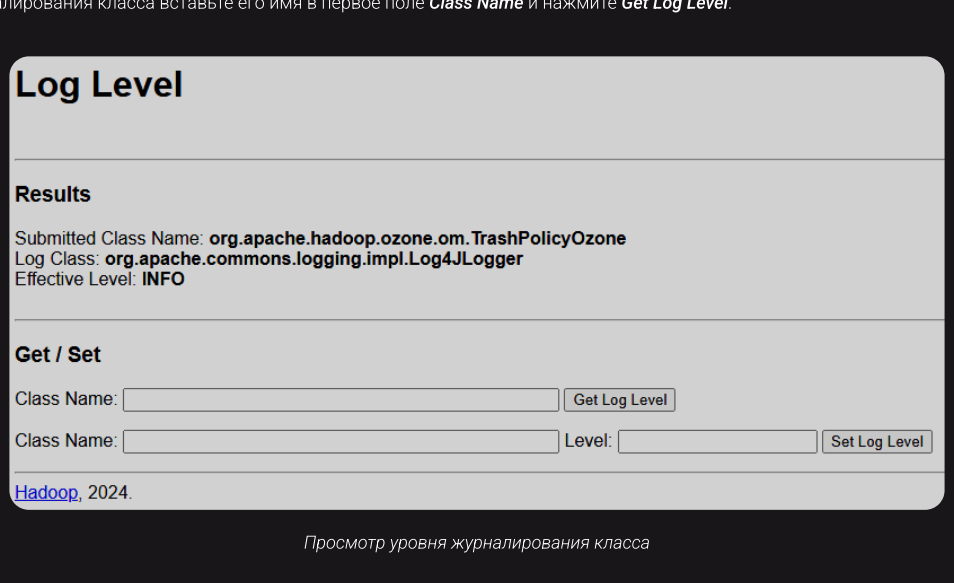
```
Process Thread Dump:
337 active threads
Thread 1162795 (grpc-default-executor-30555):
  State: TIMED_WAITING
  Blocked count: 0
  Waited count: 176
  Stack:
    sun.misc.Unsafe.park(Native Method)
    java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215)
    java.util.concurrent.SynchronousQueue$TransferStack.awaitFulfill(SynchronousQueue.java:460)
    java.util.concurrent.SynchronousQueue$TransferStack.transfer(SynchronousQueue.java:362)
    java.util.concurrent.SynchronousQueue.poll(SynchronousQueue.java:941)
    java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor.java:1073)
    java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1134)
    java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
    java.lang.Thread.run(Thread.java:750)
Thread 1162778 (qtp1433976386-1162778):
  State: TIMED_WAITING
  Blocked count: 0
  Waited count: 4
  Stack:
    sun.misc.Unsafe.park(Native Method)
    java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215)
    java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.awaitNanos(AbstractQueuedSynchronizer.java:2078)
    org.eclipse.jetty.util.BlockingArrayQueue.poll(BlockingArrayQueue.java:382)
    org.eclipse.jetty.util.thread.QueuedThreadPool$Runner.idleJobPoll(QueuedThreadPool.java:974)
    org.eclipse.jetty.util.thread.QueuedThreadPool$Runner.run(QueuedThreadPool.java:1018)
    java.lang.Thread.run(Thread.java:750)
Thread 1162777 (qtp1433976386-1162777):
  State: TIMED_WAITING
  Blocked count: 0
  Waited count: 4
  Stack:
    sun.misc.Unsafe.park(Native Method)
    java.util.concurrent.locks.LockSupport.parkNanos(LockSupport.java:215)
    java.util.concurrent.SynchronousQueue$TransferStack.awaitFulfill(SynchronousQueue.java:460)
    java.util.concurrent.SynchronousQueue$TransferStack.transfer(SynchronousQueue.java:362)
    java.util.concurrent.SynchronousQueue.poll(SynchronousQueue.java:941)
    org.eclipse.jetty.util.thread.ReservedThreadExecutor$ReservedThread.reservedWait(ReservedThreadExecutor.java:324)
    org.eclipse.jetty.util.thread.ReservedThreadExecutor$ReservedThread.run(ReservedThreadExecutor.java:399)
    org.eclipse.jetty.util.thread.QueuedThreadPool.runJob(QueuedThreadPool.java:883)
    org.eclipse.jetty.util.thread.QueuedThreadPool$Runner.run(QueuedThreadPool.java:1034)
    java.lang.Thread.run(Thread.java:750)
```

Фрагмент листинга на странице Stacks

Log levels

Страница ***Log levels*** позволяет просматривать и устанавливать уровни журналирования для классов, используемых в Ozone.

Для просмотра уровня журналирования класса вставьте его имя в первое поле ***Class Name*** и нажмите ***Get Log Level***.



Log Level

Results

Submitted Class Name: **org.apache.hadoop.ozone.om.TrashPolicyOzone**
Log Class: **org.apache.commons.logging.impl.Log4JLogger**
Effective Level: **INFO**

Get / Set

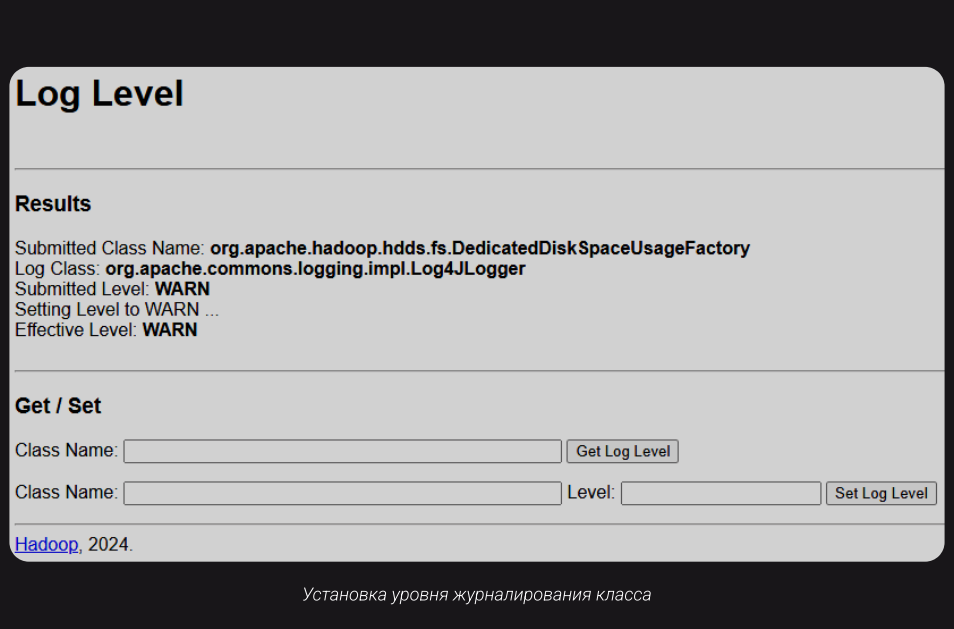
Class Name: ______ Get Log Level

Class Name: ______ Level: ______ Set Log Level

Hadoop, 2024.

Просмотр уровня журналирования класса

Чтобы установить уровень журналирования для класса, вставьте его имя во второе поле ***Class Name***, затем задайте значение уровня журналирования в поле ***Level*** и нажмите ***Set Log Level***.



Log Level

Results

Submitted Class Name: **org.apache.hadoop.hdds.fs.DedicatedDiskSpaceUsageFactory**
Log Class: **org.apache.commons.logging.impl.Log4JLogger**
Submitted Level: **WARN**
Setting Level to WARN ...
Effective Level: **WARN**

Get / Set

Class Name: ______ Get Log Level

Class Name: ______ Level: ______ Set Log Level

Hadoop, 2024.

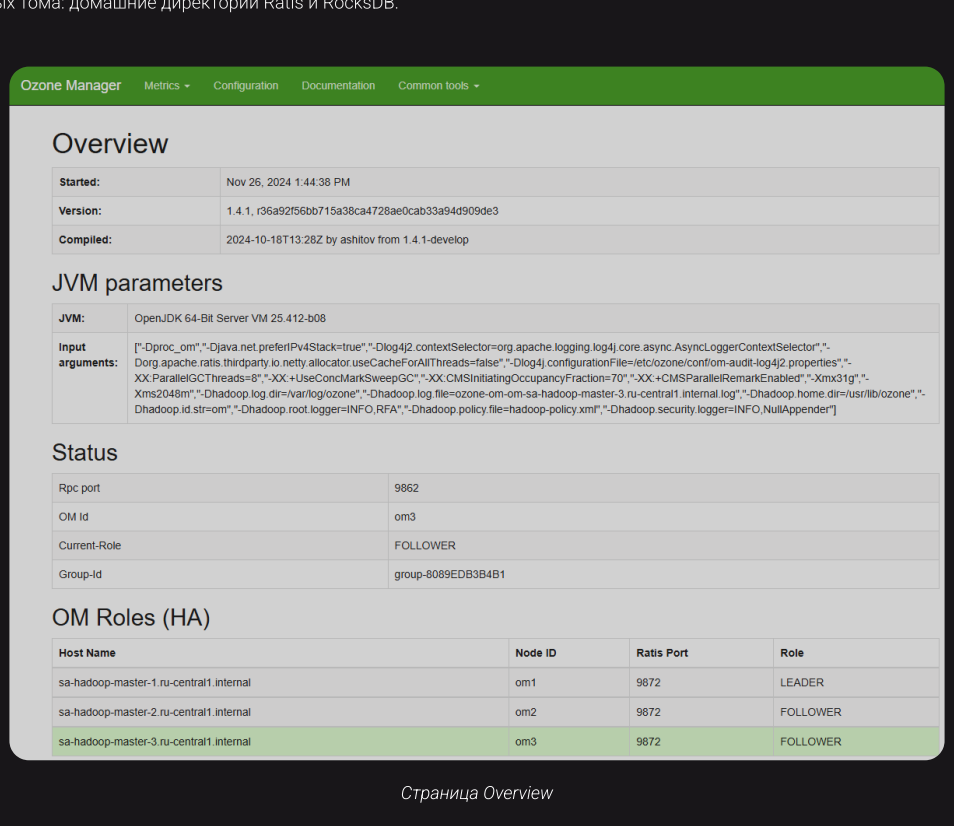
Установка уровня журналирования класса

Веб-интерфейс Ozone Manager

Overview

Начальной страницей веб-интерфейса Ozone Manager является страница ***Overview***. Она содержит следующую информацию:

- Информация о сервисе:
  - время запуска;
  - версия и контрольная сумма;
  - время компиляции, пользователь и исходная ветка.
- Информация о виртуальной машине Java: версия и строка входных аргументов.
- Статус текущего узла Ozone Manager:
  - номер RPC-порта;
  - идентификатор текущего узла Ozone Manager;
  - роль текущего узла Ozone Manager (ведущий или ведомый);
  - идентификатор группы.
- Роли узлов Ozone Manager в режиме высокой доступности. Текущий узел выделяется зеленым цветом.
- Информация о метаданных тома: домашние директории Ratis и RocksDB.



Overview

Started:	Nov 26, 2024 1:44:38 PM
Version:	1.4.1, r36a92f56bb715a38ca4728ae0cab33a94d909de3
Compiled:	2024-10-18T13:28Z by ashitov from 1.4.1-develop

JVM parameters

JVM:	OpenJDK 64-Bit Server VM 25.412-b08
Input arguments:	["-Dproc_om","-Djava.net.preferIPv4Stack=true","-Dlog4j2.contextSelector=org.apache.logging.log4j.core.async.AsyncLoggerContextSelector","-Dorg.apache.ratis.thirdparty.io.netty.allocator.useCacheForAllThreads=false","-Dlog4j.configurationFile=/etc/ozone/conf/om-audit-log4j2.properties","-XX:ParallelGCThreads=8","-XX:+UseConcMarkSweepGC","-XX:CMSInitiatingOccupancyFraction=70","-XX:+CMSParallelRemarkEnabled","-Xmx31g","-Xms2048m","-Dhadoop.log.dir=/var/log/ozone","-Dhadoop.log.file=ozone-om-om-sa-hadoop-master-3.ru-central1.internal.log","-Dhadoop.home.dir=/usr/lib/ozone","-Dhadoop.id.str=om","-Dhadoop.root.logger=INFO,RFA","-Dhadoop.policy.file=hadoop-policy.xml","-Dhadoop.security.logger=INFO,NullAppender"]

Status

Rpc port	9862
OM Id	om3
Current-Role	FOLLOWER
Group-Id	group-8089EDB3B4B1

OM Roles (HA)

Host Name	Node ID	Ratis Port	Role
sa-hadoop-master-1.ru-central1.internal	om1	9872	LEADER
sa-hadoop-master-2.ru-central1.internal	om2	9872	FOLLOWER
sa-hadoop-master-3.ru-central1.internal	om3	9872	FOLLOWER

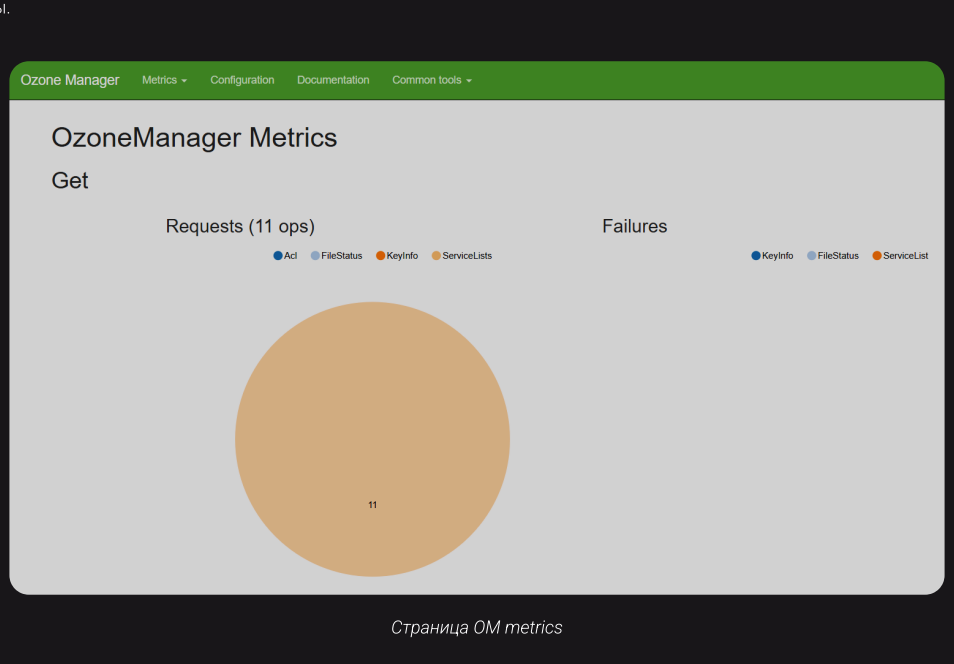
Страница Overview

Metrics

Группа ***Metrics*** содержит страницы ***Rpc metrics*** и ***OM metrics***.

OM metrics

Эта страница содержит круговые диаграммы статистики применения различных параметров JMX (успешно или со сбоем), а также таблицу с некоторыми другими параметрами внизу страницы.

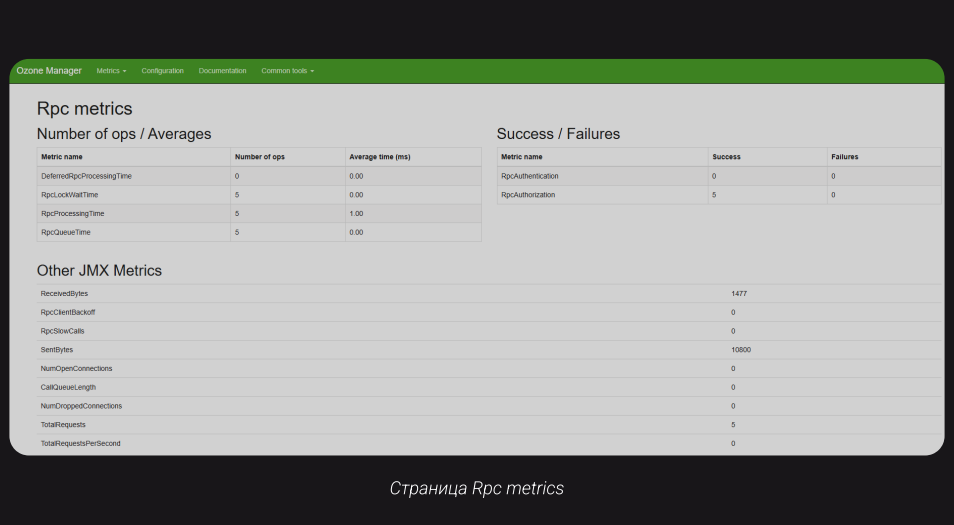


Страница OM metrics

Rpc metrics

Эта страница содержит следующую информацию:

- Количество выполненных операций и их средние времена.
- Статистика успешных и неудачных событий по метрикам RPC аутентификации и авторизации.
- Другие метрики JMX.



Rpc metrics

Number of ops / Averages

Metric name	Number of ops	Average time (ms)
DeferredRpcProcessingTime	0	0.00
RpcLockWaitTime	5	0.00
RpcProcessingTime	5	1.00
RpcQueueTime	5	0.00

Success / Failures

Metric name	Success	Failures
RpcAuthentication	0	0
RpcAuthorization	5	0

Other JMX Metrics

ReceivedBytes	1477
RpcClientBackoff	0
RpcSlowCalls	0
SentBytes	10800
NumOpenConnections	0
CallQueueLength	0
NumDroppedConnections	0
TotalRequests	5
TotalRequestsPerSecond	0

Страница Rpc metrics

Configuration

Страница ***Configuration*** идентична странице Configuration веб-интерфейса Ozone Datanode.

Documentation

Страница ***Documentation*** идентична странице Documentation веб-интерфейса Ozone Datanode.

Common tools

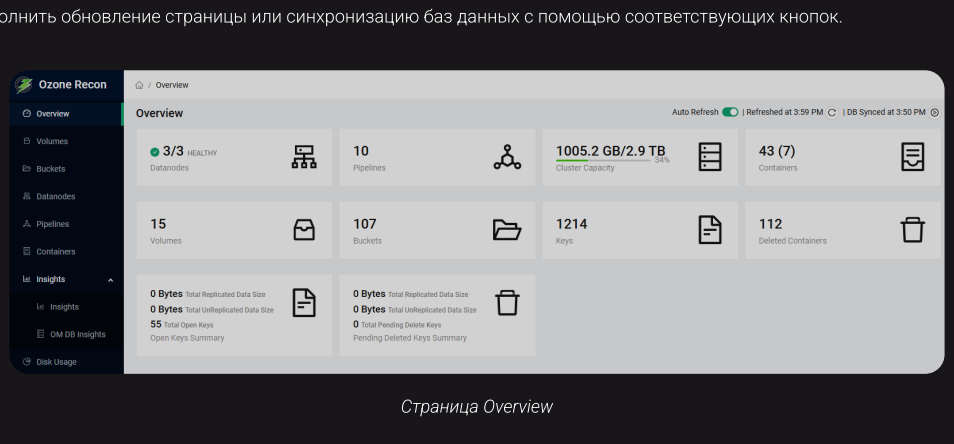
Страницы группы ***Common tools*** идентичны соответствующим страницам группы Common tools веб-интерфейса Ozone Datanode.

Ozone Recon UI

Overview

Начальной страницей веб-интерфейса Ozone Recon является страница ***Overview***. Она содержит 10 графических блоков с основной краткой статистикой. Некоторые блоки содержат ссылки на соответствующие страницы веб-интерфейса. Блок ***Cluster Capacity*** имеет легенду, всплывающую при наведении курсора. Блок ***Container*** также имеет всплывающую подсказку.

Эта страница по умолчанию обновляется автоматически. Можно отключить автообновление, нажав на его переключатель в верхней правой области страницы. В той же области можно вручную выполнить обновление страницы или синхронизацию баз данных с помощью соответствующих кнопок.



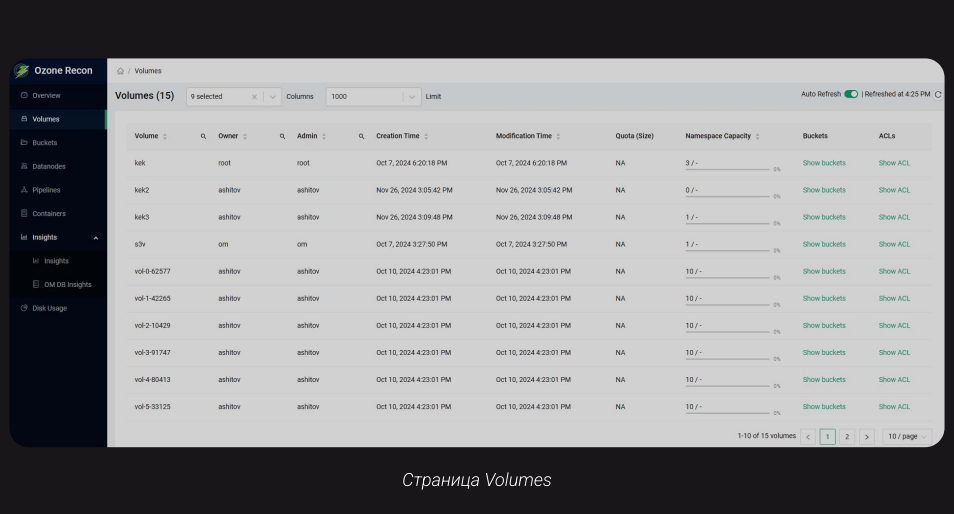
Страница Overview

Volumes

Страница ***Volumes*** содержит информацию о томах, находящихся в хранилище Ozone. Информация представлена в виде таблицы, имеющей следующие столбцы:

- ***Volume*** — имя тома.
- ***Owner*** — имя пользователя владельца тома.
- ***Admin*** — имя пользователя администратора тома.
- ***Creation Time*** — время создания тома.
- ***Modification Time*** — время последнего изменения тома.
- ***Quota*** — значение квоты для тома или бакетов, если задано.
- ***Namespace Capacity*** — статистика использования томом пространств имен (namespaces): использованные, оставшиеся и процент использованных от общего числа пространств имен.
- ***Buckets*** — ссылка на страницу ***Buckets*** с примененным к списку фильтром по этому тому.
- ***ACLs*** — кнопка просмотра списка управления доступом для тома.

В таблице доступна сортировка по всем столбцам, кроме ***Quota***, ***Buckets*** и ***ACLs***. Можно произвольно выбирать столбцы для отображения в выпадающем списке ***Columns*** в верхней области страницы. Также можно задать ограничение на количество строк в выпадающем списке ***Limit*** в той же области. Кроме того, можно включать и выключать автообновление и обновлять страницу вручную таким же образом, как и на странице ***Overview***. В нижней части страницы расположены инструменты пагинации.



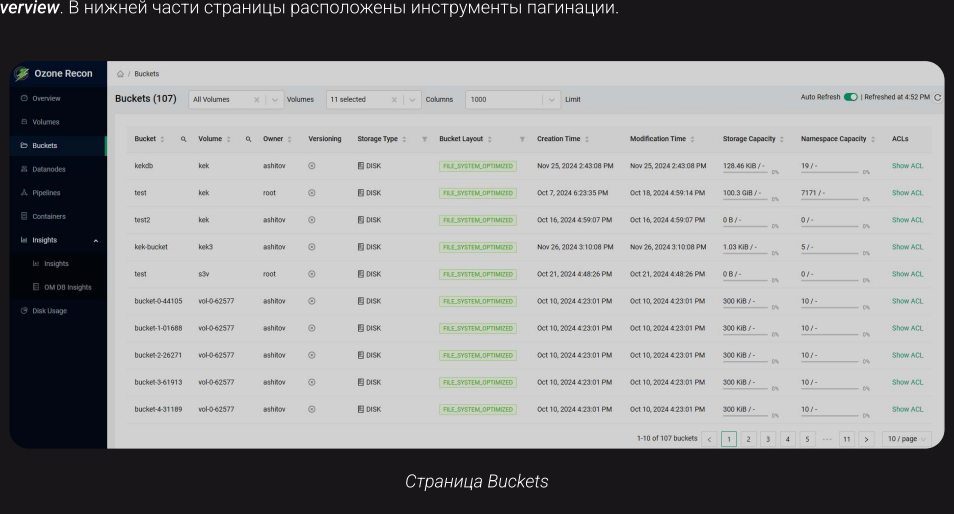
Страница Volumes

Buckets

Страница ***Buckets*** содержит информацию о бакетах, хранящихся в томах. Информация представлена в виде таблицы, имеющей следующие столбцы:

- ***Bucket*** — имя бакета.
- ***Volume*** — имя тома, которому принадлежит бакет.
- ***Owner*** — имя пользователя владельца бакета.
- ***Versioning*** — использование версионности для бакета.
- ***Storage Type*** — тип хранилища, в котором располагается бакет. Возможные значения: `SSD`, `DISK`, `ARCHIVE` или `RAM_DISK`.
- ***Bucket Layout*** — тип схемы организации данных бакета. Возможные значения: `FILE_SYSTEM_OPTIMIZED`, `OBJECT_STORE` или `LEGACY`.
- ***Creation Time*** — время создания бакета.
- ***Modification Time*** — время последнего изменения бакета.
- ***Storage Capacity*** — пространство в хранилище, занимаемое бакетом, оставшееся пространство и процент занятого пространства от общего.
- ***Namespace Capacity*** — статистика использования бакетом пространств имен: использованные, оставшиеся и процент использованных от общего числа пространств имен.
- ***Source Volume*** — том, в котором был создан бакет.
- ***Source Bucket*** — бакет, на основе которого был создан этот бакет.
- ***ACLs*** — кнопка просмотра списка управления доступом для бакета.

В таблице доступна сортировка по всем столбцам, кроме ***Versioning***, ***Source Volume***, ***Source Bucket*** и ***ACLs***. Можно произвольно выбирать столбцы для отображения в выпадающем списке ***Columns*** в верхней области страницы и фильтровать таблицу по томам с помощью выпадающего списка ***Volumes***. Также можно задать ограничение на количество строк в выпадающем списке ***Limit*** в той же области. Кроме того, можно включать и выключать автообновление и обновлять страницу вручную таким же образом, как и на странице ***Overview***. В нижней части страницы расположены инструменты пагинации.



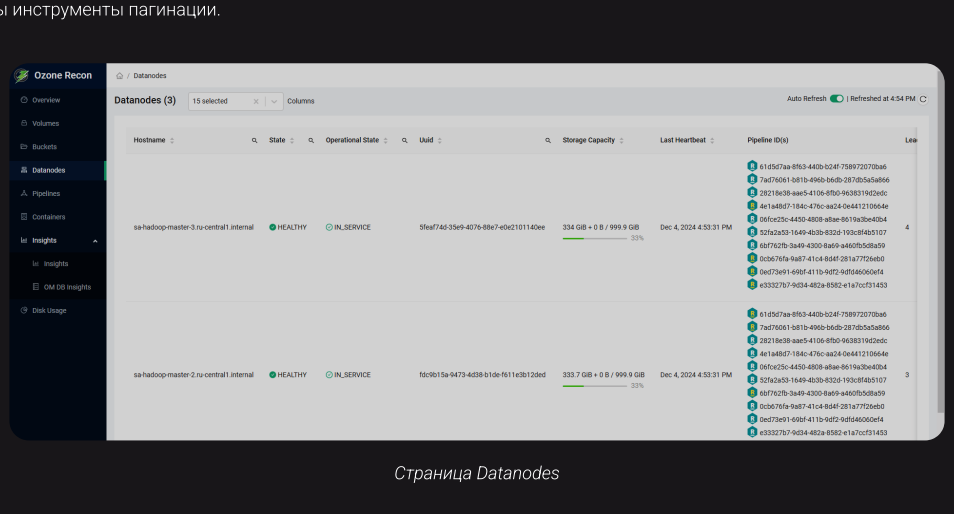
Страница Buckets

Datanodes

Страница ***Datanodes*** содержит информацию об узлах данных (datanode) в кластере Ozone. Информация представлена в виде таблицы, имеющей следующие столбцы:

- ***Hostname*** — полное доменное имя узла данных.
- ***State*** — состояние работоспособности узла данных. Возможные значения: `HEALTHY`, `HEALTHY_READONLY`, `STALE` или `DEAD`.
- ***Operational State*** — рабочее состояние узла данных. Возможные значения: `IN_SERVICE`, `DECOMMISSIONING` или `DECOMMISSIONED`.
- ***Uuid*** — UUID узла данных.
- ***Storage Capacity*** — статистика использования хранилища на узле данных: использованное пространство, оставшееся пространство и процент использованного от общего объема.
- ***Last Heartbeat*** — время получения последнего heartbeat от узла данных.
- ***Pipeline ID(s)*** — список идентификаторов пайплайнов, в которых участвует узел данных.
- ***Leader Count*** — количество пайплайнов, в которых узел данных выбран ведущим.
- ***Containers*** — количество контейнеров в узле данных.
- ***Open Containers*** — количество открытых контейнеров на один пайплайн.
- ***Version*** — номер версии Ozone, установленной в узле данных.
- ***Setup Time*** — время запуска узла данных.
- ***Revision*** — контрольная сумма дистрибутива Ozone, установленного в узле данных.
- ***Build Date*** — время, когда был собран дистрибутив Ozone, установленный в узле данных.
- ***Network Location*** — сетевое расположение узла данных.

В таблице доступна сортировка по всем столбцам, кроме ***Pipeline ID(s)***. Можно произвольно выбирать столбцы для отображения в выпадающем списке ***Columns*** в верхней области страницы. Кроме того, можно включать и выключать автообновление и обновлять страницу вручную таким же образом, как и на странице ***Overview***. В нижней части страницы расположены инструменты пагинации.



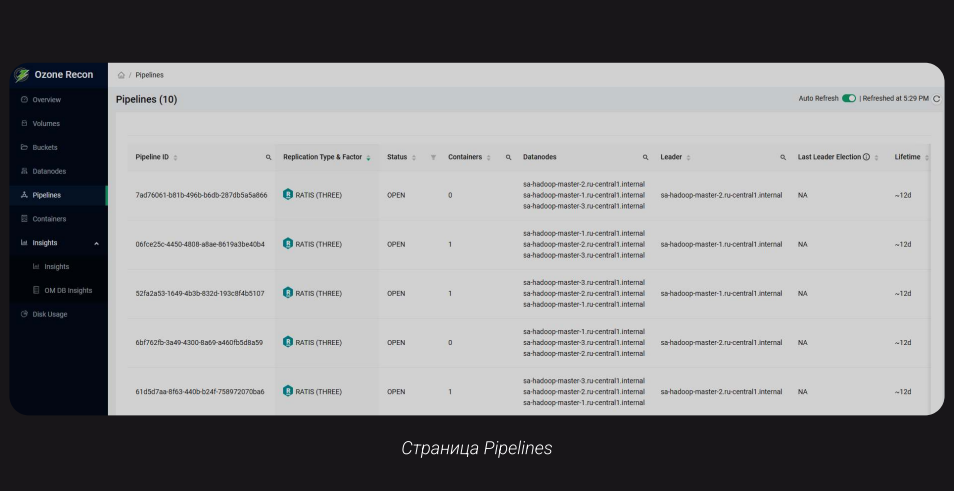
Страница Datanodes

Pipelines

Страница ***Pipelines*** содержит информацию о пайплайнах в кластере Ozone. Информация представлена в виде таблицы, имеющей следующие столбцы:

- ***Pipeline ID*** — идентификатор пайплайна.
- ***Replication Type & Factor*** — тип репликации (`RATIS` или `STANDALONE`) и значение фактора репликации.
- ***Status*** — статус пайплайна. Возможные значения: `OPEN` (открыт) или `CLOSED` (закрыт).
- ***Containers*** — количество контейнеров в пайплайне.
- ***Datanodes*** — список узлов данных в пайплайне.
- ***Leader*** — узел данных, имеющий роль Ratis `LEADER` (ведущий) в пайплайне.
- ***Last Leader Election*** — время последнего назначения узлу данных роли Ratis `LEADER` в пайплайне.
- ***Lifetime*** — время, в течение которого пайплайн является открытым.
- ***No. of Elections*** — количество назначений какому-либо узлу данных роли Ratis `LEADER` в пайплайне.

Можно включать и выключать автообновление и обновлять страницу вручную таким же образом, как и на странице ***Overview***. В нижней части страницы расположены инструменты пагинации.



Страница Pipelines

Containers

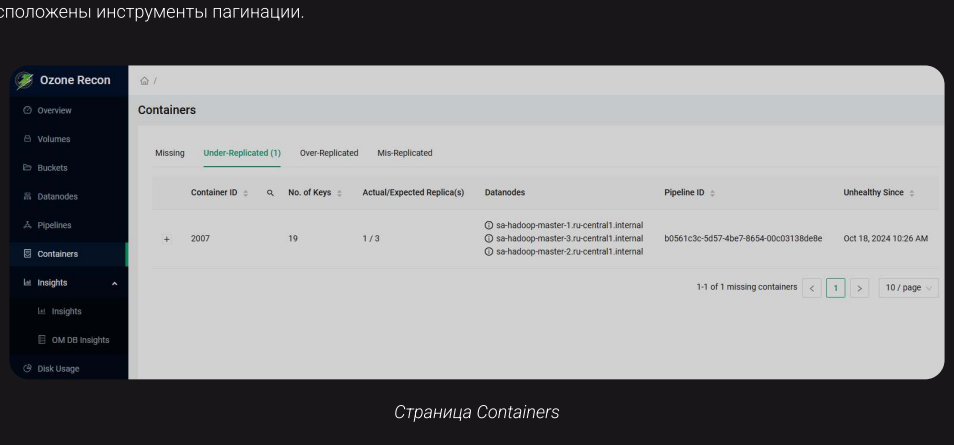
Страница ***Containers*** содержит информацию о контейнерах, хранящихся в кластере Ozone. Она имеет четыре вкладки, которые содержат следующую информацию:

- ***Missing*** — контейнеры, о которых не сообщает ни один узел данных (например, когда контейнер не был удален, но все узлы данных, на которых он был расположен, недоступны).
- ***Under-Replicated*** — контейнеры, у которых меньше реплик, чем ожидается.
- ***Over-Replicated*** — контейнеры, у которых больше реплик, чем ожидается.
- ***Mis-Replicated*** — контейнеры, реплики которых противоречат политике репликации (например, расположены на одном и том же узле).

Информация в каждой из этих вкладок представлена в виде таблицы, имеющей следующие столбцы:

- ***Container ID*** — идентификатор контейнера.
- ***No. of Keys*** — количество ключей, хранящихся в контейнере.
- ***Actual/Expected Replica(s)*** — действительное и ожидаемое количество реплик контейнера.
- ***Datanodes*** — список узлов данных, на которых ожидается хранение реплик контейнера.
- ***Pipeline ID*** — идентификатор пайплайна, которому принадлежит контейнер.
- ***Unhealthy Since*** — момент времени, в который контейнер перестал быть доступным, стал иметь неправильное количество реплик или его реплики стали противоречить политике репликации, в зависимости от вкладки.

В нижней части страницы расположены инструменты пагинации.



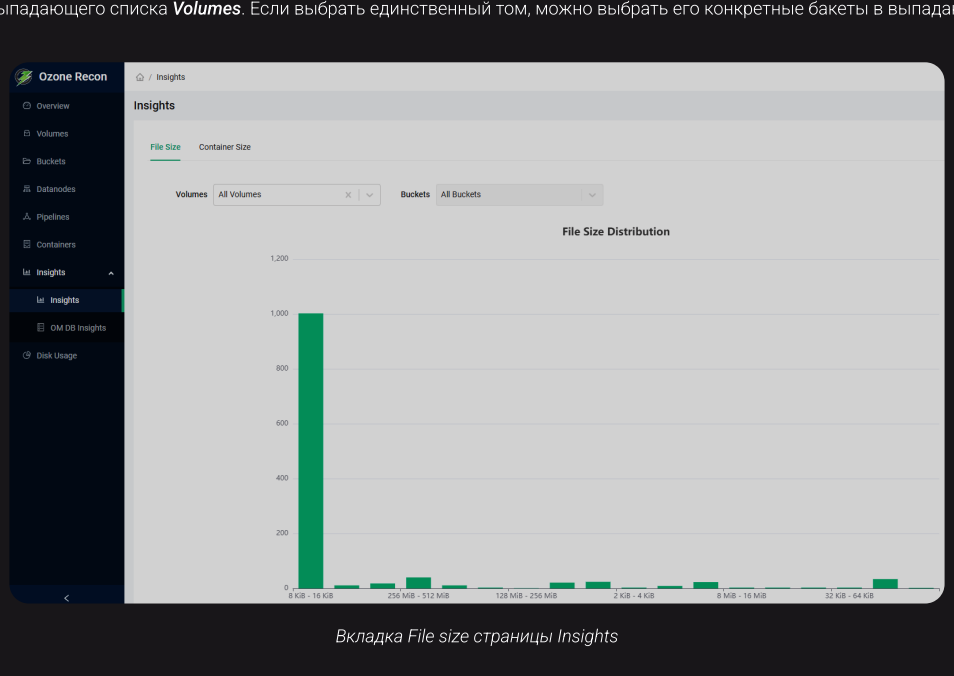
Страница Containers

Insights

Группа ***Insights*** содержит страницы ***Insights*** и ***OM DB Insights***.

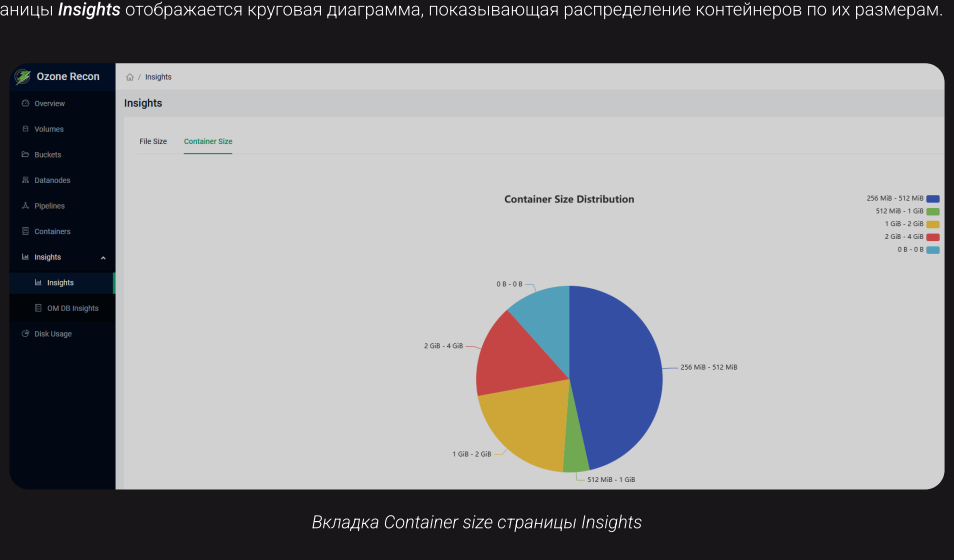
Insights

На вкладке ***File Size*** страницы ***Insights*** отображается гистограмма, показывающая распределение файлов по их размерам. Можно выбрать тома для включения в представление с помощью выпадающего списка ***Volumes***. Если выбрать единственный том, можно выбрать его конкретные бакеты в выпадающем списке ***Buckets***.



Вкладка File size страницы Insights

На вкладке ***Container Size*** страницы ***Insights*** отображается круговая диаграмма, показывающая распределение контейнеров по их размерам.



Вкладка Container size страницы Insights

OM DB Insights

Страница ***OM DB Insights*** содержит четыре вкладки:

Container Mismatch Info

Контейнеры должны присутствовать в базах данных Ozone Manager и Storage Container Manager. Эта вкладка содержит информацию о контейнерах, которые присутствуют только в одной из этих баз данных (рассогласованные контейнеры). Информация представлена в виде таблицы, имеющей следующие столбцы:

- ***Container ID*** — идентификатор рассогласованного контейнера.
- ***Count Of Keys*** — количество ключей, хранящихся в рассогласованном контейнере.
- ***Pipelines*** — идентификаторы пайплайнов, связанных с рассогласованным контейнером.
- ***Exists*** — база данных, в которой присутствует рассогласованный контейнер. Возможные значения: `OM` или `SCM`.

Open Keys

Эта вкладка содержит информацию об открытых ключах в базе данных Ozone Manager. Информация представлена в виде таблицы, имеющей следующие столбцы:

- ***Key*** — идентификатор ключа.
- ***Amount of data*** — размер, занятый содержимым ключа.
- ***Path*** — путь к расположению ключа.
- ***In state since*** — время с момента открытия ключа.
- ***Replication Factor*** — значение фактора репликации ключа.
- ***Replication Type*** — тип репликации ключа. Возможные значения: `RATIS` или `STANDALONE`.
- ***Type*** — тип ключа.

Keys Pending for Deletion

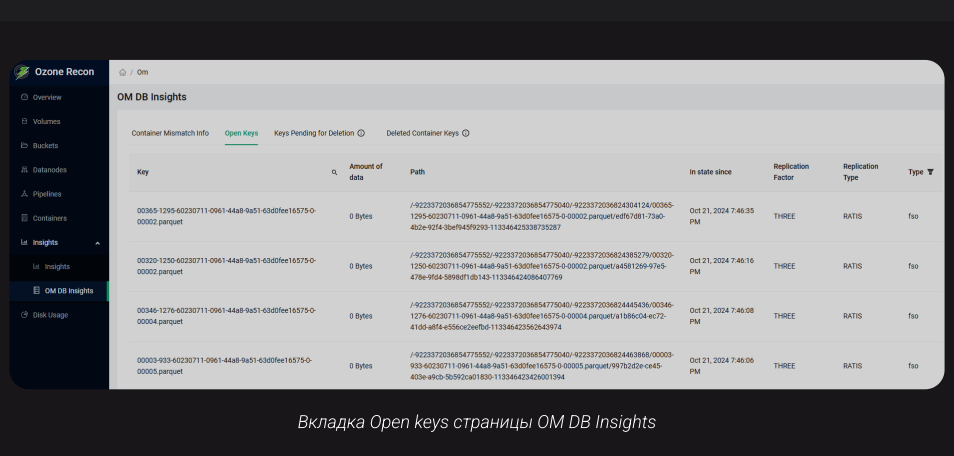
Эта вкладка содержит информацию о ключах, ожидающих удаления. Информация представлена в виде таблицы, имеющей следующие столбцы:

- ***Key Name*** — имя удаляемого ключа.
- ***Path*** — путь к расположению ключа.
- ***Total Data Size*** — размер, занятый данными ключа.
- ***Total Key Count*** — количество удаляемых ключей.

Deleted Container Keys

Эта вкладка содержит информацию о ключах, связанных с контейнерами, имеющими состояние `DELETED` (удален) в базе данных Storage Container Manager. Информация представлена в виде таблицы, имеющей следующие столбцы:

- ***Container ID*** — идентификатор удаленного контейнера.
- ***Count Of Keys*** — количество ключей, связанных с контейнером.
- ***Pipelines*** — список пайплайнов, связанных с контейнером.



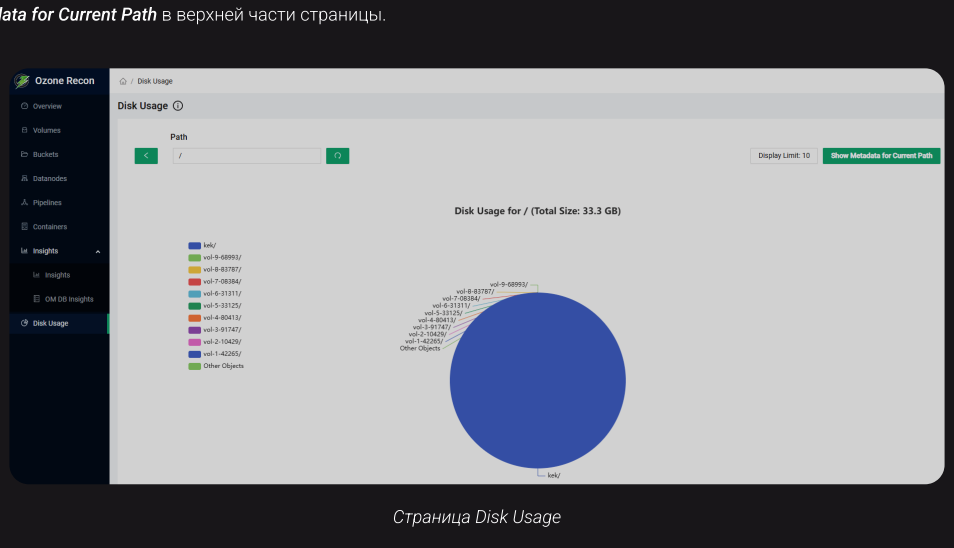
Вкладка Open keys страницы OM DB Insights

Disk Usage

Страница ***Disk Usage*** содержит круговую диаграмму использования дискового пространства бакетами, хранящимися в кластере Ozone. Можно фильтровать данные следующими способами:

- Можно ввести путь в поле ***Path*** для просмотра статистики данных, хранящихся в указанной директории и всех её поддиректориях.
- Можно нажать на любой пункт цветовой легенды в левой части страницы или на сектор круговой диаграммы (или на его выноску) для просмотра их собственной статистики. Обратите внимание, что при этом соответствующий путь появляется в поле ***Path***. Это можно проделать несколько раз до достижения самого нижнего уровня вложенности.
- Можно нажать на кнопку **<** слева от поля ***Path***, чтобы переместиться на одну директорию вверх.

Также можно выбрать ограничение на количество пунктов для отображения с помощью выпадающего списка ***Display Limit*** и просматривать метаданные заданного пути с помощью кнопки ***Show Metadata for Current Path*** в верхней части страницы.

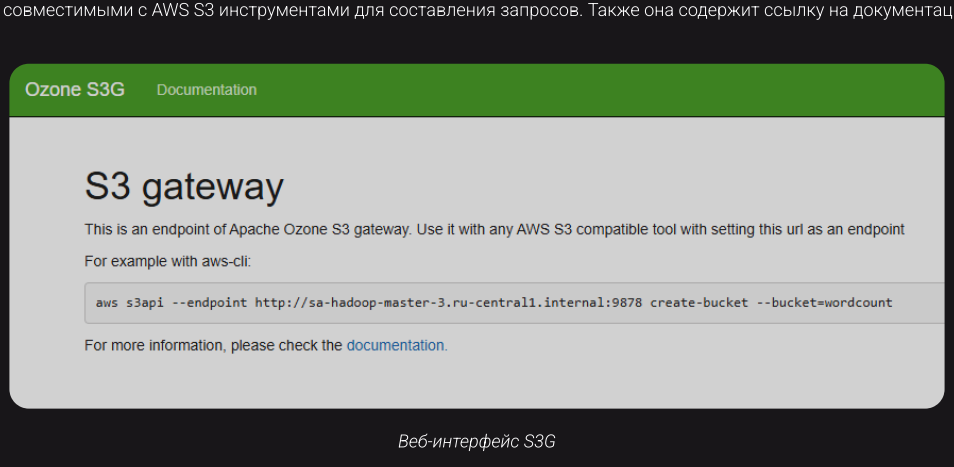


Ozone Recon
Disk Usage
Path
Display Limit: 10
Show Metadata for Current Path
Disk Usage for / (Total Size: 33.3 GB)
Other Objects

Страница Disk Usage

Веб-интерфейс Ozone S3G

Единственная страница веб-интерфейса S3G по существу является заглушкой. Она сообщает, что её адрес является конечной точкой (endpoint) шлюза S3 и что он может быть использован с любыми совместимыми с AWS S3 инструментами для составления запросов. Также она содержит ссылку на документацию Apache Ozone.



S3 gateway

This is an endpoint of Apache Ozone S3 gateway. Use it with any AWS S3 compatible tool with setting this url as an endpoint

For example with aws-cli:

```
aws s3api --endpoint http://sa-hadoop-master-3.ru-central1.internal:9878 create-bucket --bucket=wordcount
```

For more information, please check the documentation.

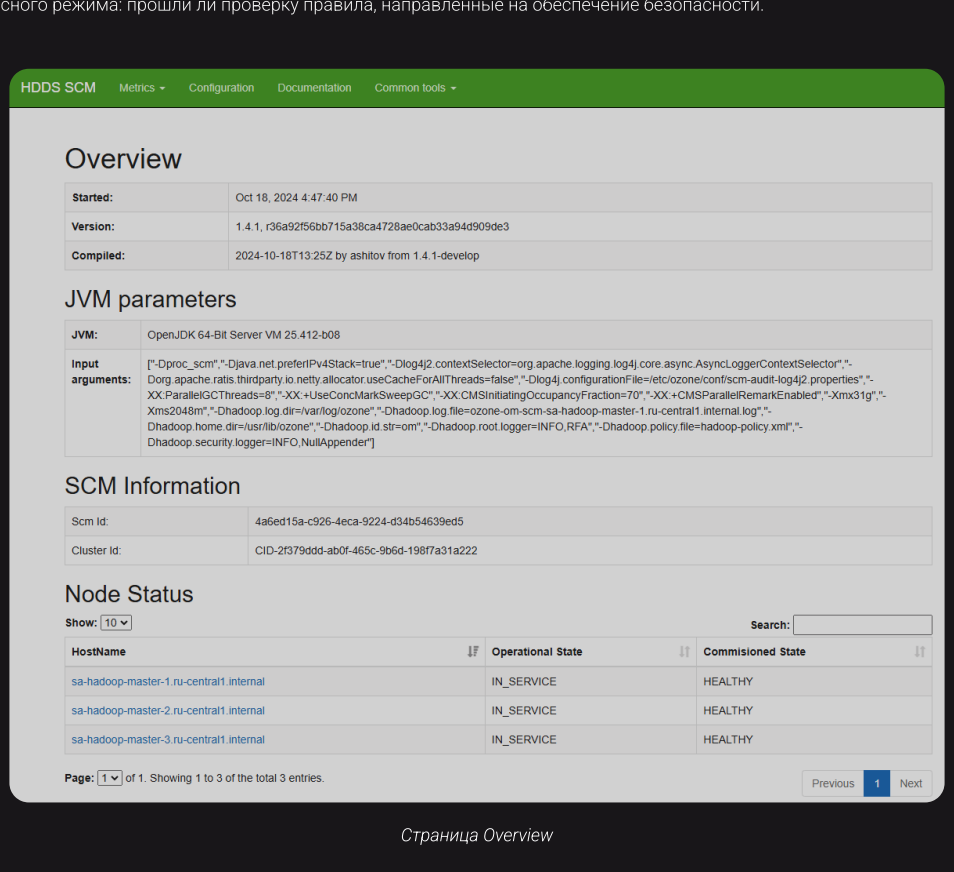
Веб-интерфейс S3G

Веб-интерфейс Ozone Storage Container Manager

Overview

Начальной страницей веб-интерфейса Ozone Storage Container Manager (SCM) является страница ***Overview***. Она содержит следующую информацию:

- Информация о сервисе:
  - время запуска;
  - версия и контрольная сумма;
  - время компиляции, пользователь и исходная ветка.
- Информация о виртуальной машине Java: версия и строка входных аргументов.
- Информация SCM: идентификаторы SCM и кластера.
- Статусы узлов в кластере SCM.
- Состояние режима высокой доступности:
  - RPC-порты клиентов и узлов данных;
  - состояние безопасного режима менеджера узлов;
  - роли узлов SCM;
  - имя первичного узла SCM.
- Информация о метаданных тома: домашние директории Ratis и RocksDB.
- Состояния правил безопасного режима: прошли ли проверку правила, направленные на обеспечение безопасности.



Overview

Started:	Oct 18, 2024 4:47:40 PM
Version:	1.4.1, r36a92f56bb715a38ca4728ae0cab33a94d909de3
Compiled:	2024-10-18T13:25Z by ashitov from 1.4.1-develop

JVM parameters

JVM:	OpenJDK 64-Bit Server VM 25.412-b08
Input arguments:	["-Dproc_scm","-Djava.net.preferIPv4Stack=true","-Dlog4j2.contextSelector=org.apache.logging.log4j.core.async.AsyncLoggerContextSelector","-Dorg.apache.ratis.thirdparty.io.netty.allocator.useCacheForAllThreads=false","-Dlog4j.configurationFile=/etc/ozone/conf/scm-audit-log4j2.properties","-XX:ParallelGCThreads=8","-XX:+UseConcMarkSweepGC","-XX:CMSInitiatingOccupancyFraction=70","-XX:+CMSParallelRemarkEnabled","-Xmx31g","-Xms2048m","-Dhadoop.log.dir=/var/log/ozone","-Dhadoop.log.file=ozone-om-scm-sa-hadoop-master-1.ru-central1.internal.log","-Dhadoop.home.dir=/usr/lib/ozone","-Dhadoop.id.str=om","-Dhadoop.root.logger=INFO,RFA","-Dhadoop.policy.file=hadoop-policy.xml","-Dhadoop.security.logger=INFO,NullAppender"]

SCM Information

Scm Id:	4a6ed15a-c926-4eca-9224-d34b54639ed5
Cluster Id:	CID-2f379ddd-ab0f-465c-9b6d-198f7a31a222

Node Status

Show: 10 Search:

HostName	Operational State	Commisioned State
sa-hadoop-master-1.ru-central1.internal	IN_SERVICE	HEALTHY
sa-hadoop-master-2.ru-central1.internal	IN_SERVICE	HEALTHY
sa-hadoop-master-3.ru-central1.internal	IN_SERVICE	HEALTHY

Page: 1 of 1. Showing 1 to 3 of the total 3 entries. Previous 1 Next

Страница Overview

Metrics

Страница ***Metrics*** идентична странице Metrics веб-интерфейса Ozone Datanode.

Configuration

Страница ***Configuration*** идентична странице Configuration веб-интерфейса Ozone Datanode.

Documentation

Страница ***Documentation*** идентична странице Documentation веб-интерфейса Ozone Datanode.

Common tools

Страницы группы ***Common tools*** идентичны соответствующим страницам группы Common tools веб-интерфейса Ozone Datanode.

Интеграция Ozone с сервисами кластера

Константин Алпашкин

Содержание

Назад к содержанию

- Тестовый кластер
- Настройка сервисов ADH для работы с Ozone
  - Шаг 1. Подготовка файловой системы Ozone
  - Шаг 2. Настройка Spark3
  - Шаг 3. Настройка Hive
  - Шаг 4. Настройка YARN
- Использование Ozone с HDFS
  - Hive
  - Spark3

В данной статье описан кейс интеграции сервиса Ozone с сервисами уже развернутого кластера ADH. В примерах показано, как настроить сервисы ADH (Spark3, Hive, Impala, YARN), чтобы вместо HDFS использовалась файловая система Ozone.

После переключения на Ozone сервисы ADH могут продолжать взаимодействовать с HDFS, а обе файловые системы могут использоваться вместе.

Тестовый кластер

Для приведенного сценария используется следующее окружение:

- Развернутый кластер ADH версии 3.3.6.2 или выше. Кластер керберизирован и включает следующие сервисы:
  - Core configuration
  - ZooKeeper
  - HDFS
  - YARN
  - ADPG
  - Hive
  - Impala
  - Spark3
- Сервис Ozone добавлен в уже функционирующий ADH-кластер. После установки сервиса Ozone необходимо выполнить действие ***Update Core configuration*** сервиса Core configuration, а также перезапустить следующие сервисы:
  - YARN
  - Hive
  - Impala
  - Spark3

  Это необходимо, чтобы подгрузить нужные зависимости Ozone в classpath.
- Сервисы HDFS/Ozone используют следующие пространства имен (namespaces).

Сервис/Компонент	Конфигурационный параметр	Значение
HDFS	dfs.internal.nameservices	adh
Ozone	ozone.service.id	adho
Ozone Manager	ozone.om.service.ids	adho
Ozone Storage Container Manager	ozone.scm.service.ids	adho

Настройка сервисов ADH для работы с Ozone

Шаг 1. Подготовка файловой системы Ozone

1. Используя Ozone CLI, создайте том (volume) и бакет (bucket), где будут храниться данные сервисов ADH:

```
$ ozone sh volume create ozone
$ ozone sh bucket create /ozone/adh
```

Проверьте созданный бакет:

```
$ hdfs dfs -ls ofs://adho/ozone
```

Вывод:

```
drwxrwxrwx   - k_alpashkin_krb1 k_alpashkin_krb1          0 2025-02-05 15:41 ofs://adho/ozone/adh
```

2. В сервисе Core configuration установите Ozone в качестве дефолтной файловой системы с помощью следующего параметра.

Раздел настроек	Конфигурационный параметр	Значение
core-site.xml	fs.defaultFS	ofs://adho

3. Выполните действие ***Update Core configuration*** сервиса Core configuration.

Шаг 2. Настройка Spark3

1. Скопируйте необходимые архивы Spark из HDFS в Ozone и создайте лог-директории:

```
$ hdfs dfs -get hdfs://adh/apps/spark/spark3-yarn-archive.tgz
$ hdfs dfs -mkdir -p ofs://adho/ozone/adh/apps/spark
$ hdfs dfs -put spark3-yarn-archive.tgz ofs://adho/ozone/adh/apps/spark/spark3-yarn-archive.tgz
$ hdfs dfs -mkdir -p ofs://adho/ozone/adh/var/log/spark3/apps
```

2. Настройте сервис Spark3 для работы с Ozone. Для этого укажите параметры конфигурации, перечисленные в таблице.

Раздел настроек	Конфигурационный параметр	Значение
spark-defaults.conf	spark.yarn.archive	ofs://adho/ozone/adh/apps/spark/spark3-yarn-archive.tgz
	spark.eventLog.dir	ofs://adho/ozone/adh/var/log/spark3/apps
	spark.history.fs.logDirectory	

3. Перезапустите сервис Spark3.

Шаг 3. Настройка Hive

1. Скопируйте необходимые файлы Hive из HDFS в Ozone и создайте новую warehouse-директорию:

```
$ hdfs dfs -get hdfs://adh/apps/tez/tez-0.10.3.tar.gz
$ hdfs dfs -mkdir -p ofs://adho/ozone/adh/apps/tez
$ hdfs dfs -put tez-0.10.3.tar.gz ofs://adho/ozone/adh/apps/tez/tez-0.10.3.tar.gz
$ hdfs dfs -mkdir -p ofs://adho/ozone/adh/apps/hive/warehouse
```

2. Настройте сервис Hive на использование новой warehouse-директории в Ozone.

Раздел настроек	Конфигурационный параметр	Значение
hive-site.xml	hive.metastore.warehouse.dir	ofs://adho/ozone/adh/apps/hive/warehouse

3. (Опционально) Измените расположение файлов базы данных `default`.

Изменение расположения файлов БД

1. На хосте с установленным клиентом Hive запустите оболочку `beeline`:

```
$ beeline
```

2. В оболочке `beeline` подключитесь к Hive, используя строку JDBC для керберизированной среды, например:

```
!connect jdbc:hive2://ka-adh-1.ru-central1.internal:2181,ka-adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.
internal:2181/;principal=hive/ka-adh-2.ru-central1.internal@AD.RANGER-
TEST;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=arenadata/cluster/17/hiveserver2
```

РЕКОМЕНДАЦИЯ

Актуальную JDBC-строку можно найти на странице сервиса Hive в ADCM (***Clusters*** → ***<cluster_name>*** → ***Services*** → ***Hive*** → ***Info***).

3. Измените расположение файлов базы данных, выполнив команду:

```
ALTER DATABASE default SET LOCATION 'ofs://adho/ozone/adh/apps/hive/warehouse';
```

4. Проверьте новое расположение данных таблицы:

```
DESCRIBE DATABASE EXTENDED default;
```

Вывод:

```
+----------+------------------------+--------------------------------------------+-----------------------------------------
--+-------------+-------------+-----------------+----------------+-------------+
| db_name  |        comment         |                  location                  |              managedlocation
| owner_name  | owner_type  | connector_name  | remote_dbname  | parameters  |
+----------+------------------------+--------------------------------------------+-----------------------------------------
--+-------------+-------------+-----------------+----------------+-------------+
| default  | Default Hive database  | ofs://adho/ozone/adh/apps/hive/warehouse   | ofs://adho/ozone/adh/apps/hive/warehouse
| public      | ROLE        |                 |                |             |
+----------+------------------------+--------------------------------------------+-----------------------------------------
--+-------------+-------------+-----------------+----------------+-------------+
```

4. Перезапустите сервис Hive.

Шаг 4. Настройка YARN

1. Создайте необходимые для работы YARN директории:

```
$ hdfs dfs -mkdir -p ofs://adho/ozone/adh/system/yarn/node-labels
```

2. Настройте YARN для работы с Ozone. Для этого укажите параметры конфигурации, перечисленные в таблице.

Раздел настроек	Конфигурационный параметр	Значение
Custom yarn-site.xml	yarn.node-labels.fs-store.root-dir	ofs://adho/ozone/adh/system/yarn/node-labels
	yarn.nodemanager.remote-app-log-dir	ofs://adho/ozone/adh/logs

3. Перезапустите сервис YARN.

После выполнения этих шагов сервисы Hive, Spark и Impala будут по умолчанию сохранять данные в Ozone, а не в HDFS. Кроме того, временные и пользовательские файлы, генерируемые сервисом YARN во время выполнения приложений, также будут сохраняться в Ozone.

Использование Ozone с HDFS

После переключения на файловую систему Ozone вышеупомянутые сервисы могут также взаимодействовать с HDFS.

Hive

Например, все таблицы Hive, которые были сохранены в HDFS до перехода на Ozone, остаются доступны. Для создания новых объектов Hive в HDFS, необходимо явно указать локацию. Например:

```
CREATE DATABASE IF NOT EXISTS db_for_hdfs
COMMENT 'This is a database in HDFS.'
LOCATION 'hdfs://adh/apps/hive/warehouse/db_for_hdfs.db';
```

ПРИМЕЧАНИЕ

Все таблицы, созданные в базе данных `db_for_hdfs`, по умолчанию будут сохраняться в HDFS.

Проверьте расположение файлов базы данных:

```
DESCRIBE DATABASE EXTENDED db_for_hdfs;
```

Вывод:

```
+--------------+---------------------------+------------------------------------------------+------------------------------------
----------------+-------------+-------------+-----------------+----------------+-------------+
|   db_name    |          comment          |                    location                    |                 managedlocation
| owner_name  | owner_type  | connector_name  | remote_dbname  | parameters  |
+--------------+---------------------------+------------------------------------------------+------------------------------------
----------------+-------------+-------------+-----------------+----------------+-------------+
| db_for_hdfs  | This is database in hdfs  | hdfs://adh/apps/hive/warehouse/db_for_hdfs.db  |
ofs://adho/ozone/adh/apps/hive/warehouse/db_for_hdfs.db | hive  | USER        |                 |                |             |
+--------------+---------------------------+------------------------------------------------+------------------------------------
----------------+-------------+-------------+-----------------+----------------+-------------+
```

Spark3

После переключения керберизированного сервиса Spark3 на файловую систему Ozone кластер Spark3 может также работать с HDFS. Однако для этого требуется явно указать, что для доступа к HDFS необходимо пройти Kerberos-аутентификацию. Для этого используйте параметр `spark.kerberos.access.hadoopFileSystems`, принимающий список керберизированных файловых систем, с которыми будет работать приложение Spark. Например:

```
/bin/spark3-submit \
    --deploy-mode cluster \
    --master yarn \
    --conf spark.kerberos.access.hadoopFileSystems=hdfs://adh
    test.py
```

ПРИМЕЧАНИЕ

При использовании Ozone сервис Spark3 записывает staging-данные в бакет Ozone */user/{username}*. Необходимо учитывать, что в керберизированной среде в качестве `{username}` используется имя Kerberos-принципала. Если имя пользователя, от которого запускается приложение Spark, не соответствует требованиям к именованию бакетов Ozone (например, содержит символы `_`), приложение Spark не выполнится, так как Ozone не сможет создать нужный бакет. Чтобы записать staging-данные Spark в заранее созданный бакет Ozone, используйте параметр `spark.yarn.stagingDir`. Данный параметр можно указать либо в настройках ADCM (***Custom spark-defaults.conf***), либо с помощью `--conf`, например:

```
/bin/spark3-submit \
    --deploy-mode cluster \
    --master yarn \
    --conf spark.yarn.stagingDir=ofs://adho/ozone/adh/var \
    test.py
```

Rack awareness в Ozone

Владимир Адаменков

Назад к содержанию

Rack awareness в Ozone — это функция, учитывающая физическую топологию сети при размещении данных. Она очень важна для локальности данных, отказоустойчивости и общей производительности, особенно в географически распределенном кластере. Если эта функция включена, Ozone будет размещать каждую реплику ключа на хосте в отдельной стойке. Это гарантирует доступность данных в случае сетевого сбоя или других проблем.

Чтобы настроить rack awareness для Ozone через ADCM, выполните следующие шаги:

1. Перейдите в веб-интерфейс ADCM и выберите ваш кластер на странице ***Clusters***.
2. Перейдите на вкладку ***Services*** и нажмите ***Ozone***.
3. Переведите в активное состояние переключатель ***Show advanced*** и найдите поля ***Topology script*** и ***Topology data***.
4. Вставьте скрипт, описывающий топологию вашей сети, в качестве значения параметра ***Topology script***.

Пример Topology script

```
#!/bin/bash

# Adjust/Add the property "net.topology.script.file.name"
# to core-site.xml with the "absolute" path the this
# file.  ENSURE the file is "executable".

# Supply appropriate rack prefix
RACK_PREFIX=default

# To test, supply a hostname as script input:
if [ $# -gt 0 ]; then

CTL_FILE=${CTL_FILE:-"topology.data"}

HADOOP_CONF=${HADOOP_CONF:-"/etc/hadoop/conf"}

if [ ! -f ${HADOOP_CONF}/${CTL_FILE} ]; then
  echo -n "/$RACK_PREFIX/rack "
  exit 0
fi

while [ $# -gt 0 ] ; do
  nodeArg=$1
  exec< ${HADOOP_CONF}/${CTL_FILE}
  result=""
  while read line ; do
    ar=( $line )
    if [ "${ar[0]}" = "$nodeArg" ] ; then
      result="${ar[1]}"
    fi
  done
  shift
  if [ -z "$result" ] ; then
    echo -n "/$RACK_PREFIX/rack "
  else
    echo -n "/$RACK_PREFIX/rack_$result "
  fi
done

else
  echo -n "/$RACK_PREFIX/rack "
fi
```

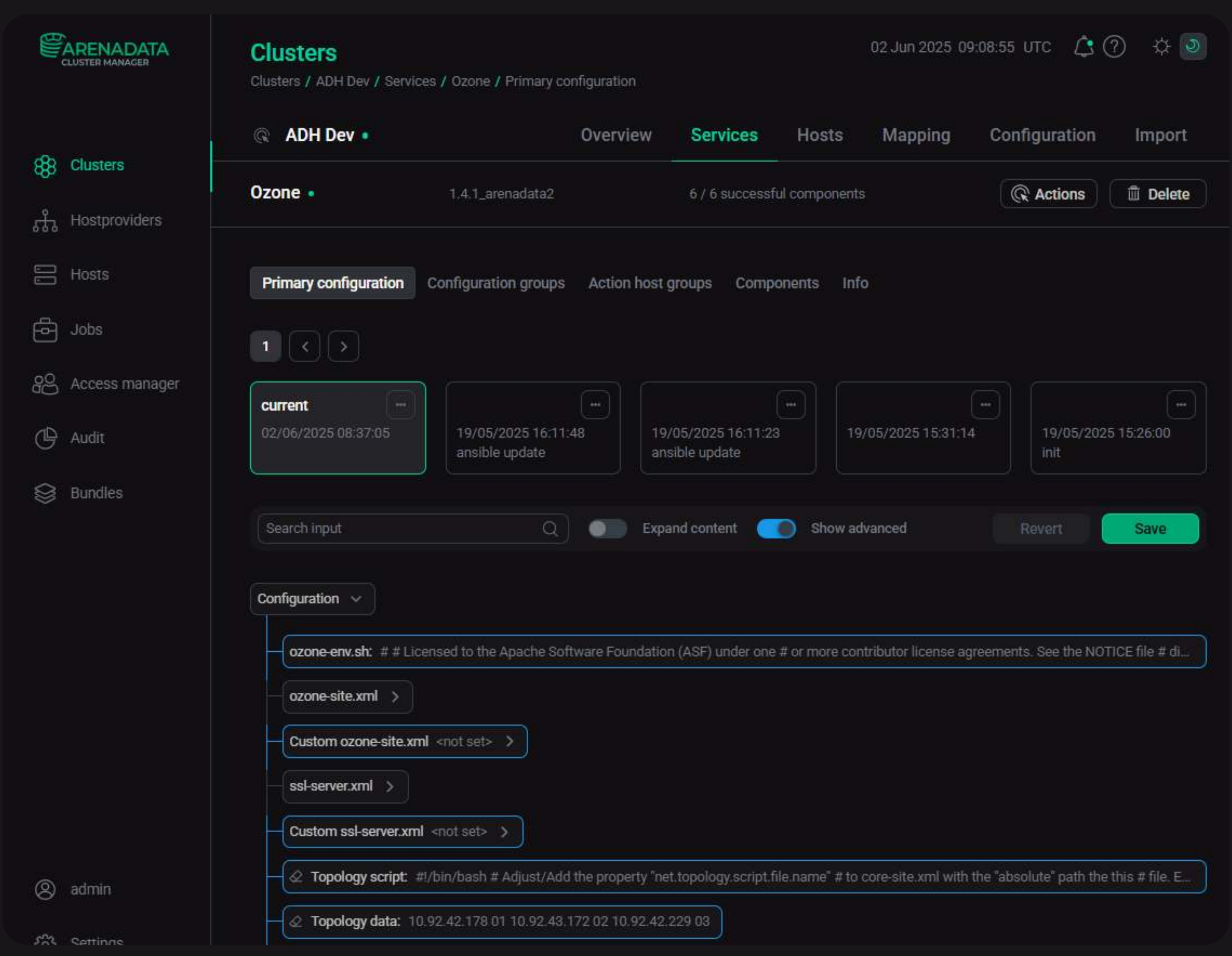
Больше примеров скриптов топологии доступно в документации Rack Awareness.

5. В поле ***Topology data*** укажите IP-адреса стоек и соответствующие им идентификаторы следующим образом:

Пример Topology data

```
# This file should be:
#  - Placed in the /etc/hadoop/conf directory
#    - On the Namenode (and backups IE: HA, Failover, etc)
#    - On the Job Tracker OR Resource Manager (and any Failover JT's/RM's)
# This file should be placed in the /etc/hadoop/conf directory.

# Add Hostnames to this file. Format <host ip> <rack_location>
10.92.42.178 01
10.92.43.172 02
10.92.42.229 03
```

6. Нажмите ***Save***, затем ***Create***.
7. В меню ***Actions*** выберите ***Restart***.
8. Убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.


ARENADATA
CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Clusters
Clusters / ADH Dev / Services / Ozone / Primary configuration
02 Jun 2025 09:08:55 UTC
ADH Dev
Overview
Services
Hosts
Mapping
Configuration
Import
Ozone
1.4.1_arenadata2
6 / 6 successful components
Actions
Delete
Primary configuration
Configuration groups
Action host groups
Components
Info
1
current
02/06/2025 08:37:05
19/05/2025 16:11:48
ansible update
19/05/2025 16:11:23
ansible update
19/05/2025 15:31:14
19/05/2025 15:26:00
init
Search input
Expand content
Show advanced
Revert
Save
Configuration

ozone-env.sh: # # Licensed to the Apache Software Foundation (ASF) under one # or more contributor license agreements. See the NOTICE file # di...

ozone-site.xml

Custom ozone-site.xml <not set>

ssl-server.xml

Custom ssl-server.xml <not set>

Topology script: #!/bin/bash # Adjust/Add the property "net.topology.script.file.name" # to core-site.xml with the "absolute" path the this # file. E...

Topology data: 10.92.42.178 01 10.92.43.172 02 10.92.42.229 03

Параметры Topology script и Topology data в ADCM

Чтобы проверить, успешно ли настроен rack awareness, можно выполнить следующую команду:

```
$ ozone admin datanode list
```

Вывод команды должен иметь следующий вид:

```
Datanode 10.92.42.178:9866 - Rack: /rack01 - Status: UP
Datanode 10.92.43.172:9866 - Rack: /rack02 - Status: UP
Datanode 10.92.42.229:9866 - Rack: /rack03 - Status: UP
```

Управление сервисом Ozone через ADCM

Владимир Адаменков

Содержание

Назад к содержанию

Обзор

Обзор

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Ozone. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Ozone, перечислены в таблице ниже.

Действия с сервисом Ozone

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавлять, удалять и распределять компоненты Ozone
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Manage Ranger plugin	Включает или отключает плагин Ranger для Ozone

Сервис Ozone имеет пять компонентов:

- Ozone Datanode
- Ozone Manager
- Ozone Recon
- Ozone S3G
- Ozone Storage Container Manager

Все эти компоненты имеют одинаковый набор доступных действий, перечисленных в таблице ниже.

Действия с компонентами Ozone

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Подключение к Phoenix

Альберт Багдасарян

Содержание

Назад к содержанию

SQLLine

JDBC

Графический интерфейс

- Шаг 1. Добавление драйвера
- Шаг 2. Добавление соединения
- Шаг 3. Работа с таблицами Phoenix

Существует несколько способов соединения с Phoenix, некоторые из которых приведены в следующих секциях.

SQLLine

SQLLine предоставляет простейший способ для начала работы с Phoenix. Это Java-консоль, доступная на каждом узле кластера HBase сразу после его установки. Начните работу с SQLLine, выполнив следующую команду:

```
$ cd /usr/lib/phoenix/bin
$ ./sqlline.py
```

Консоль SQLLine выводит строку приглашения, заканчивающуюся символом `>`, после которого следует вводить команду Phoenix.

Приглашение к вводу команды в консоли SQLLine

```
Setting property: [incremental, false]
Setting property: [isolation, TRANSACTION_READ_COMMITTED]
issuing: !connect jdbc:phoenix: none none org.apache.phoenix.jdbc.PhoenixDriver
Connecting to jdbc:phoenix:
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-client.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
21/11/30 13:51:00 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
Connected to: Phoenix (version 5.0)
Driver: PhoenixEmbeddedDriver (version 5.0)
Autocommit status: true
Transaction isolation: TRANSACTION_READ_COMMITTED
Building list of tables and columns for tab-completion (set fastconnect to true to skip)...
142/142 (100%) Done
Done
sqlline version 1.2.0
0: jdbc:phoenix:>
```

Для закрытия консоли SQLLine введите следующую команду:

```
!quit
```

Простейшие операции управления данными в Phoenix с помощью консоли SQLLine приведены в разделе Использование SQLLine для работы с Phoenix.

JDBC

Существуют два способа подключения к Phoenix через JDBC:

- **Phoenix Thick Driver**. Его используют для подключения через сервис ZooKeeper (ZK). Полный путь соединения (JDBC URL) имеет следующий вид:

  ```
  jdbc:phoenix:[список узлов ZooKeeper через запятую [: порт ZK [:корневой узел ZK для HBase [:kerberos_principal [:путь к файлу
  keytab для Kerberos]]]]
  ```

  Простейший пример:

  ```
  jdbc:phoenix:bds-adh-1.ru-central1.internal,bds-adh-2.ru-central1.internal,bds-adh-3.ru-central1.internal:2181
  ```

- **Phoenix Thin Driver**. Это отдельный клиент с минимальным набором зависимостей. Он использует возможности специальных серверов запросов (Phoenix Query Server services). В этом случае JDBC URL имеет следующий вид:

  ```
  jdbc:phoenix:thin:[key=value[;key=value...]]
  ```

  Например:

  ```
  jdbc:phoenix:thin:url=http://10.92.6.9:8765;serialization=PROTOBUF
  ```

ПРИМЕЧАНИЕ

Более подробная информация о драйверах JDBC для Phoenix приведена в документации Phoenix.

На практике используют оба типа драйверов. Выбор зависит от топологии кластера, количества серверов запросов Phoenix и возможности добавления зависимостей в клиентское приложение. При пакетной обработке производительность обоих драйверов примерно одинаковая, без использования пакетов производительность Thick Driver обычно на 30% выше.

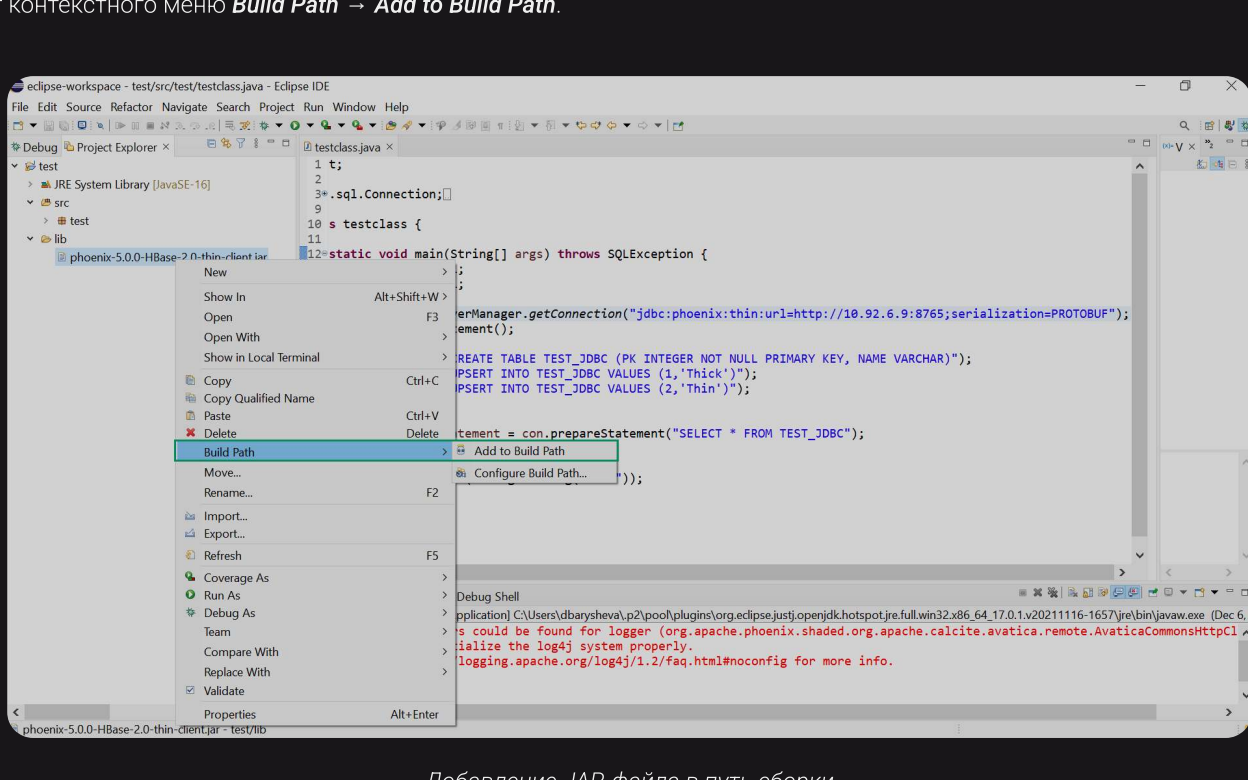
Рассмотрим приложение Java, использующее Phoenix Thin Driver:

1. Создайте класс, используя приведенный ниже код (например в среде разработки Eclipse IDE). Замените `<IP>` на IP-адрес вашего сервера запросов (Phoenix Query Server). Следующий код создает таблицу `TEST_JDBC` с двумя колонками, записывает в них некоторые значения и затем выводит содержимое таблицы в выходной поток:

```java
package test;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.PreparedStatement;
import java.sql.Statement;

public class testclass {

        public static void main(String[] args) throws SQLException {
                Statement stmt = null;
                ResultSet rset = null;

                Connection con = DriverManager.getConnection("jdbc:phoenix:thin:url=http://<IP>:8765;serialization=PROTOBUF");
                stmt = con.createStatement();

                stmt.executeUpdate("CREATE TABLE TEST_JDBC (PK INTEGER NOT NULL PRIMARY KEY, NAME VARCHAR)");
                stmt.executeUpdate("UPSERT INTO TEST_JDBC VALUES (1,'Thick')");
                stmt.executeUpdate("UPSERT INTO TEST_JDBC VALUES (2,'Thin')");
                con.commit();

                PreparedStatement statement = con.prepareStatement("SELECT * FROM TEST_JDBC");
                rset = statement.executeQuery();
                while (rset.next()) {
                        System.out.println(rset.getString("NAME"));
                }
                statement.close();
                con.close();
        }
}
```

2. Определите необходимый вам JAR-файл, название которого содержит версии HBase и Phoenix и имеет вид *phoenix-<phoenix_version>-HBase-<hbase_version>-thin-client.jar*. Например, для версий HBase 2.0 и Phoenix 5.0.0 следует использовать *phoenix-5.0.0-HBase-2.0-thin-client.jar*. Его можно скопировать из узла кластера HBase (из каталога */usr/lib/phoenix*) или загрузить из репозитория Phoenix.

3. Добавьте загруженный файл в путь сборки (build path) приложения Java. При работе в среде Eclipse скопируйте его в каталог */lib* в вашем рабочем пространстве, а затем используйте пункт контекстного меню ***Build Path → Add to Build Path***.



Добавление JAR-файла в путь сборки

4. Запустите ваш класс на исполнение как приложение Java. Пример вывода на экран:

```
log4j:WARN No appenders could be found for logger (org.apache.phoenix.shaded.org.apache.calcite.avatica.remote.
AvaticaCommonsHttpClientImpl).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Thick
Thin
```

ВНИМАНИЕ

Если ваш кластер ADH не использует аутентификацию Kerberos, соединение JDBC будет производиться по умолчанию под пользователем по имени `phoenix`. Этот пользователь должен иметь необходимые права в HBase. Например, если необходимо только создавать таблицы и записывать данные в них, расширьте полномочия этого пользователя с помощью следующей команды в среде HBase shell:

```
grant 'phoenix', 'WC'
```

Если этого не сделать, то при попытке создать таблицу возникнет ошибка со следующим сообщением:

```
Insufficient permissions (user=phoenix, scope=default, params=
[namespace=default,table=default:TEST_JDBC,family=0],action=CREATE)
Insufficient permissions (user=phoenix, scope=default, params=[table=default:SYSTEM.CATALOG, family=0:USE_STATS_FOR_PARALLELIZATION,
params=[table=default:SYSTEM.CATALOG,family=0:USE_STATS_FOR_PARALLELIZATION],action=WRITE)
```

Более подробная информация о примененной команде приведена в описании grant.

5. Проверьте наличие таблицы `TEST_JDBC` в Phoenix:

```
!tables
```

Вывод на экран:

```
+------------+--------------+-------------+---------------+----------+--------+
| TABLE_CAT  | TABLE_SCHEM  | TABLE_NAME  |  TABLE_TYPE   | REMARKS  | TYPE_N |
+------------+--------------+-------------+---------------+----------+--------+
|            | SYSTEM       | CATALOG     | SYSTEM TABLE  |          |        |
|            | SYSTEM       | FUNCTION    | SYSTEM TABLE  |          |        |
|            | SYSTEM       | LOG         | SYSTEM TABLE  |          |        |
|            | SYSTEM       | SEQUENCE    | SYSTEM TABLE  |          |        |
|            | SYSTEM       | STATS       | SYSTEM TABLE  |          |        |
|            |              | BEST_BOOKS  | TABLE         |          |        |
|            |              | TEST_JDBC   | TABLE         |          |        |
|            |              | employee    | TABLE         |          |        |
|            |              | people_ages | TABLE         |          |        |
+------------+--------------+-------------+---------------+----------+--------+
0: jdbc:phoenix:> select * from TEST_JDBC;
+-----+--------+
| PK  |  NAME  |
+-----+--------+
| 1   | Thick  |
| 2   | Thin   |
+-----+--------+
2 rows selected (0.058 seconds)
```

Графический интерфейс

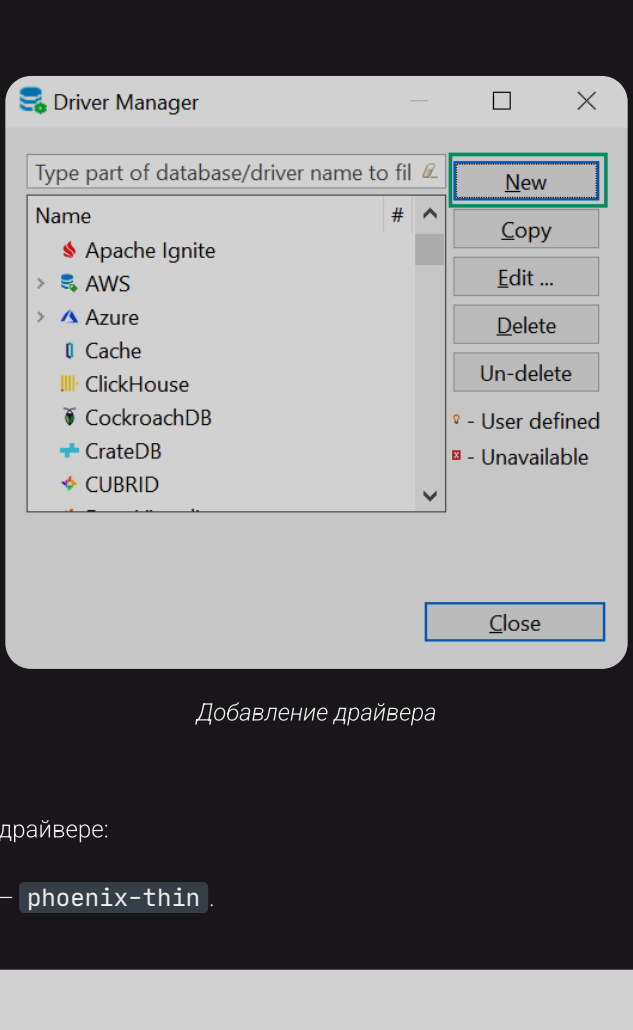
Используя драйверы JDBC, можно также подключиться к Phoenix через инструменты с графическим интерфейсом. Рассмотрим пример применения Phoenix Thin Driver в DBeaver.

ПРИМЕЧАНИЕ

Данная секция служит только в качестве примера. Если необходимо использовать графический интерфейс, мы настоятельно рекомендуем обратить внимание на Zeppelin, поставляемый с пакетом ADH и специально разработанный для аналитиков больших данных. Информация о подключении Zeppelin к Phoenix приведена в разделе Добавление интерпретатора в группу.

Шаг 1. Добавление драйвера

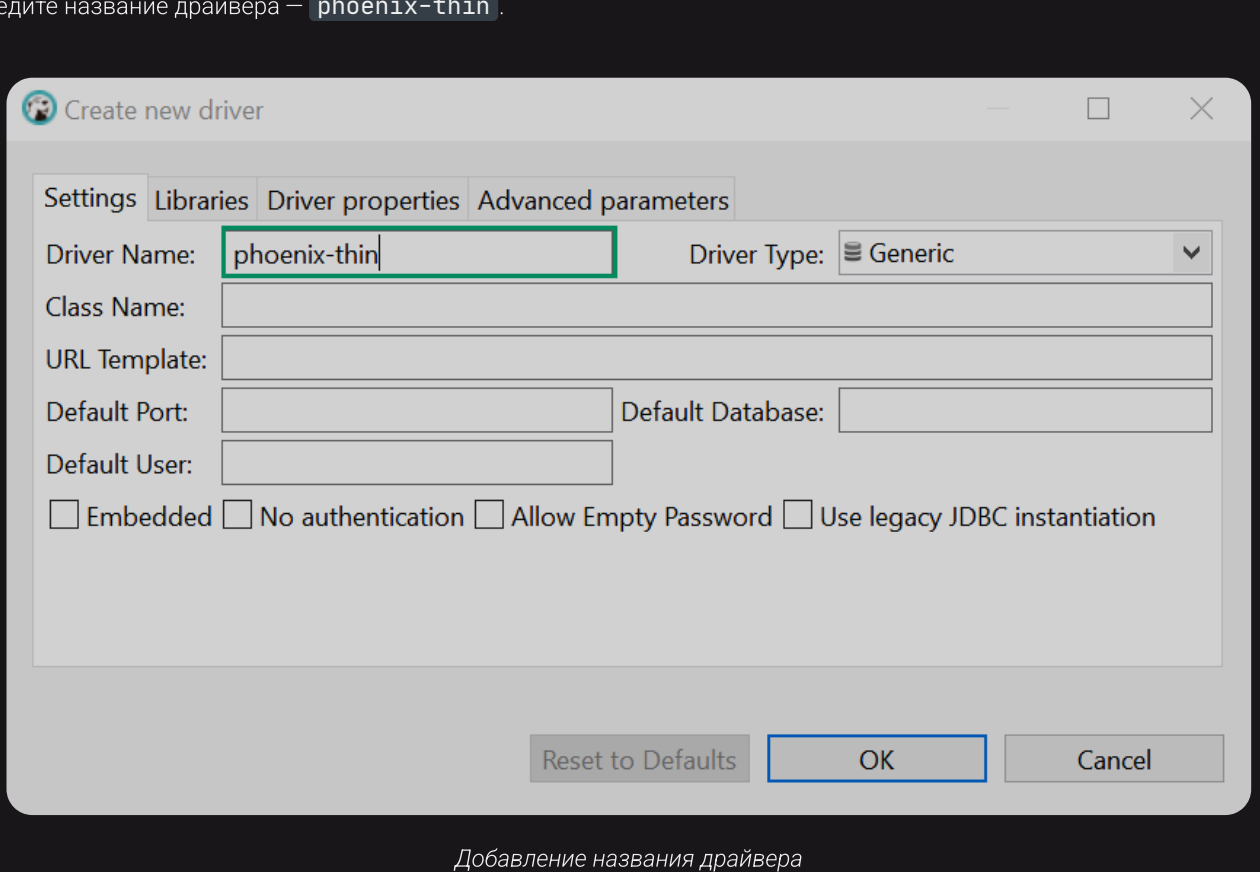
1. Установите DBeaver на вашем локальном компьютере.
2. Определите необходимый вам JAR-файл, название которого содержит версии HBase и Phoenix и имеет вид *phoenix-<phoenix_version>-HBase-<hbase_version>-thin-client.jar*. Например, для версий HBase 2.0 и Phoenix 5.0.0 следует использовать *phoenix-5.0.0-HBase-2.0-thin-client.jar*. Его можно скопировать из узла кластера HBase (из каталога */usr/lib/phoenix*) или загрузить из репозитория Phoenix.
3. Поместите загруженный файл в папку *plugins* внутри рабочего каталога DBeaver. Например, полный путь в Windows — *C:/Program Files/DBeaver/plugins/*.
4. В главном меню DBeaver выберите ***Database → Driver Manager***
5. В открывшейся форме кликните ***New***.



Добавление драйвера

6. В открывшемся окне введите информацию о новом драйвере:

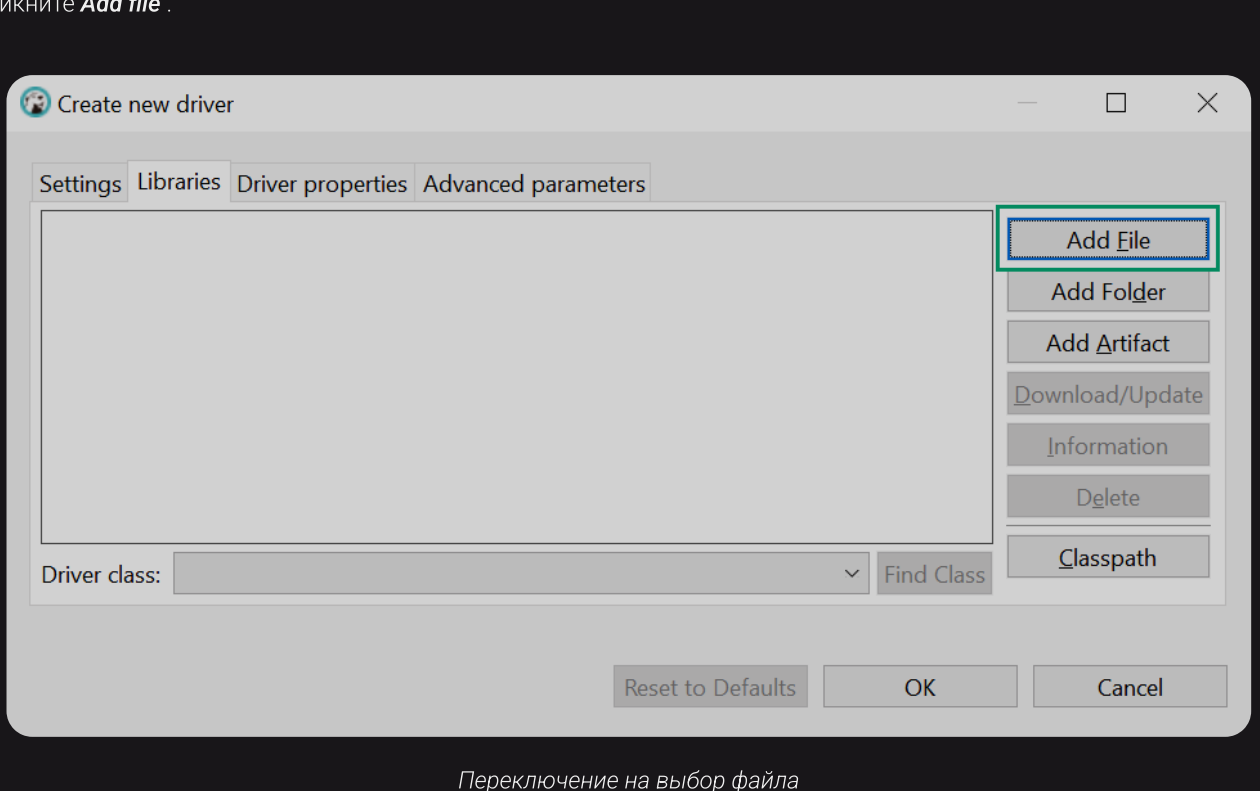
   - Во вкладке ***Settings*** введите название драйвера — `phoenix-thin`.



Driver Name: phoenix-thin
Driver Type: Generic

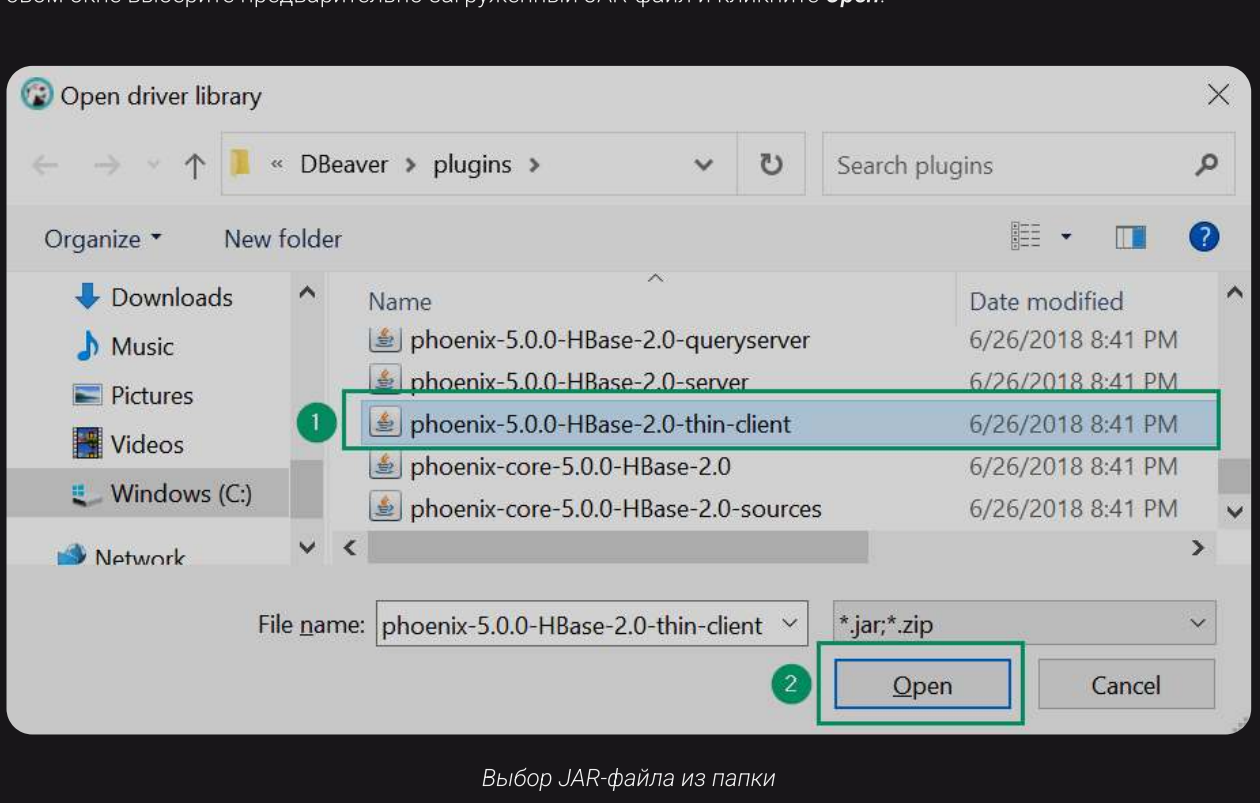
Добавление названия драйвера

   - Во вкладке ***Libraries*** кликните ***Add file***.



Переключение на выбор файла

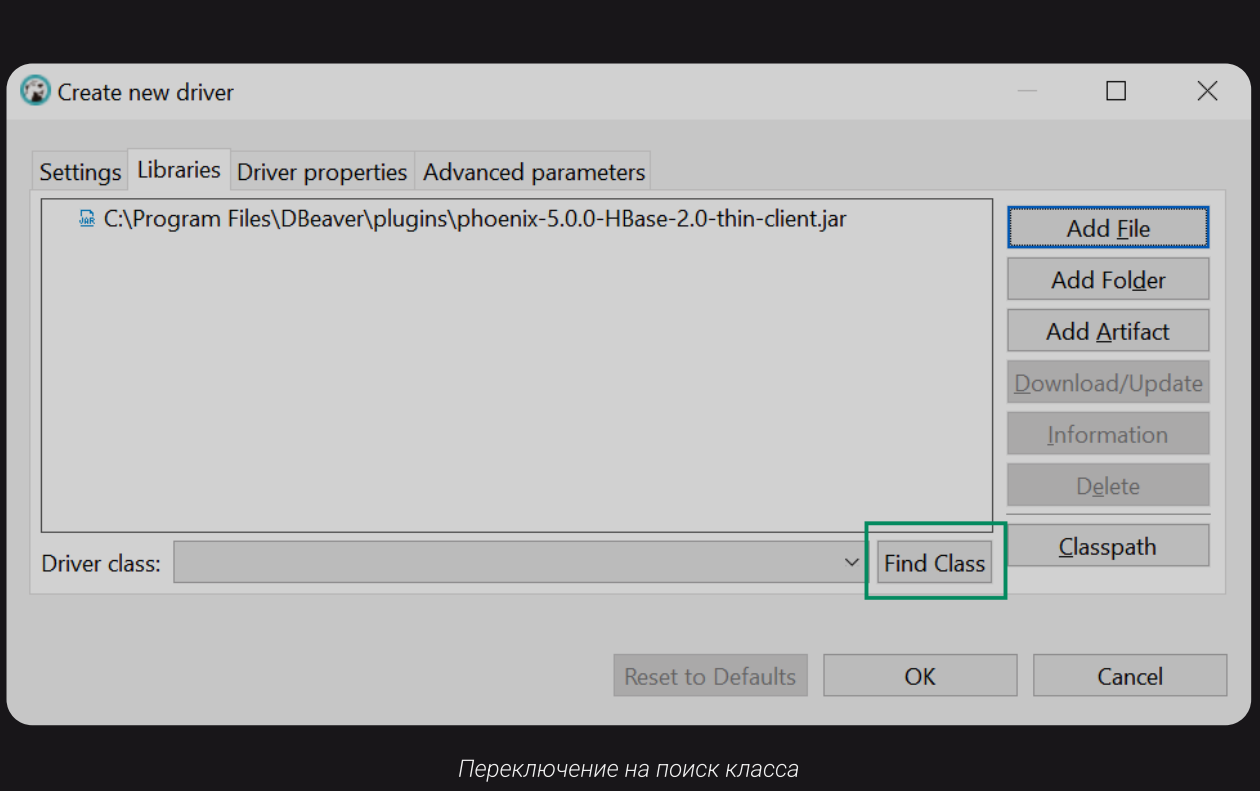
   - В открывшемся диалоговом окне выберите предварительно загруженный JAR-файл и кликните ***Open***.


Open driver library
DBeaver > plugins
phoenix-5.0.0-HBase-2.0-queryserver
phoenix-5.0.0-HBase-2.0-server
phoenix-5.0.0-HBase-2.0-thin-client
phoenix-core-5.0.0-HBase-2.0
phoenix-core-5.0.0-HBase-2.0-sources
1
2
Open
Cancel

File name: phoenix-5.0.0-HBase-2.0-thin-client

Выбор JAR-файла из папки

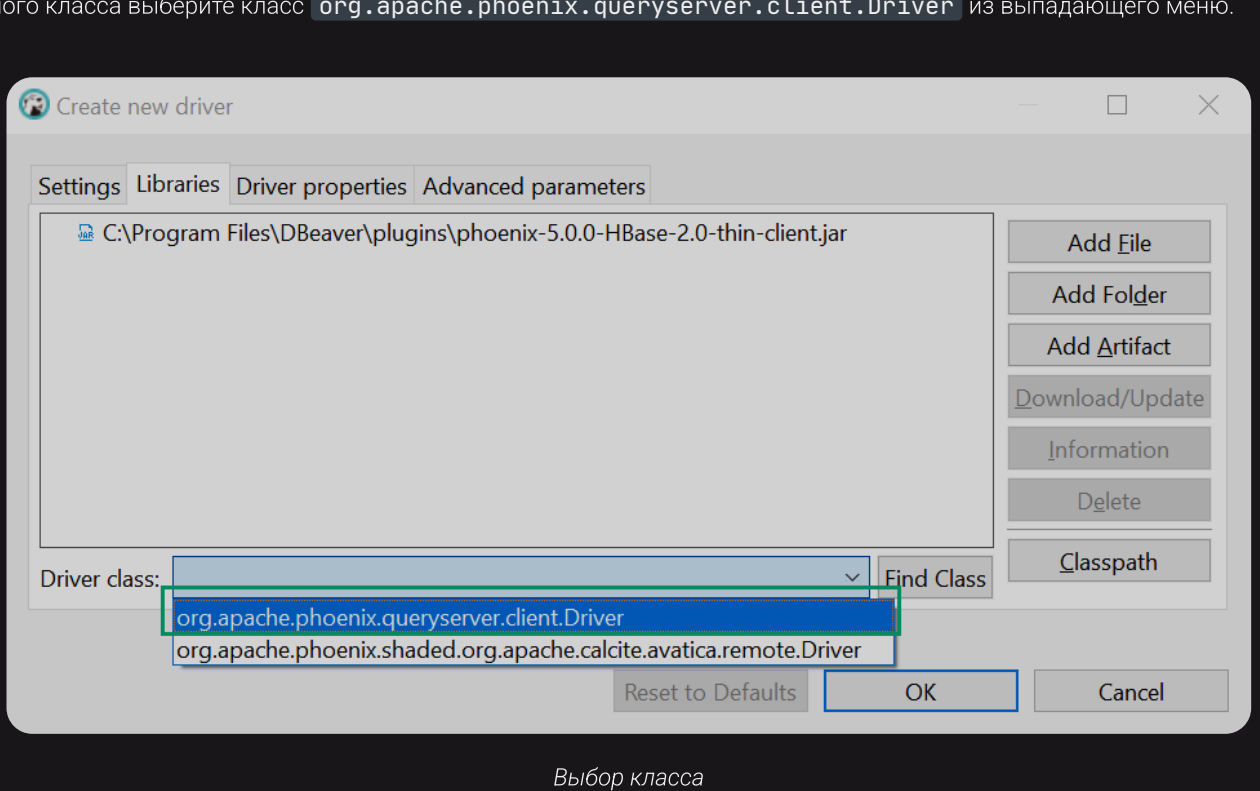
   - Кликните ***Find Class***.



C:\Program Files\DBeaver\plugins\phoenix-5.0.0-HBase-2.0-thin-client.jar

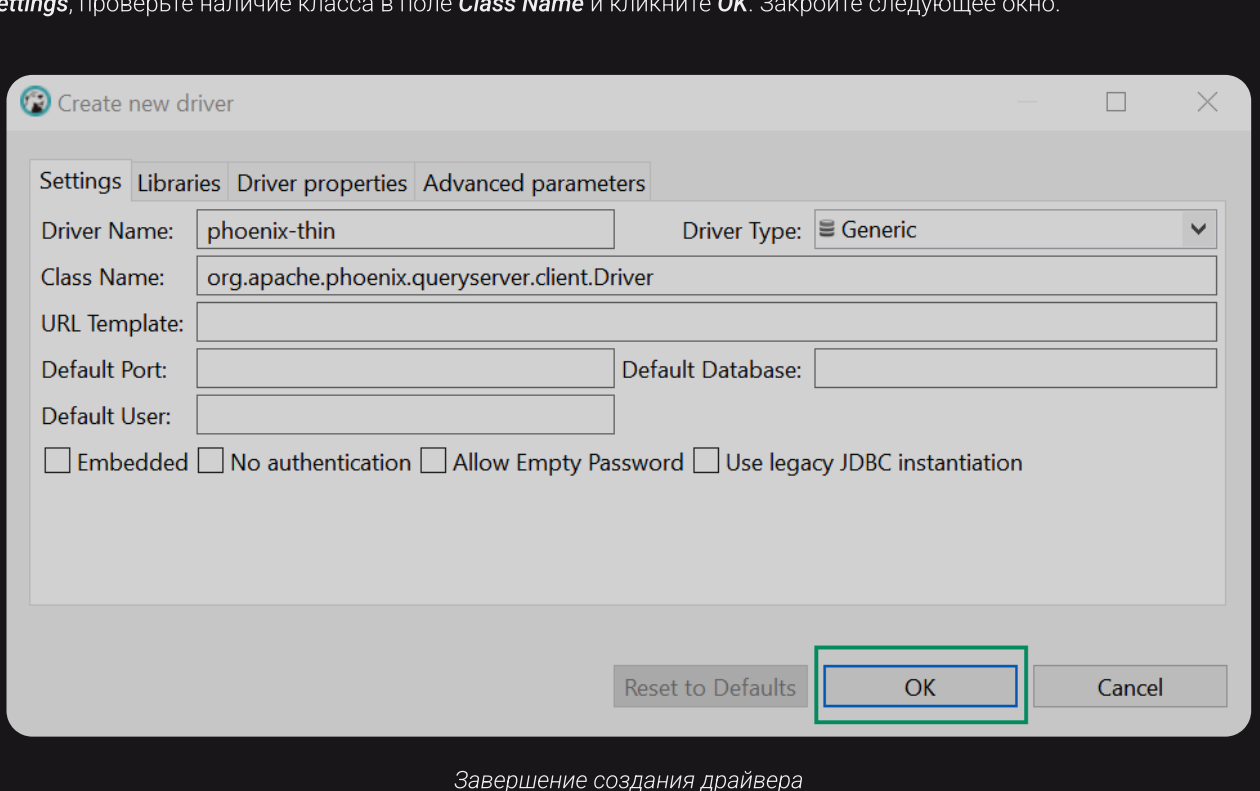
Переключение на поиск класса

   - После выбора требуемого класса выберите класс `org.apache.phoenix.queryserver.client.Driver` из выпадающего меню.


org.apache.phoenix.queryserver.client.Driver
org.apache.phoenix.shaded.org.apache.calcite.avatica.remote.Driver

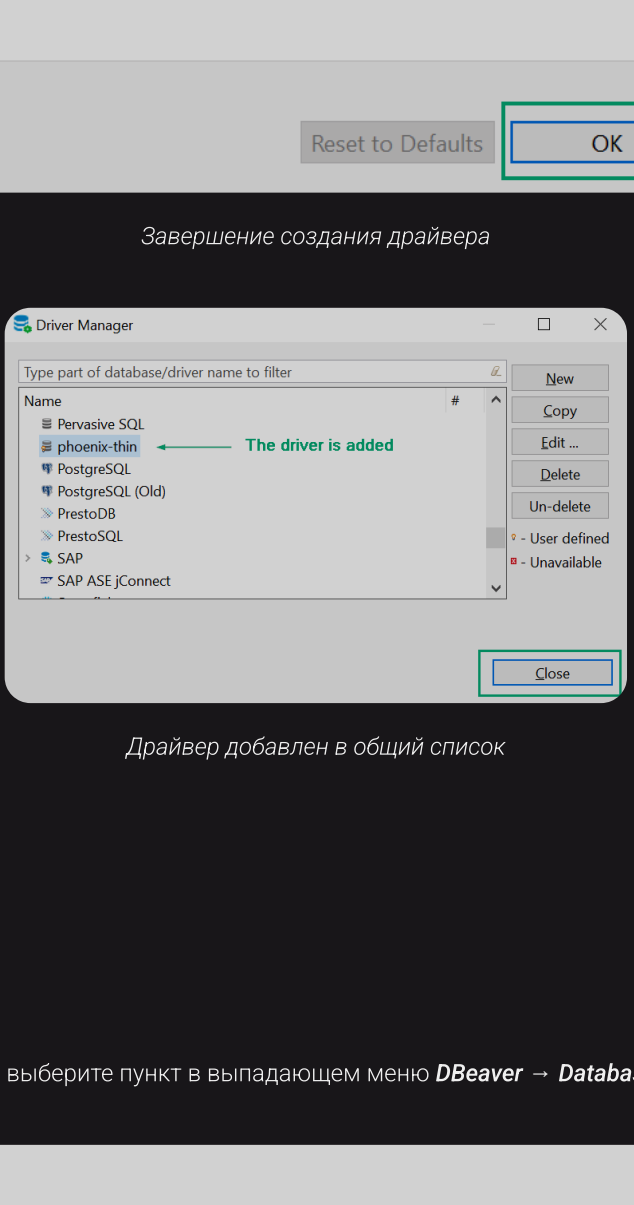
Выбор класса

   - Вернитесь к вкладке ***Settings***, проверьте наличие класса в поле ***Class Name*** и кликните ***OK***. Закройте следующее окно.



Driver Name: phoenix-thin
Driver Type: Generic
Class Name: org.apache.phoenix.queryserver.client.Driver

Завершение создания драйвера

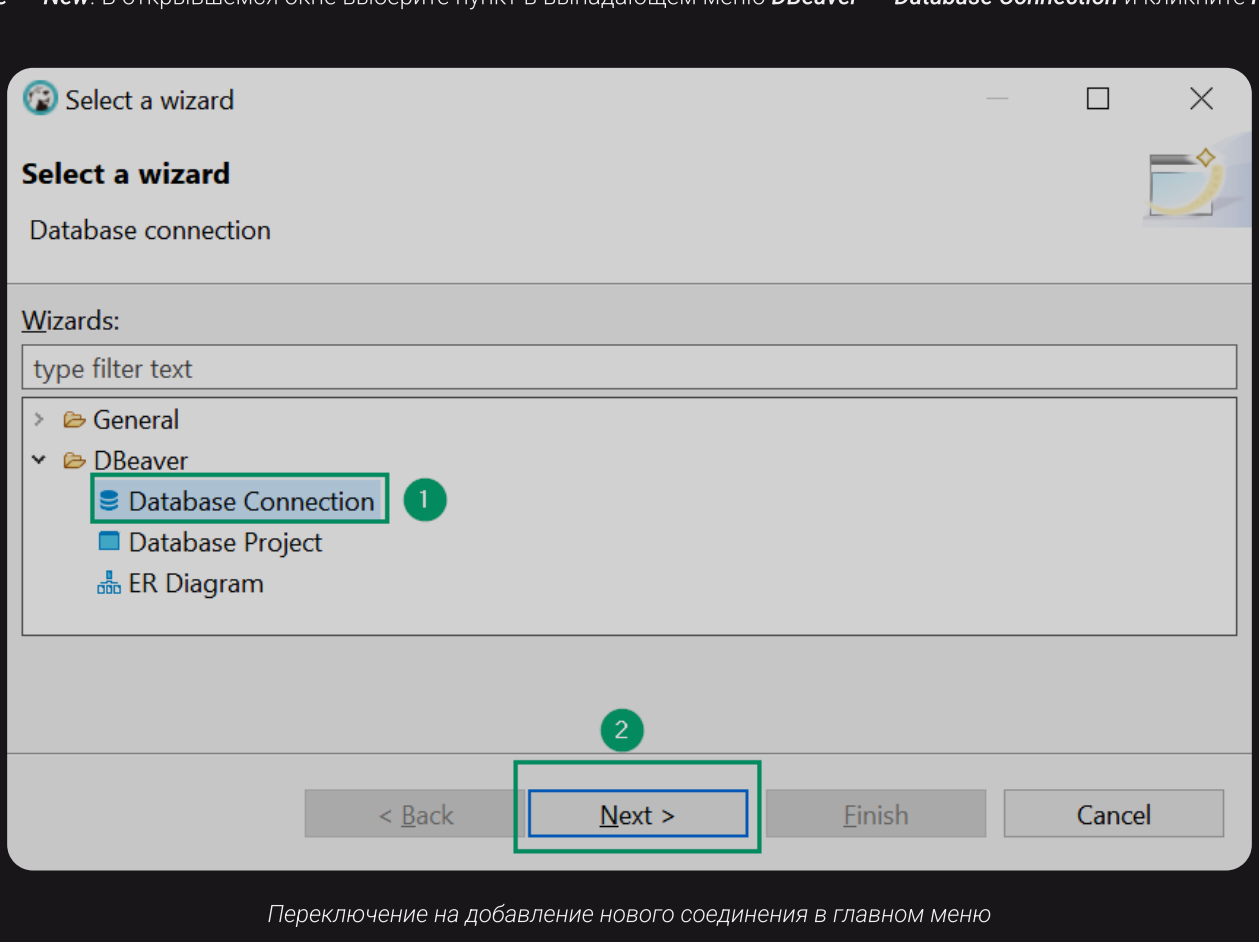


Драйвер добавлен в общий список

Новый драйвер готов к применению.

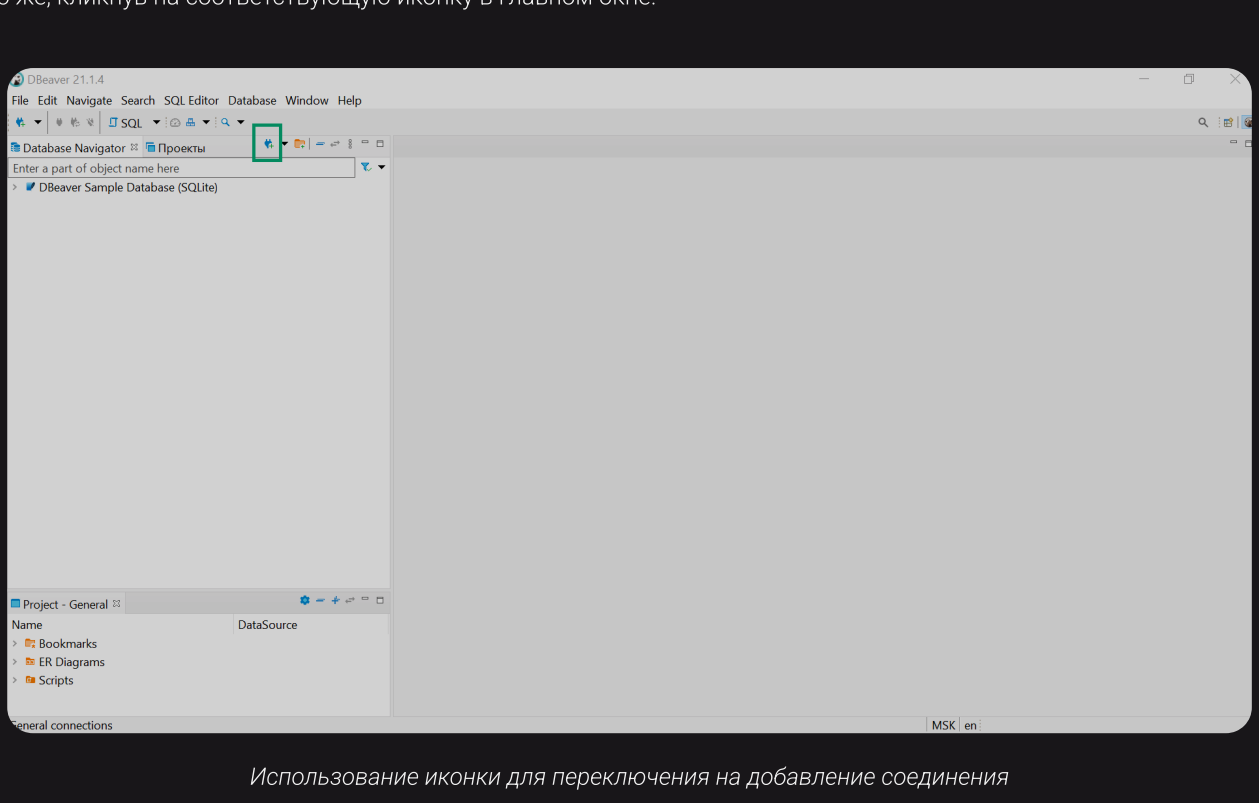
Шаг 2. Добавление соединения

1. Откройте пункт меню ***File → New***. В открывшемся окне выберите пункт в выпадающем меню ***DBeaver → Database Connection*** и кликните ***Next***.



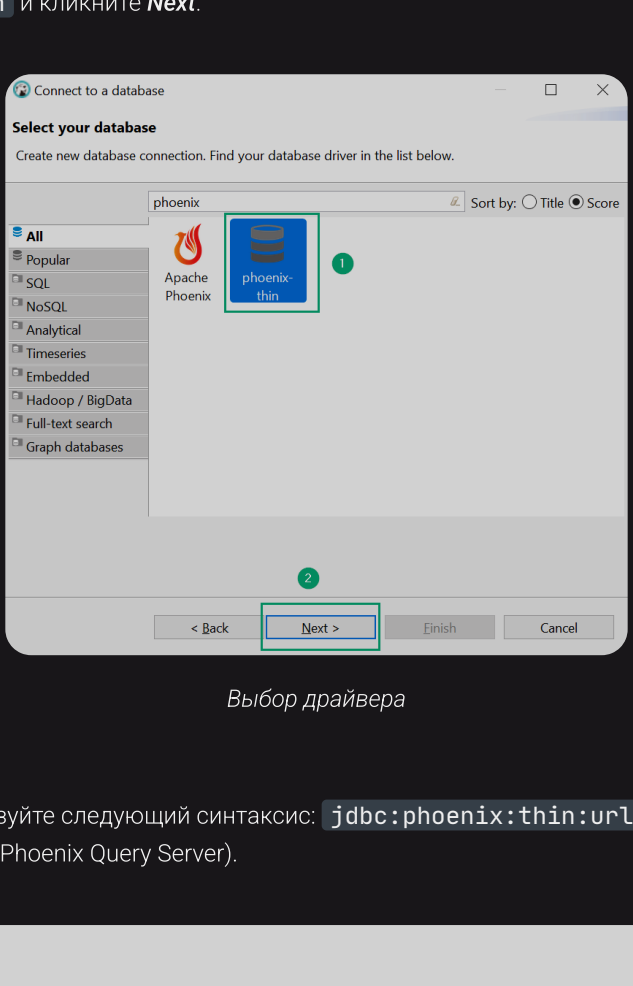
Переключение на добавление нового соединения в главном меню

Вы можете добиться того же, кликнув на соответствующую иконку в главном окне.



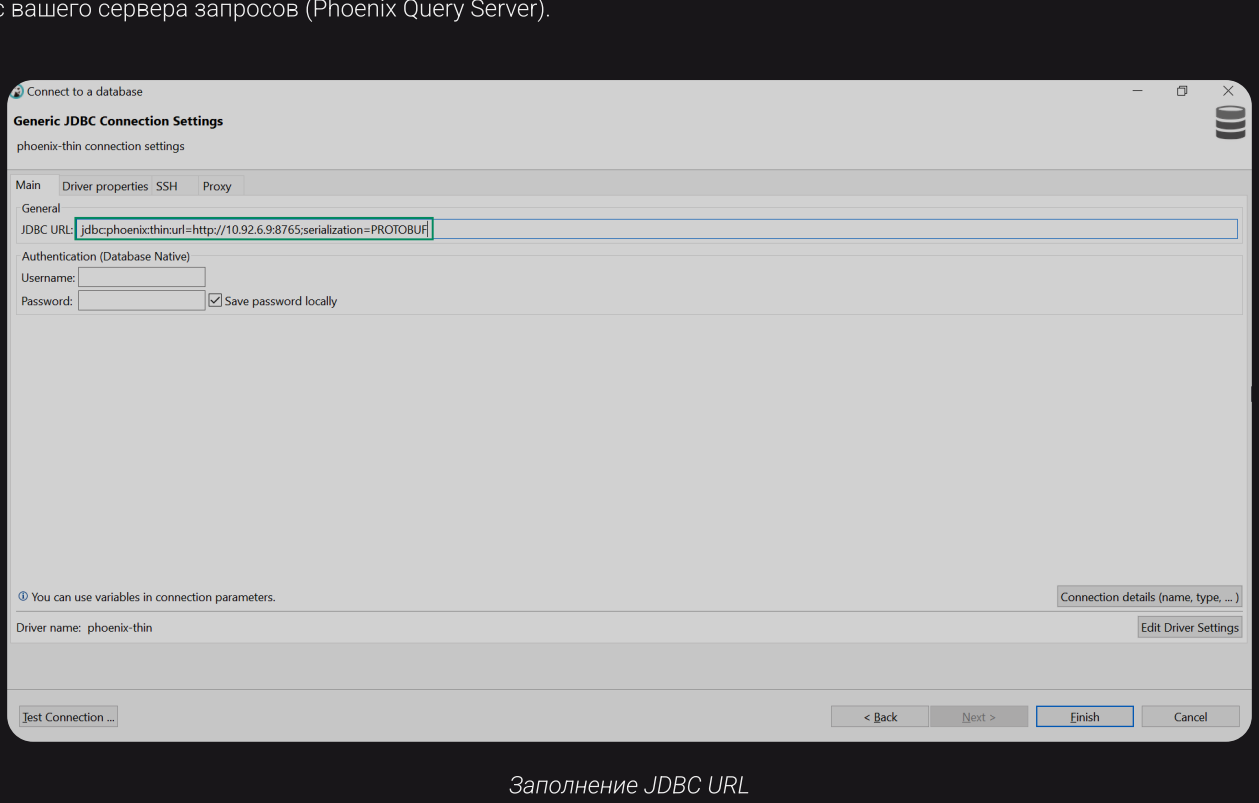
Использование иконки для переключения на добавление соединения

2. Выберите созданный ранее драйвер `phoenix-thin` и кликните ***Next***.



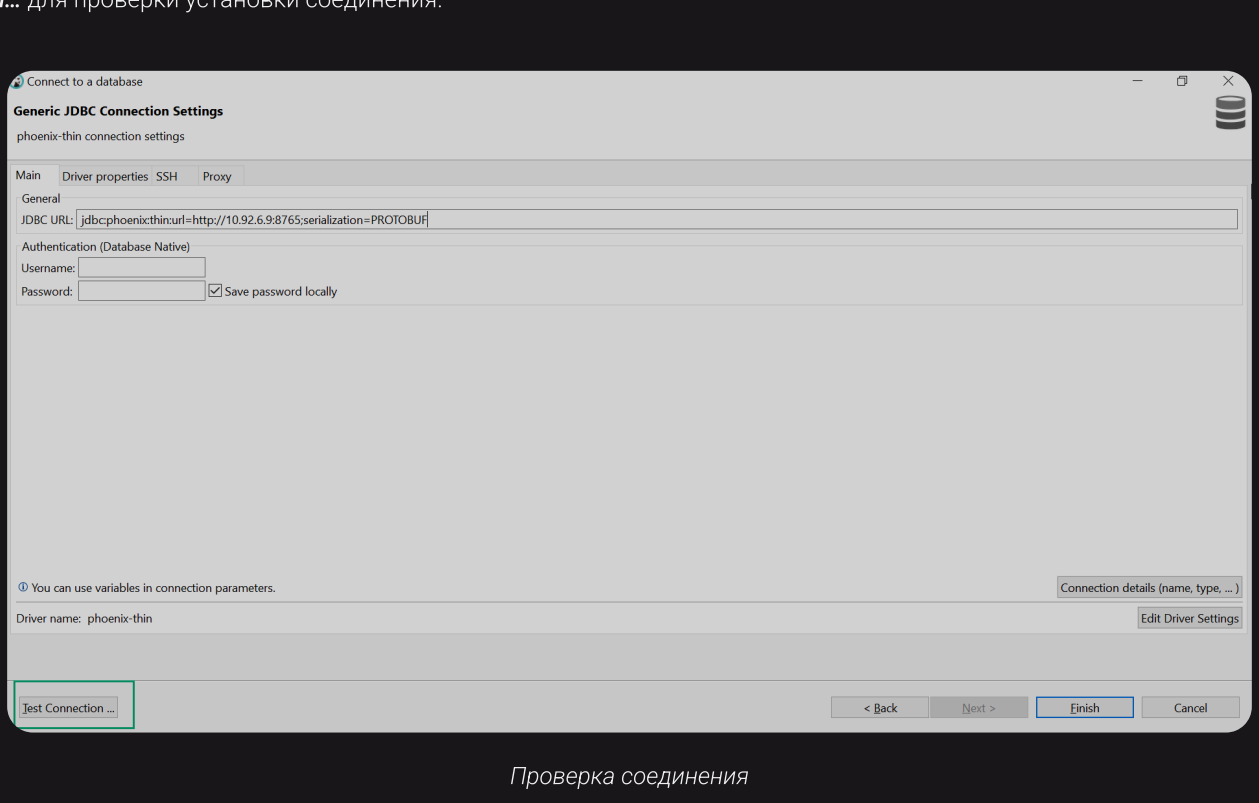
Выбор драйвера

3. В поле ***JDBC URL*** введите адрес соединения. Используйте следующий синтаксис: `jdbc:phoenix:thin:url=http://<IP>:8765;serialization=PROTOBUF`, где `<IP>` означает IP-адрес вашего сервера запросов (Phoenix Query Server).



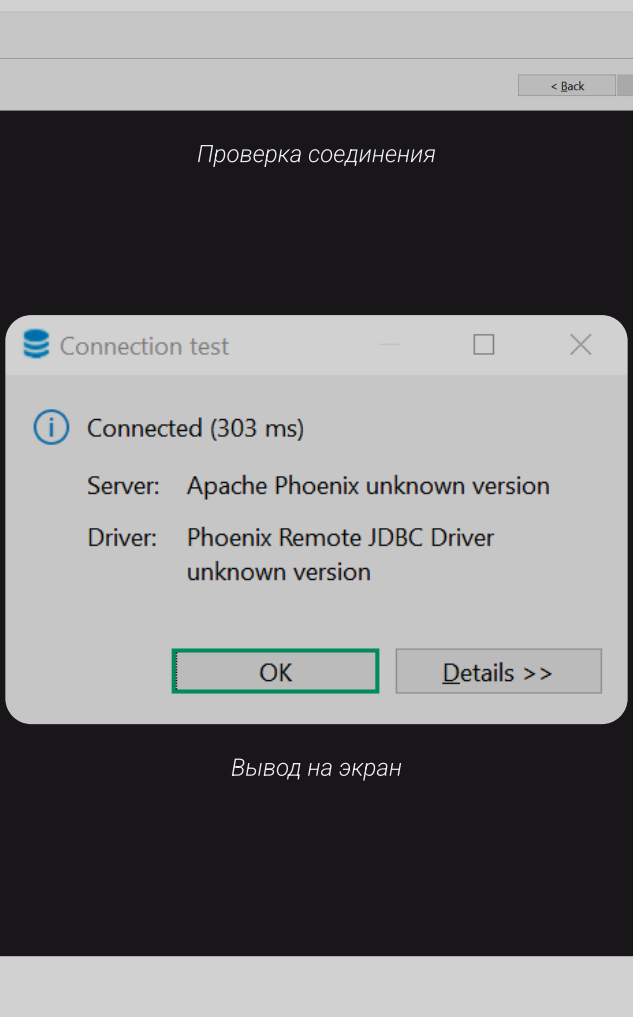
Заполнение JDBC URL

4. Кликните ***Test Connection...*** для проверки установки соединения.



Проверка соединения

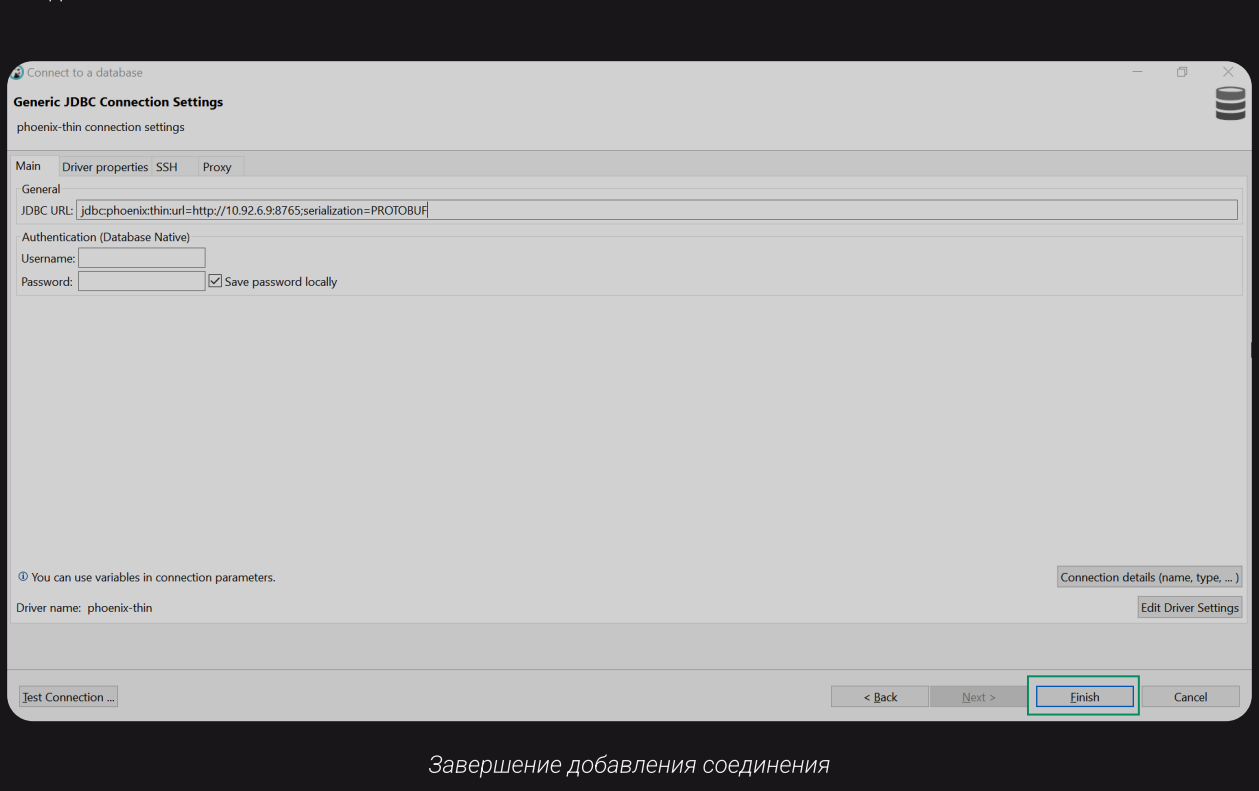
Пример вывода результата на экран:



Connected (303 ms)
Server: Apache Phoenix unknown version
Driver: Phoenix Remote JDBC Driver unknown version

Вывод на экран

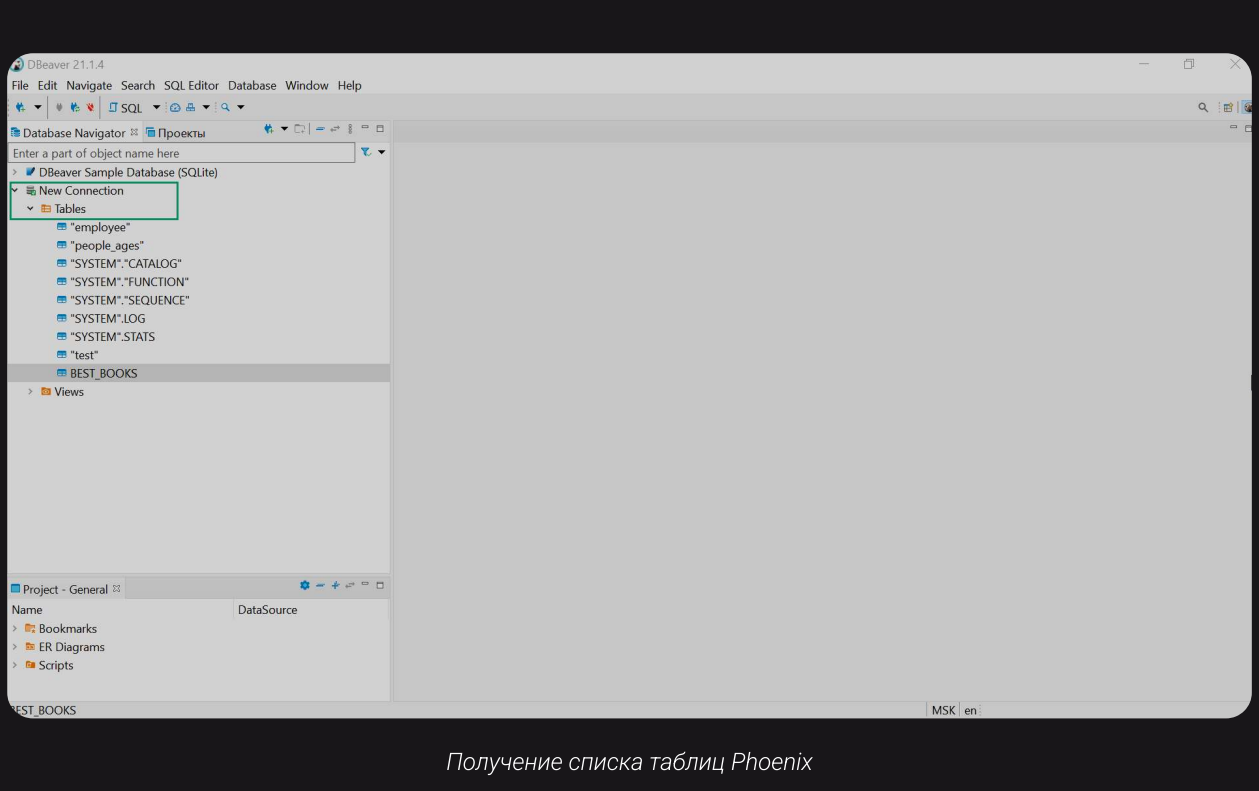
5. Кликните ***Finish***. Соединение добавлено.



Завершение добавления соединения

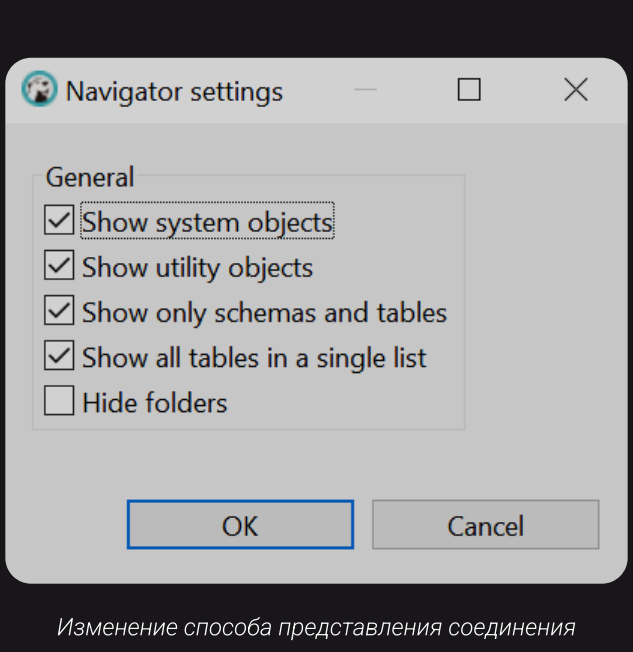
Шаг 3. Работа с таблицами Phoenix

После установки соединения соответствующая ему строка покажется в списке соединений слева от главного окна. Развернув пункт ***Tables***, вы увидите все таблицы Phoenix.



Получение списка таблиц Phoenix

Если таблицы не показываются и при этом отсутствуют ошибки, попробуйте настроить собственный способ представления путем выбора в контекстном меню пункта ***Connection View → Custom...*** и настройки флагов.



General
☑ Show system objects
☑ Show utility objects
☑ Show only schemas and tables
☑ Show all tables in a single list
☐ Hide folders

Изменение способа представления соединения

ВНИМАНИЕ

Если в вашем кластере ADH отсутствует аутентификация Kerberos, соединение JDBC будет использовать по умолчанию имя пользователя `phoenix`. Этот пользователь должен обладать необходимыми правами в HBase. Например, если вам необходимо только читать данные с помощью DBeaver, обеспечьте право на чтение с помощью следующей команды в HBase shell:

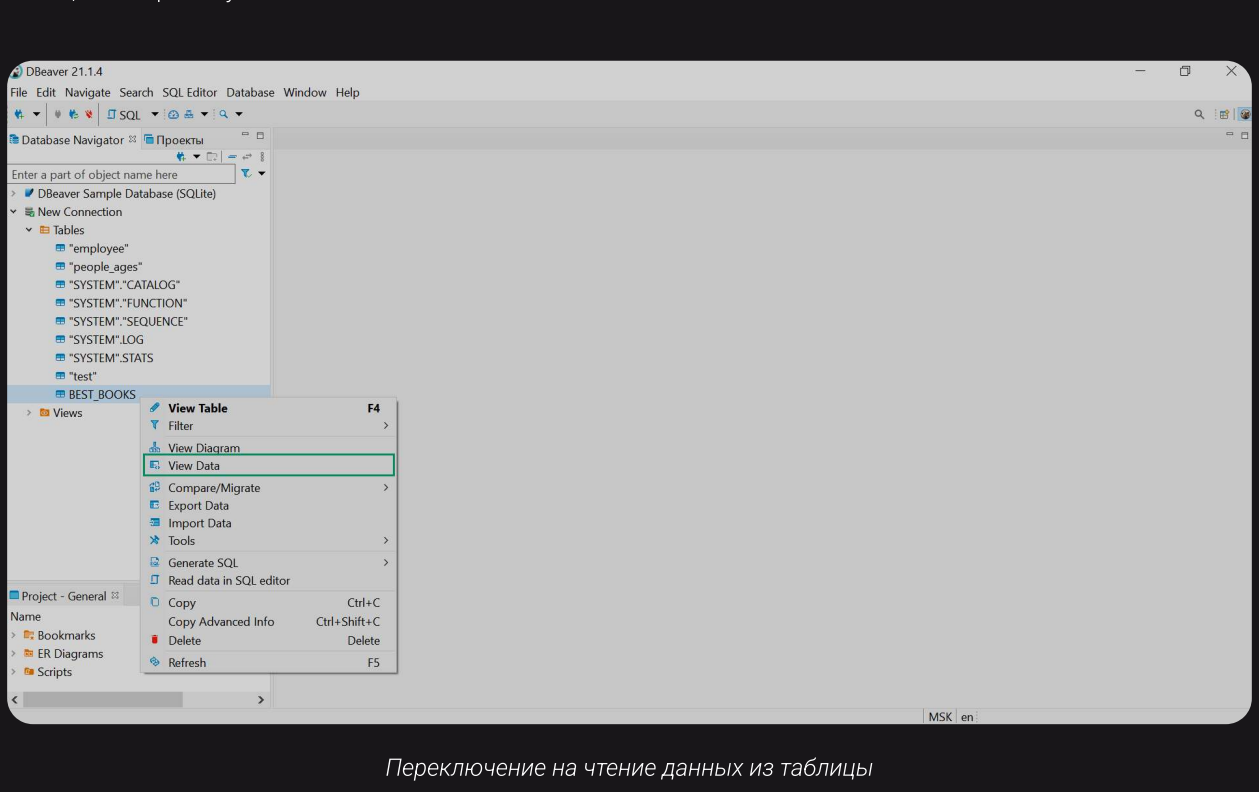
```
grant 'phoenix', 'R'
```

Если этого не сделать, то возникнет ошибка при попытке DBeaver прочитать таблицу с выводом следующего сообщения:

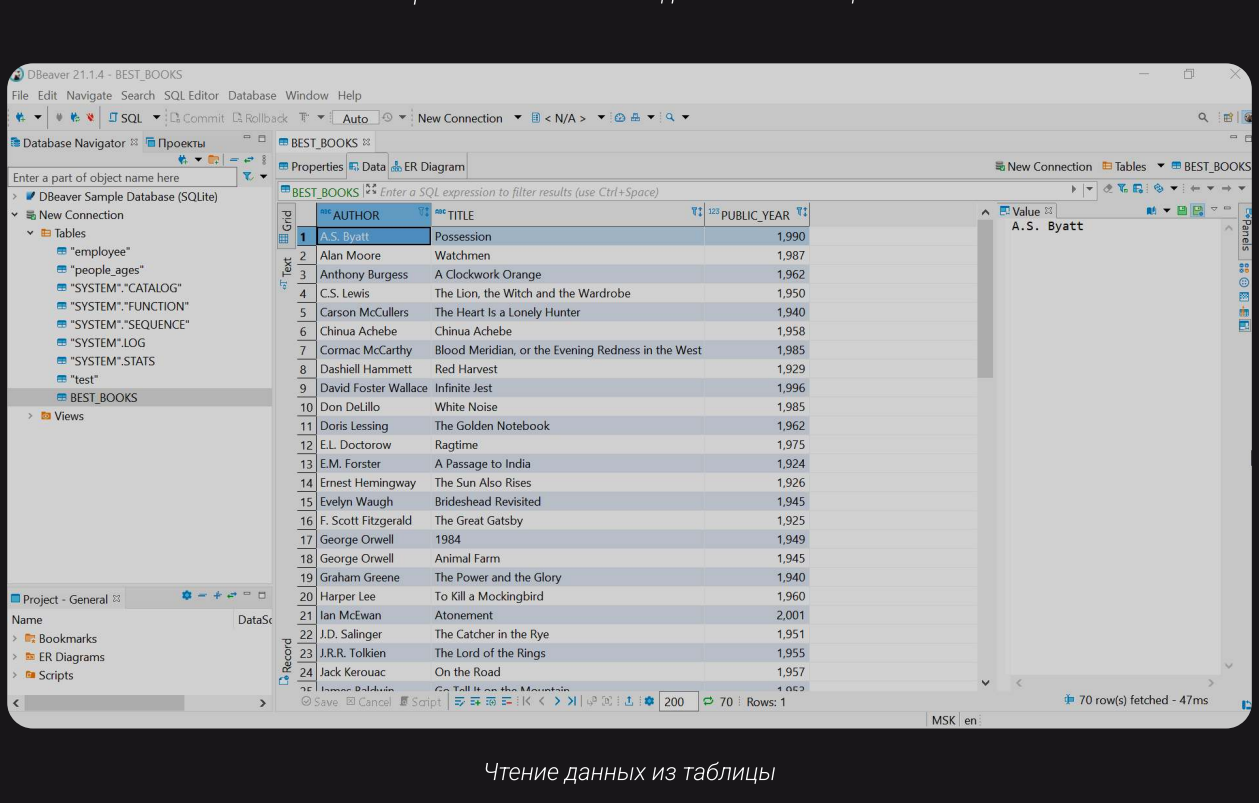
```
Insufficient permissions for user 'phoenix' (table=SYSTEM.CATALOG, action=READ).
```

Более подробная информация об использованной выше команде приведена в описании grant.

Для получения содержимого таблицы выберите пункт ***View Data*** в контекстном меню.



Переключение на чтение данных из таблицы

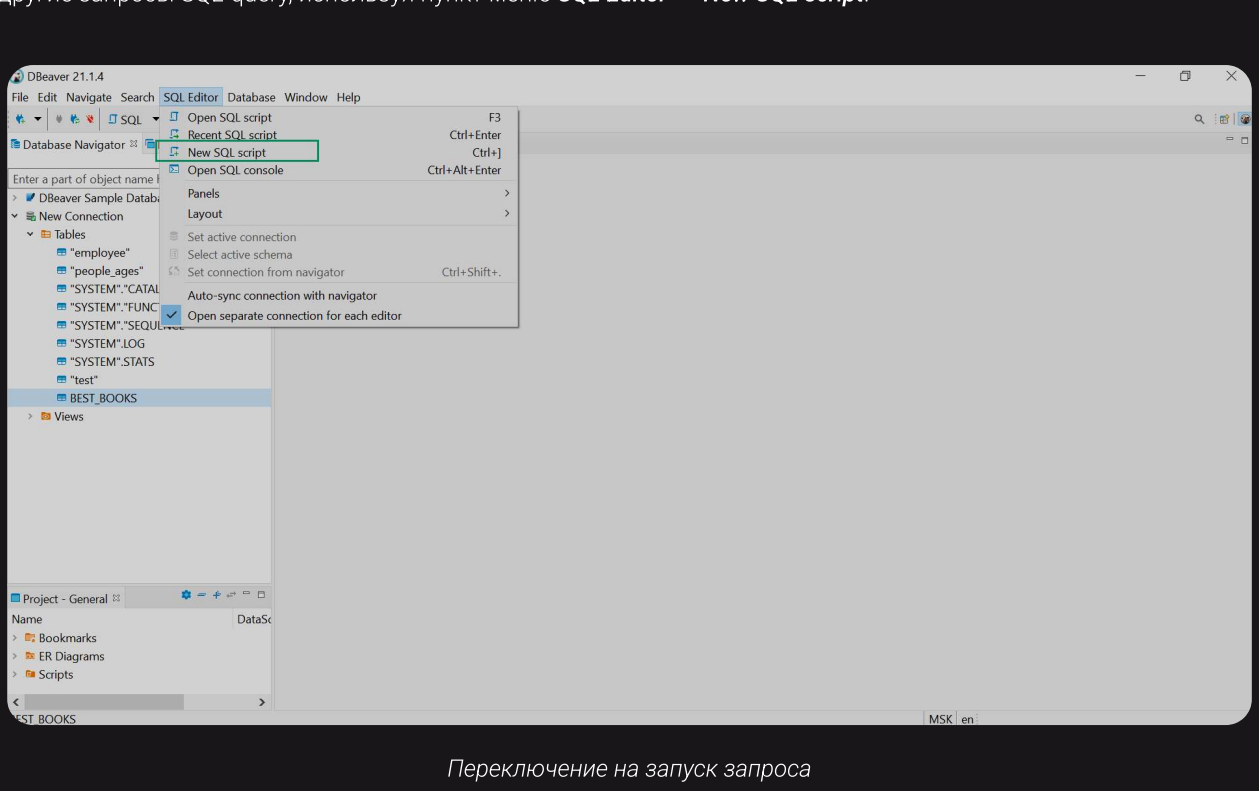


	AUTHOR	TITLE	PUBLIC_YEAR
1	A.S. Byatt	Possession	1,990
2	Alan Moore	Watchmen	1,987
3	Anthony Burgess	A Clockwork Orange	1,962
4	C.S. Lewis	The Lion, the Witch and the Wardrobe	1,950
5	Carson McCullers	The Heart Is a Lonely Hunter	1,940
6	Chinua Achebe	Chinua Achebe	1,958
7	Cormac McCarthy	Blood Meridian, or the Evening Redness in the West	1,985
8	Dashiell Hammett	Red Harvest	1,929
9	David Foster Wallace	Infinite Jest	1,996
10	Don DeLillo	White Noise	1,985
11	Doris Lessing	The Golden Notebook	1,962
12	E.L. Doctorow	Ragtime	1,975
13	E.M. Forster	A Passage to India	1,924
14	Ernest Hemingway	The Sun Also Rises	1,926
15	Evelyn Waugh	Brideshead Revisited	1,945
16	F. Scott Fitzgerald	The Great Gatsby	1,925
17	George Orwell	1984	1,949
18	George Orwell	Animal Farm	1,945
19	Graham Greene	The Power and the Glory	1,940
20	Harper Lee	To Kill a Mockingbird	1,960
21	Ian McEwan	Atonement	2,001
22	J.D. Salinger	The Catcher in the Rye	1,951
23	J.R.R. Tolkien	The Lord of the Rings	1,955
24	Jack Kerouac	On the Road	1,957

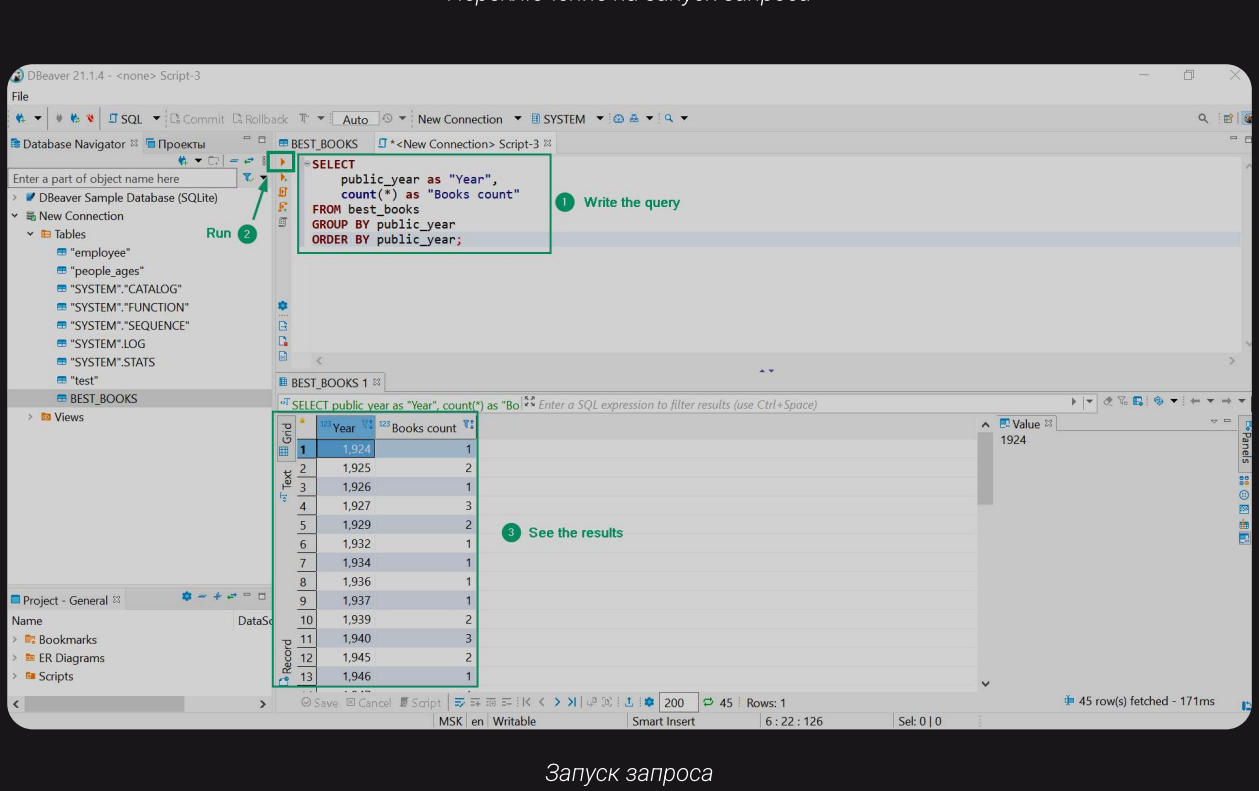
70 row(s) fetched - 47ms

Чтение данных из таблицы

Вы также можете выполнить другие запросы SQL-query, используя пункт меню ***SQL Editor → New SQL script***.


DBeaver 21.1.4
Open SQL script F3
Recent SQL script Ctrl+Enter
New SQL script Ctrl+]
Open SQL console Ctrl+Alt+Enter
Panels
Layout
Set active connection
Select active schema
Set connection from navigator Ctrl+Shift+.
Auto-sync connection with navigator
Open separate connection for each editor

Переключение на запуск запроса



```
SELECT
    public_year as "Year",
    count(*) as "Books count"
FROM best_books
GROUP BY public_year
ORDER BY public_year;
```

	Year	Books count
1	1,924	1
2	1,925	2
3	1,926	1
4	1,927	3
5	1,929	2
6	1,932	1
7	1,934	1
8	1,936	1
9	1,937	1
10	1,939	2
11	1,940	3
12	1,945	2
13	1,946	1

45 row(s) fetched - 171ms

Запуск запроса

Использование SQLLine для работы с Phoenix

Альберт Багдасарян

Содержание

Назад к содержанию

Шаг 1. Создание таблицы

Шаг 2. Получение информации о таблице

Шаг 3. Добавление данных в таблицу

Шаг 4. Выбор данных из таблицы

Шаг 5. Обновление данных в таблице

Шаг 6. Удаление данных из таблицы

Шаг 7. Изменение структуры таблицы

Шаг 8. Удаление таблицы

Секция SQLLine описывает способ подключения к Phoenix с помощью SQLLine. Основные операции с данными, предоставляемые Phoenix при работе в среде SQLLine, приведены ниже. Они основаны на примере, приведенном в разделе Модель данных HBase. Мы используем этот пример для демонстрации работы как в HBase, так и в Phoenix — для понимания сходства и различия между ними.

Полный список этих команд и их описание приведены в документации Phoenix.

ПРИМЕЧАНИЕ

Перед тем как начать выполнять данные команды, рекомендуем ознакомиться с моделью данных HBase и базовыми операциями с данными в HBase.

Шаг 1. Создание таблицы

Большинство синтаксических конструкций Phoenix соотвествуют SQL. Для создания таблицы используйте ключевые слова CREATE TABLE, затем введите название таблицы в двойных кавычках и описание колонок с указанием их типов в круглых скобках. Если необходимо распределить колонки по разным семействам в HBase, используйте комбинированное название для каждой колонки, состоящее из разделенных точкой названия семейства колонок и квалификатора колонки, как продемонстрировано в примере ниже. Название семейства колонок по умолчанию — `0`. После описания колонок можно указать общие параметры таблицы.

Следующая команда создает таблицу `articles` с первичным ключом (primary key) `row_id`, который будет использован в HBase в качестве ключа строки (row key). Кроме того, команда определяет две колонки (`author` и `header`) внутри семейства `basic` и четыре колонки (`arch`, `concepts`, `tutorials` и `ref`) в семействе `tags`. Она также устанавливает, что колонки могу хранить до пяти версий значений.

РЕКОМЕНДАЦИЯ

Заключайте в двойные кавычки `"` названия таблиц, названия семейств колонок и квалификаторы колонок в данной и последующих операциях. В противном случае Phoenix будет конвертировать все символы в верхний регистр. Например, `ARTICLES` вместо `articles`.

```
CREATE TABLE "articles" ("row_id" VARCHAR PRIMARY KEY, "basic"."author" VARCHAR, "basic"."header" VARCHAR, "tags"."arch" BOOLEAN,
"tags"."concepts" BOOLEAN, "tags"."tutorials" BOOLEAN, "tags"."ref" BOOLEAN) VERSIONS=5;
```

Вывод на экран:

```
No rows affected (1.313 seconds)
```

Шаг 2. Получение информации о таблице

Для проверки наличия таблицы и получения ее описания можно использовать одну из следующих команд в SQLLine:

- !tables — возвращает список всех таблиц в Phoenix:

```
!tables
```

Вывод на экран:

```
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------------+--------------+--------------+-----------------+------------+-------------+----------------+-
---------------------+--------------------+----------------------+
| TABLE_CAT  | TABLE_SCHEM  | TABLE_NAME  |  TABLE_TYPE   | REMARKS  | TYPE_NAME  | SELF_REFERENCING_COL_NAME  | REF_GENERATION
| INDEX_STATE  | IMMUTABLE_ROWS  | SALT_BUCKETS  | MULTI_TENANT  | VIEW_STATEMENT  | VIEW_TYPE  | INDEX_TYPE  | TRANSACTIONAL  |
IS_NAMESPACE_MAPPED  | GUIDE_POSTS_WIDTH  | TRANSACTION_PROVIDER  |
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------------+--------------+--------------+-----------------+------------+-------------+----------------+-
---------------------+--------------------+----------------------+
|            | SYSTEM       | CATALOG     | SYSTEM TABLE  |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
|            | SYSTEM       | FUNCTION    | SYSTEM TABLE  |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
|            | SYSTEM       | LOG         | SYSTEM TABLE  |          |            |                            |
|              | true            | 32            | false         |                 |            |             | false          |
false                | null               |                      |
|            | SYSTEM       | SEQUENCE    | SYSTEM TABLE  |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
|            | SYSTEM       | STATS       | SYSTEM TABLE  |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
|            |              | articles    | TABLE         |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------------+--------------+--------------+-----------------+------------+-------------+----------------+-
---------------------+--------------------+----------------------+
```

- !describe — возвращает описание указанной таблицы, содержащее названия всех колонок, типов данных и другие атрибуты:

```
!describe "articles"
```

Вывод на экран:

```
+------------+--------------+-------------+--------------+------------+------------+--------------+----------------+------------
----+-----------------+-----------+----------+------------+----------+
| TABLE_CAT  | TABLE_SCHEM  | TABLE_NAME  | COLUMN_NAME  | DATA_TYPE  | TYPE_NAME  | COLUMN_SIZE  | BUFFER_LENGTH  |
DECIMAL_DIGITS  | NUM_PREC_RADIX  | NULLABLE  | REMARKS  | COLUMN_DEF  | SQL_DATA |
+------------+--------------+-------------+--------------+------------+------------+--------------+----------------+------------
----+-----------------+-----------+----------+------------+----------+
|            |              | articles    | row_id       | 12         | VARCHAR    | null         | null           | null
| null            | 0         |          |            | null     |
|            |              | articles    | author       | 12         | VARCHAR    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | header       | 12         | VARCHAR    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | arch         | 16         | BOOLEAN    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | concepts     | 16         | BOOLEAN    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | tutorials    | 16         | BOOLEAN    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | ref          | 16         | BOOLEAN    | null         | null           | null
| null            | 1         |          |            | null     |
+------------+--------------+-------------+--------------+------------+------------+--------------+----------------+------------
----+-----------------+-----------+----------+------------+----------+
```

Если вывод команды не умещается в окне консоли по ширине, завершите сеанс SQLLine (используя команду `!quit`) и выполните следующую команду для расширения поля вывода:

```
$ stty cols 400
```

Другой способ заключается в переводе SQLLine в режим построчного вывода с помощью следующей команды (для возврата в прежний режим используйте аргумент `horizontal`):

```
!outputformat vertical
```

Кроме использования команд в среде SQLLine, можно проверить создание таблицы с помощью команд `exists` и `describe` в консоли HBase shell. Проверьте наличие таблицы и заданных в ней семейств колонок:

```
exists 'articles'
```

Вывод на экран:

```
Table articles does exist
Took 0.0453 seconds
=> true
```

```
describe 'articles'
```

Вывод на экран:

```
Table articles is ENABLED
articles, {TABLE_ATTRIBUTES => {coprocessor$1 => '|org.apache.phoenix.coprocessor.ScanRegionObserver|805306366|', coprocessor$2 =>
'|org.apache.phoenix.coprocessor.UngroupedAggregateRegionObserver|805306366|', coprocessor$3 => '|org.apache.phoenix.coprocessor.
GroupedAggregateRegionObserver|805306366|', coprocessor$4 => '|org.apache.phoenix.coprocessor.
ServerCachingEndpointImpl|805306366|', coprocessor$5 => '|org.apache.phoenix.hbase.index.Indexer|805306366|index.builder=org.
apache.phoenix.index.PhoenixIndexBuilder,org.apache.hadoop.hbase.index.codec.class=org.apache.phoenix.index.PhoenixIndexCodec'}
COLUMN FAMILIES DESCRIPTION
{NAME => 'basic', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'FAST_DIFF', TTL => 'FOREVER', MIN_VERSIONS => '0',
REPLICATION_SCOPE => '0', BLOOMFILTER => 'NONE', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE =>
'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
{NAME => 'tags', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'FAST_DIFF', TTL => 'FOREVER', MIN_VERSIONS => '0',
REPLICATION_SCOPE => '0', BLOOMFILTER => 'NONE', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE =>
'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
2 row(s)
Took 0.0356 seconds
```

Шаг 3. Добавление данных в таблицу

Для ввода новых данных в созданную таблицу используйте команду UPSERT. Она объединяет две команды SQL: `INSERT` и `UPDATE`. Если строка с указанным ключом уже есть, данные в ней будут обновлены, в противном случае вставится новая строка.

Следующая команда добавляет новые данные в строку с ключом `article1`. Если названия колонок не указаны в круглых скобках сразу после названия таблицы, то приведенные значения появятся в колонках согласно порядку их объявления при создании таблицы. Значение для последней колонки не задано, поэтому она не будет сохранена в HBase.

РЕКОМЕНДАЦИЯ

Значения для колонок типа `CHAR` и `VARCHAR` заключайте в одинарные кавычки `'`, а не двойные.

```
UPSERT INTO "articles" VALUES('article1', 'Test author', 'Test article', true, true, true);
```

Следующая команда добавляет новые данные для строки `article2`. Названия колонок определены явно. HBase не сохранит пропущенные колонки в таблице.

```
UPSERT INTO "articles"("row_id", "basic"."author", "basic"."header", "tags"."ref") VALUES('article2', 'Test author2', 'Test
article2', true);
```

Пример вывода на экран для каждой из команд:

```
1 row affected (0.018 seconds)
```

Шаг 4. Выбор данных из таблицы

Для выбора всех данных из таблицы используйте команду SELECT:

```
SELECT * FROM "articles";
```

Вывод на экран:

```
+-----------+---------------+----------------+-------+-----------+------------+-------+
|  row_id   |    author     |     header     | arch  | concepts  | tutorials  |  ref  |
+-----------+---------------+----------------+-------+-----------+------------+-------+
| article1  | Test author   | Test article   | true  | true      | true       |       |
| article2  | Test author2  | Test article2  |       |           |            | true  |
+-----------+---------------+----------------+-------+-----------+------------+-------+
2 rows selected (0.06 seconds)
```

Можно применять различные конструкции для фильтрации, сортировки и группирования выходных данных: `WHERE`, `GROUP BY`, `ORDER BY` и другие. Можно также использовать подзапросы.

Следующий запрос выбирает данные по значению первичного ключа:

```
SELECT * FROM "articles" WHERE "row_id" = 'article1';
```

Вывод на экран:

```
+-----------+--------------+--------------+-------+-----------+------------+------+
|  row_id   |    author    |    header    | arch  | concepts  | tutorials  | ref  |
+-----------+--------------+--------------+-------+-----------+------------+------+
| article1  | Test author  | Test article | true  | true      | true       |      |
+-----------+--------------+--------------+-------+-----------+------------+------+
1 row selected (0.038 seconds)
```

Можно также проверить содержимое таблицы с помощью команды `scan` в консоли HBase shell:

```
scan 'articles'
```

На экране вы увидите добавленные значения. Кроме колонок, определенных пользователем, Phoenix также добавляет одну служебную колонку для каждой строки — в нашем примере с кодированным ключом `\x00\x00\x00\x00` и кодированным значением `x`. Эта колонка необходима для повышения производительности операций сканирования; не изменяйте ее.

РЕКОМЕНДАЦИЯ

Обратите внимание на то, что Phoenix кодирует данные. Поэтому не рекомендуется использовать HBase API для работы с этими данными, поскольку их необходимо декодировать. Вместо этого используйте Phoenix.

```
ROW                                COLUMN+CELL
 article1                          column=basic:\x00\x00\x00\x00, timestamp=1638285442663, value=x
 article1                          column=basic:\x80\x0B, timestamp=1638285442663, value=Test author
 article1                          column=basic:\x80\x0C, timestamp=1638285442663, value=Test article
 article1                          column=tags:\x80\x0D, timestamp=1638285442663, value=\x01
 article1                          column=tags:\x80\x0E, timestamp=1638285442663, value=\x01
 article1                          column=tags:\x80\x0F, timestamp=1638285442663, value=\x01
 article2                          column=basic:\x00\x00\x00\x00, timestamp=1638285450913, value=x
 article2                          column=basic:\x80\x0B, timestamp=1638285450913, value=Test author2
 article2                          column=basic:\x80\x0C, timestamp=1638285450913, value=Test article2
 article2                          column=tags:\x80\x10, timestamp=1638285450913, value=\x01
2 row(s)
Took 0.0214 seconds
```

Шаг 5. Обновление данных в таблице

Для обновления данных в таблице применяйте ту же команду, что и для вставки новых данных, то есть UPSERT. Следующая команда обновляет значение колонки `header` в строке с ключом `article1`:

```
UPSERT INTO "articles"("row_id", "basic"."header") VALUES('article1', 'Test article. Version 2');
```

Вывод на экран:

```
1 row affected (0.015 seconds)
```

Для просмотра обновленных данных выполните команду `SELECT`:

```
SELECT * FROM "articles";
```

Вывод на экран:

```
+-----------+---------------+--------------------------+-------+-----------+------------+-------+
|  row_id   |    author     |          header          | arch  | concepts  | tutorials  |  ref  |
+-----------+---------------+--------------------------+-------+-----------+------------+-------+
| article1  | Test author   | Test article. Version 2  | true  | true      | true       |       |
| article2  | Test author2  | Test article2            |       |           |            | true  |
+-----------+---------------+--------------------------+-------+-----------+------------+-------+
2 rows selected (0.041 seconds)
```

Проверьте это также в консоли HBase shell с помощью следующей команды:

```
scan 'articles'
```

Результат подтверждает предыдущие выводы:

```
ROW                                COLUMN+CELL
 article1                          column=basic:\x00\x00\x00\x00, timestamp=1638285553403, value=x
 article1                          column=basic:\x80\x0B, timestamp=1638285442663, value=Test author
 article1                          column=basic:\x80\x0C, timestamp=1638285553403, value=Test article. Version 2
 article1                          column=tags:\x80\x0D, timestamp=1638285442663, value=\x01
 article1                          column=tags:\x80\x0E, timestamp=1638285442663, value=\x01
 article1                          column=tags:\x80\x0F, timestamp=1638285442663, value=\x01
 article2                          column=basic:\x00\x00\x00\x00, timestamp=1638285450913, value=x
 article2                          column=basic:\x80\x0B, timestamp=1638285450913, value=Test author2
 article2                          column=basic:\x80\x0C, timestamp=1638285450913, value=Test article2
 article2                          column=tags:\x80\x10, timestamp=1638285450913, value=\x01
2 row(s)
Took 0.0125 seconds
```

Напомним, что при создании таблицы мы потребовали хранение до пяти версий значений. Поэтому HBase сохраняет не только последнее значение, но и предыдущие тоже. Чтобы увидеть это, выполните в консоли HBase shell команду `scan` с аргументом `TIMERANGE`:

```
scan 'articles', {TIMERANGE => [1638285442663, 1638285450914]}
```

Вывод содержит версии значений в указанном диапазоне временных меток:

```
ROW                                COLUMN+CELL
 article1                          column=basic:\x00\x00\x00\x00, timestamp=1638285442663, value=x
 article1                          column=basic:\x80\x0B, timestamp=1638285442663, value=Test author
 article1                          column=basic:\x80\x0C, timestamp=1638285442663, value=Test article
 article1                          column=tags:\x80\x0D, timestamp=1638285442663, value=\x01
 article1                          column=tags:\x80\x0E, timestamp=1638285442663, value=\x01
 article1                          column=tags:\x80\x0F, timestamp=1638285442663, value=\x01
 article2                          column=basic:\x00\x00\x00\x00, timestamp=1638285450913, value=x
 article2                          column=basic:\x80\x0B, timestamp=1638285450913, value=Test author2
 article2                          column=basic:\x80\x0C, timestamp=1638285450913, value=Test article2
 article2                          column=tags:\x80\x10, timestamp=1638285450913, value=\x01
2 row(s)
Took 0.0112 seconds
```

Шаг 6. Удаление данных из таблицы

Для удаления данных из таблицы используйте команду DELETE. Следующая команда удаляет все строки таблицы, в которых значение колонки `header` равно `Test article. Version 2`.

```
DELETE FROM "articles" WHERE "basic"."header" = 'Test article. Version 2';
```

Вывод на экран:

```
1 row affected (0.018 seconds)
```

ВНИМАНИЕ

В отличие от HBase, в Phoenix команда `DELETE` удаляет все версии значений, а не только последнюю, несмотря на указание сохранять множество версий, заданное при создании таблицы.

С помощью команды `SELECT` убедитесь, что в таблице осталась только одна строка:

```
SELECT * FROM "articles";
```

Вывод подтверждает, что первая строка удалена:

```
+-----------+---------------+----------------+-------+-----------+-------------+
|  row_id   |    author     |     header     | arch  | concepts  | tutorials   |
+-----------+---------------+----------------+-------+-----------+-------------+
| article2  | Test author2  | Test article2  |       |           |             |
+-----------+---------------+----------------+-------+-----------+-------------+
1 row selected (0.042 seconds)
```

HBase shell показывает аналогичный результат:

```
scan 'articles'
```

Вывод на экран:

```
ROW                                COLUMN+CELL
 article2                          column=basic:\x00\x00\x00\x00, timestamp=1638283068620, value=x
 article2                          column=basic:\x80\x0B, timestamp=1638283068620, value=Test author2
 article2                          column=basic:\x80\x0C, timestamp=1638283068620, value=Test article2
 article2                          column=tags:\x80\x10, timestamp=1638283068620, value=\x01
1 row(s)
Took 0.0101 seconds
```

Следующая команда удаляет строку с заданным значением главного ключа:

```
DELETE FROM "articles" WHERE "row_id" = 'article2';
```

После этой операции таблица стала пустой:

```
SELECT * FROM "articles";
```

Вывод на экран:

```
+---------+---------+---------+-------+-----------+-----------+------+
| row_id  | author  | header  | arch  | concepts  | tutorials  | ref  |
+---------+---------+---------+-------+-----------+-----------+------+
+---------+---------+---------+-------+-----------+-----------+------+
No rows selected (0.027 seconds)
```

Шаг 7. Изменение структуры таблицы

Для изменения структуры таблицы используйте команду ALTER TABLE. Следующий запрос добавляет семейство колонок `temp` и колонку `review` внутри этого семейства:

```
ALTER TABLE "articles" ADD "temp"."review" VARCHAR VERSIONS=3;
```

Вывод на экран:

```
No rows affected (6.604 seconds)
```

Проверьте обновленную структуру с помощью команды `!describe` в SQLLine:

```
!describe "articles"
```

Вывод на экран содержит информацию о новой колонке `review`:

```
+------------+--------------+-------------+--------------+------------+------------+--------------+----------------+------------
----+-----------------+-----------+----------+------------+----------+
| TABLE_CAT  | TABLE_SCHEM  | TABLE_NAME  | COLUMN_NAME  | DATA_TYPE  | TYPE_NAME  | COLUMN_SIZE  | BUFFER_LENGTH  |
DECIMAL_DIGITS  | NUM_PREC_RADIX  | NULLABLE  | REMARKS  | COLUMN_DEF  | SQL_DATA |
+------------+--------------+-------------+--------------+------------+------------+--------------+----------------+------------
----+-----------------+-----------+----------+------------+----------+
|            |              | articles    | row_id       | 12         | VARCHAR    | null         | null           | null
| null            | 0         |          |            | null     |
|            |              | articles    | author       | 12         | VARCHAR    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | header       | 12         | VARCHAR    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | arch         | 16         | BOOLEAN    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | concepts     | 16         | BOOLEAN    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | tutorials    | 16         | BOOLEAN    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | ref          | 16         | BOOLEAN    | null         | null           | null
| null            | 1         |          |            | null     |
|            |              | articles    | review       | 12         | VARCHAR    | null         | null           | null
| null            | 1         |          |            | null     |
+------------+--------------+-------------+--------------+------------+------------+--------------+----------------+------------
----+-----------------+-----------+----------+------------+----------+
```

Для проверки добавления нового семейства колонок используйте команду `describe` в HBase shell:

```
describe 'articles'
```

Вывод на экран:

```
Table articles is ENABLED
articles, {TABLE_ATTRIBUTES => {coprocessor$1 => '|org.apache.phoenix.coprocessor.ScanRegionObserver|805306366|', coprocesso
r$2 => '|org.apache.phoenix.coprocessor.UngroupedAggregateRegionObserver|805306366|', coprocessor$3 => '|org.apache.phoenix.
coprocessor.GroupedAggregateRegionObserver|805306366|', coprocessor$4 => '|org.apache.phoenix.coprocessor.ServerCachingEndpo
intImpl|805306366|', coprocessor$5 => '|org.apache.phoenix.hbase.index.Indexer|805306366|index.builder=org.apache.phoenix.in
dex.PhoenixIndexBuilder,org.apache.hadoop.hbase.index.codec.class=org.apache.phoenix.index.PhoenixIndexCodec'}
COLUMN FAMILIES DESCRIPTION
{NAME => 'basic', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'FAST_DIFF', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICAT
ION_SCOPE => '0', BLOOMFILTER => 'NONE', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'fa
lse', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
{NAME => 'tags', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS => '
FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'FAST_DIFF', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATI
ON_SCOPE => '0', BLOOMFILTER => 'NONE', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'fal
se', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
{NAME => 'temp', VERSIONS => '3', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS => '
FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'FAST_DIFF', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATI
ON_SCOPE => '0', BLOOMFILTER => 'NONE', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'fal
se', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
3 row(s)
Took 0.0222 seconds
```

Шаг 8. Удаление таблицы

Для окончательного удаления таблицы используйте команду DROP TABLE:

```
DROP TABLE "articles";
```

Вывод на экран:

```
No rows affected (1.246 seconds)
```

Теперь команда `!tables` покажет, что таблица `articles` более не существует:

```
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------------+--------------+--------------+-----------------+------------+-------------+----------------+-
---------------------+--------------------+----------------------+
| TABLE_CAT  | TABLE_SCHEM  | TABLE_NAME  |  TABLE_TYPE   | REMARKS  | TYPE_NAME  | SELF_REFERENCING_COL_NAME  | REF_GENERATION
| INDEX_STATE  | IMMUTABLE_ROWS  | SALT_BUCKETS  | MULTI_TENANT  | VIEW_STATEMENT  | VIEW_TYPE  | INDEX_TYPE  | TRANSACTIONAL  |
IS_NAMESPACE_MAPPED  | GUIDE_POSTS_WIDTH  | TRANSACTION_PROVIDER  |
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------------+--------------+--------------+-----------------+------------+-------------+----------------+-
---------------------+--------------------+----------------------+
|            | SYSTEM       | CATALOG     | SYSTEM TABLE  |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
|            | SYSTEM       | FUNCTION    | SYSTEM TABLE  |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
|            | SYSTEM       | LOG         | SYSTEM TABLE  |          |            |                            |
|              | true            | 32            | false         |                 |            |             | false          |
false                | null               |                      |
|            | SYSTEM       | SEQUENCE    | SYSTEM TABLE  |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
|            | SYSTEM       | STATS       | SYSTEM TABLE  |          |            |                            |
|              | false           | null          | false         |                 |            |             | false          |
false                | null               |                      |
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------------+--------------+--------------+-----------------+------------+-------------+----------------+-
---------------------+--------------------+----------------------+
```

Оператор JOIN в Phoenix

Владимир Адаменков

Содержание

Назад к содержанию

Обзор
Типы JOIN
  - INNER JOIN
  - LEFT/RIGHT OUTER JOIN
  - FULL OUTER JOIN
  - SELF JOIN
  - CROSS JOIN
  - GROUP JOIN
Исключение части INNER из OUTER JOIN
Использование индексов
Hints
  - Алгоритмы hash и sort-merge
  - Оптимизация star join

Обзор

Phoenix полностью поддерживает стандартный SQL-синтаксис оператора `JOIN`. Он позволяет совмещать записи из разных таблиц при наличии общих значений.

В этой статье для примеров используются три таблицы со следующими столбцами.

Таблицы для примеров

Таблица	Столбец	Описание
states	name	Название штата
	abbr	Сокращение названия штата
	capitol	Столица штата
	largest	Крупнейший город штата
enterprises	name	Название предприятия
	country	Название страны, в которой расположено предприятие
	state	Сокращение названия штата (в США) или название региона, в котором расположено предприятие
	town	Название города, в котором расположено предприятие
people	name	Имя человека
	age	Возраст человека
	town	Город проживания человека
	state	Сокращение названия штата (в США) или название региона проживания человека
	country	Страна проживания человека

Чтобы иметь возможность выполнять примеры, приведенные в этой статье, нужно создать эти таблицы в вашем окружении Phoenix. Выполните следующее:

1. Загрузите CSV-файлы с тестовыми данными: people.csv, enterprises.csv и states.csv.
2. Поместите загруженные файлы в файловую систему хоста, на котором установлен компонент ***Phoenix Query Server***, в любой каталог на ваше усмотрение.
3. Подключитесь к этому хосту и загрузите командную оболочку Phoenix, выполнив следующую команду:

```
$ /usr/lib/phoenix/bin/sqlline.py
```

4. Создайте пустые таблицы, выполнив следующие команды:

```sql
CREATE TABLE enterprises (
name VARCHAR(50) NOT NULL,
country VARCHAR(50),
state VARCHAR(50),
town VARCHAR(50)
CONSTRAINT pk PRIMARY KEY (name));
```

```sql
CREATE TABLE people (
name VARCHAR(50) NOT NULL,
age SMALLINT,
town VARCHAR(50),
state VARCHAR(50),
country VARCHAR(50)
CONSTRAINT pk PRIMARY KEY (name));
```

```sql
CREATE TABLE states (
name VARCHAR(50) NOT NULL,
abbr VARCHAR(50),
capitol VARCHAR(50),
largest VARCHAR(50)
CONSTRAINT pk PRIMARY KEY (name));
```

5. Выйдите из командной оболочки, нажав `Ctrl+D`, и загрузите данные из CSV-файлов в таблицы, выполнив следующие команды:

```
$ /usr/lib/phoenix/bin/psql.py /<csv_dir>/people.csv
$ /usr/lib/phoenix/bin/psql.py /<csv_dir>/enterprises.csv
$ /usr/lib/phoenix/bin/psql.py /<csv_dir>/states.csv
```

где `<csv_dir>` — путь к каталогу, в котором расположены загруженные CSV-файлы.

Типы JOIN

INNER JOIN

`INNER JOIN` берет каждую строку в первой таблице и ищет строку во второй таблице, для которой выполняется заданное условие. Если такая строка найдена, `INNER JOIN` составляет строку из значений столбцов, заданных в запросе. Эта строка добавляется к результату, и `INNER JOIN` продолжает работу.

Этот режим объединения используется по умолчанию, поэтому модификатор `INNER` в тексте запроса можно не писать.

Например, чтобы вывести на экран все предприятия из США вместе с городами их размещения и столицами соответствующих штатов, нужно выполнить следующий запрос:

```sql
SELECT e.name, e.town, s.capitol
FROM states AS s
INNER JOIN enterprises AS e
ON s.abbr=e.state;
```

Результат:

```
+------------------+---------------+-------------+
|      E.NAME      |    E.TOWN     |  S.CAPITOL  |
+------------------+---------------+-------------+
| Rockstar         | New York City | Albany      |
| Bethesda         | Rockville     | Annapolis   |
| DigitalAnvil     | Austin        | Austin      |
| Shaw Group       | Baton Rouge   | Baton Rouge |
| Microsoft        | Redmond       | Olympia     |
| Avnet            | Phoenix       | Phoenix     |
| Apple            | Cupertino     | Sacramento  |
| Obsidian         | Irvine        | Sacramento  |
| Sierra           | Los Angeles   | Sacramento  |
| Intel            | Sacramento    | Sacramento  |
| Northrop Grumman | Albuquerque   | Santa Fe    |
+------------------+---------------+-------------+
```

LEFT/RIGHT OUTER JOIN

`LEFT OUTER JOIN` работает как `INNER JOIN`, но также добавляет данные из первой таблицы в любом случае, независимо от выполнения заданного условия.

Например, чтобы вывести на экран все предприятия вместе с городами их размещения, а также столицы штатов для тех из них, которые располагаются в США, нужно выполнить следующий запрос:

```sql
SELECT e.name, e.town, s.capitol
FROM enterprises AS e
LEFT OUTER JOIN states AS s
ON e.state=s.abbr;
```

Результат:

```
+------------------+---------------+-------------+
|      E.NAME      |    E.TOWN     |  S.CAPITOL  |
+------------------+---------------+-------------+
| Northrop Grumman | Albuquerque   | Santa Fe    |
| DigitalAnvil     | Austin        | Austin      |
| Shaw Group       | Baton Rouge   | Baton Rouge |
| Apple            | Cupertino     | Sacramento  |
| Obsidian         | Irvine        | Sacramento  |
| Eidos            | London        |             |
| Sierra           | Los Angeles   | Sacramento  |
| Alluxi           | Monterrey     |             |
| Rockstar         | New York City | Albany      |
| Empresa          | Nogales       |             |
| Avnet            | Phoenix       | Phoenix     |
| Microsoft        | Redmond       | Olympia     |
| Bethesda         | Rockville     | Annapolis   |
| Intel            | Sacramento    | Sacramento  |
| Ubisoft          | Saint-Mande   |             |
+------------------+---------------+-------------+
```

`RIGHT OUTER JOIN` делает то же самое, но в отношении второй таблицы, а не первой.

Например, чтобы вывести на экран все штаты США вместе с их столицами, а также предприятия вместе с городами их размещения, которые могут находиться в некоторых из этих штатов, нужно выполнить следующий запрос:

```sql
SELECT s.name, s.capitol, e.name, e.town
FROM enterprises AS e
RIGHT OUTER JOIN states AS s
ON e.state=s.abbr
ORDER BY s.name
LIMIT 20;
```

Результат:

```
+-------------+--------------+----------+-------------+
|   S.NAME    |  S.CAPITOL   |  E.NAME  |   E.TOWN    |
+-------------+--------------+----------+-------------+
| Alabama     | Montgomery   |          |             |
| Alaska      | Juneau       |          |             |
| Arizona     | Phoenix      | Avnet    | Phoenix     |
| Arkansas    | Little Rock  |          |             |
| California  | Sacramento   | Apple    | Cupertino   |
| California  | Sacramento   | Obsidian | Irvine      |
| California  | Sacramento   | Sierra   | Los Angeles |
| California  | Sacramento   | Intel    | Sacramento  |
| Colorado    | Denver       |          |             |
| Connecticut | Hartford     |          |             |
| Delaware    | Dover        |          |             |
| Florida     | Tallahassee  |          |             |
| Georgia     | Atlanta      |          |             |
| Hawaii      | Honolulu     |          |             |
| Idaho       | Boise        |          |             |
| Illinois    | Springfield  |          |             |
| Indiana     | Indianapolis |          |             |
| Iowa        | Des Moines   |          |             |
| Kansas      | Topeka       |          |             |
| Kentucky    | Frankfort    |          |             |
+-------------+--------------+----------+-------------+
```

Модификатор `OUTER` подразумевается при использовании модификаторов `LEFT` и `RIGHT`, поэтому его можно опустить.

FULL OUTER JOIN

`FULL OUTER JOIN` объединяет `LEFT JOIN` и `RIGHT JOIN`: он соединяет строки, удовлетворяющие заданному условию, со всеми остальными строками из обеих таблиц.

Пример:

```sql
SELECT e.name, e.town, s.capitol
FROM enterprises AS e
FULL OUTER JOIN states AS s
ON e.state=s.abbr
LIMIT 20;
```

Результат:

```
+------------------+---------------+----------------+
|      E.NAME      |    E.TOWN     |   S.CAPITOL    |
+------------------+---------------+----------------+
|                  |               | Juneau         |
|                  |               | Montgomery     |
|                  |               | Little Rock    |
| Avnet            | Phoenix       | Phoenix        |
| Apple            | Cupertino     | Sacramento     |
| Obsidian         | Irvine        | Sacramento     |
| Intel            | Sacramento    | Sacramento     |
| Sierra           | Los Angeles   | Sacramento     |
|                  |               | Denver         |
|                  |               | Hartford       |
|                  |               | Dover          |
| Eidos            | London        |                |
|                  |               | Tallahassee    |
|                  |               | Atlanta        |
|                  |               | Honolulu       |
|                  |               | Des Moines     |
|                  |               | Boise          |
|                  |               | Springfield    |
|                  |               | Indianapolis   |
| Ubisoft          | Saint-Mande   |                |
+------------------+---------------+----------------+
```

Модификатор `OUTER` подразумевается при использовании модификатора `FULL`, поэтому его можно опустить.

SELF JOIN

Таблицу можно объединить саму с собой (что может быть использовано для фильтрации). Например, чтобы вывести на экран столицы штатов, которые также являются крупнейшими городами, нужно выполнить следующий запрос:

```sql
SELECT s.name, s.capitol
FROM states AS s
JOIN states AS ss
ON s.capitol=ss.largest;
```

Результат:

```
+---------------+----------------+
|    S.NAME     |   S.CAPITOL    |
+---------------+----------------+
| Arizona       | Phoenix        |
| Arkansas      | Little Rock    |
| Colorado      | Denver         |
| Georgia       | Atlanta        |
| Hawaii        | Honolulu       |
| Idaho         | Boise          |
| Indiana       | Indianapolis   |
| Iowa          | Des Moines     |
| Massachusetts | Boston         |
| Mississippi   | Jackson        |
| Ohio          | Columbus       |
| Oklahoma      | Oklahoma City  |
| Rhode Island  | Providence     |
| Tennessee     | Nashville      |
| Utah          | Salt Lake City |
| West Virginia | Charleston     |
| West Virginia | Charleston     |
| Wyoming       | Cheyenne       |
+---------------+----------------+
```

CROSS JOIN

В большинстве случаев в Phoenix не применяется `CROSS JOIN`, так как таблицы обычно очень большие. По этой причине Phoenix игнорирует модификатор `CROSS`. Тем не менее `CROSS JOIN` возможен в Phoenix и использует следующий синтаксис:

```sql
SELECT <col_1>, <col_2>, ... <col_n>
FROM t1, t2;
```

GROUP JOIN

Phoenix поддерживает составной синтаксис оператора `JOIN`. Результат объединения двух таблиц можно сгруппировать с другой таблицей при помощи скобок. Также вместо любой таблицы в выражении `JOIN` можно использовать подзапрос (производную таблицу), который в свою очередь может содержать собственное выражение `JOIN`.

Например, чтобы вывести на экран список людей, проживающих в столице того или иного штата, в которой также размещается предприятие, нужно выполнить следующий запрос:

```sql
SELECT p.name, e.name, s.capitol
FROM people AS p
JOIN
    (enterprises AS e
    JOIN states AS s
    ON e.town=s.capitol)
ON p.town=e.town
LIMIT 20;
```

Результат:

```
+------------------+--------------+-------------+
|      P.NAME      |    E.NAME    |  S.CAPITOL  |
+------------------+--------------+-------------+
| Abbott Howard    | Shaw Group   | Baton Rouge |
| Alvarado Melvin  | DigitalAnvil | Austin      |
| Baker Daniel     | Shaw Group   | Baton Rouge |
| Barnes Mollie    | Shaw Group   | Baton Rouge |
| Becker Samuel    | Shaw Group   | Baton Rouge |
| Bush Cameron     | Shaw Group   | Baton Rouge |
| Carpenter Carrie | Avnet        | Phoenix     |
| Clarke Joe       | Avnet        | Phoenix     |
| Cox Dennis       | Intel        | Sacramento  |
| Cox Shawn        | Avnet        | Phoenix     |
| Daniel Ada       | DigitalAnvil | Austin      |
| Day Hattie       | Intel        | Sacramento  |
| Erickson Mayme   | Intel        | Sacramento  |
| Evans Sophie     | Avnet        | Phoenix     |
| Farmer Dean      | Shaw Group   | Baton Rouge |
| Fisher Gordon    | Intel        | Sacramento  |
| Fletcher Julia   | Avnet        | Phoenix     |
| Garner Juan      | Shaw Group   | Baton Rouge |
| Gill Josephine   | Shaw Group   | Baton Rouge |
| Gonzales Edna    | Shaw Group   | Baton Rouge |
+------------------+--------------+-------------+
```

Исключение части INNER из OUTER JOIN

Иногда бывает нужно исключить строки, удовлетворяющие заданному условию из результата `JOIN`. Для этого в запрос нужно добавить выражение `WHERE`.

Например, чтобы вывести на экран все предприятия не из США, нужно выполнить следующий запрос:

```sql
SELECT e.name, e.country, e.state, e.town
FROM enterprises AS e
LEFT JOIN states AS s
ON e.state=s.abbr
WHERE s.abbr IS NULL;
```

Результат:

```
+---------+-----------+---------------+-------------+
| E.NAME  | E.COUNTRY |    E.STATE    |   E.TOWN    |
+---------+-----------+---------------+-------------+
| Alluxi  | Mexico    | Nuevo Leon    | Monterrey   |
| Eidos   | UK        | England       | London      |
| Empresa | Mexico    | Sonora        | Nogales     |
| Ubisoft | France    | Ile-de-France | Saint-Mande |
+---------+-----------+---------------+-------------+
```

Использование индексов

Можно создать вторичные индексы для таблиц, которые будут автоматически использоваться для запросов с оператором `JOIN`. Пример:

```sql
CREATE INDEX ipeople ON people (town) INCLUDE (state);
CREATE INDEX istates ON states (capitol) INCLUDE (name, abbr);
CREATE INDEX ienterprises ON enterprises (town) INCLUDE (name, state);
```

Эти индексы будут иметь приоритет при выполнении определенных запросов. Пример:

```sql
EXPLAIN SELECT p.name, e.name, s.capitol
FROM people AS p
JOIN
    (enterprises AS e
    JOIN states AS s
    ON e.town=s.capitol)
ON p.town=e.town;
```

Результат:

```
+----------------------------------------------------------------------------------+----------------+---------------+------------
-+
|                                       PLAN                                       | EST_BYTES_READ | EST_ROWS_READ | EST_INFO_TS
|
+----------------------------------------------------------------------------------+----------------+---------------+------------
-+
| CLIENT 1-CHUNK PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER IPEOPLE                 | null           | null          | null
|
|     SERVER FILTER BY FIRST KEY ONLY                                              | null           | null          | null
|
|     PARALLEL INNER-JOIN TABLE 0                                                  | null           | null          | null
|
|         CLIENT 1-CHUNK PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER IENTERPRISES    | null           | null          | null
|
|             SERVER FILTER BY FIRST KEY ONLY                                      | null           | null          | null
|
|             PARALLEL INNER-JOIN TABLE 0                                          | null           | null          | null
|
|                 CLIENT 1-CHUNK PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER ISTATES | null           | null          | null
|
|                     SERVER FILTER BY FIRST KEY ONLY                              | null           | null          | null
|
|     DYNAMIC SERVER FILTER BY "P.0:TOWN" IN ("E.LOCATION:TOWN")                  | null           | null          | null
|
+----------------------------------------------------------------------------------+----------------+---------------+------------
-+
```

Как видно, для данного запроса Phoenix использует все три ранее созданных индекса вместо оригинальных таблиц.

Hints

Phoenix использует **указания (hints)** для переключения между различными способами работы `JOIN`.

Алгоритмы hash и sort-merge

Phoenix использует два типа алгоритмов `JOIN`: **hash** и **sort-merge**. Hash — самый быстрый алгоритм, поэтому он используется по умолчанию. Однако если общий размер таблиц превосходит размер доступной памяти, этот алгоритм не может быть использован. В этом случае следует использовать алгоритм sort-merge с помощью указания `/*+ USE_SORT_MERGE_JOIN */` в запросе:

```sql
SELECT /*+ USE_SORT_MERGE_JOIN */ <values>
FROM <table>
JOIN <clause_condition>
```

Оптимизация star join

Phoenix использует оптимизацию star join для запросов, содержащих несколько выражений `JOIN`. Например, запрос имеет следующую структуру:

```sql
SELECT p.name, s.abbr, e.name
FROM people AS p
JOIN states AS s
ON p.state=s.abbr
JOIN enterprises AS e
ON p.state=e.state;
```

При обработке такого запроса Phoenix загрузит все указанные таблицы в оперативную память и будет выполнять оба выражения `JOIN` одновременно. Это можно увидеть в плане запроса, который можно получить с помощью оператора `EXPLAIN`:

```sql
EXPLAIN SELECT p.name, s.abbr, e.name
FROM people AS p
JOIN states AS s
ON p.state=s.abbr
JOIN enterprises AS e
ON p.state=e.state;
```

Результат:

```
+----------------------------------------------------------------------------------------------------------+----------------+----
-----------+--------+
|                                                   PLAN                                                   | EST_BYTES_READ |
EST_ROWS_READ | EST_IN |
+----------------------------------------------------------------------------------------------------------+----------------+----
-----------+--------+
| CLIENT 1-CHUNK PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER PEOPLE                                          | null           |
null          | null   |
|     PARALLEL INNER-JOIN TABLE 0                                                                          | null           |
null          | null   |
|         CLIENT 1-CHUNK PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER STATES                                  | null           |
null          | null   |
|     PARALLEL INNER-JOIN TABLE 1                                                                          | null           |
null          | null   |
|         CLIENT 1-CHUNK 976934 ROWS 314572800 BYTES PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER ENTERPRISES | null           |
null          | null   |
+----------------------------------------------------------------------------------------------------------+----------------+----
-----------+--------+
```

Иногда таблицы могут быть слишком велики для того, чтобы поместиться в оперативной памяти все вместе. В таких случаях нужно отключить оптимизацию star join с помощью указания `/+ NO_STAR_JOIN/` в запросе:

```sql
EXPLAIN SELECT /*+ NO_STAR_JOIN*/ p.name, s.abbr, e.name
FROM people AS p
JOIN states AS s
ON p.state=s.abbr
JOIN enterprises AS e
ON p.state=e.state;
```

Результат:

```
+--------------------------------------------------------------------------------------------------+----------------+------------
---+-------------+
|                                               PLAN                                               | EST_BYTES_READ |
EST_ROWS_READ | EST_INFO_TS |
+--------------------------------------------------------------------------------------------------+----------------+------------
---+-------------+
| CLIENT 1-CHUNK 976934 ROWS 314572800 BYTES PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER ENTERPRISES | null           | null
| null        |
|     PARALLEL INNER-JOIN TABLE 0                                                                  | null           | null
| null        |
|         CLIENT 1-CHUNK PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER PEOPLE                          | null           | null
| null        |
|             PARALLEL INNER-JOIN TABLE 0                                                          | null           | null
| null        |
|                 CLIENT 1-CHUNK PARALLEL 1-WAY ROUND ROBIN FULL SCAN OVER STATES                  | null           | null
| null        |
+--------------------------------------------------------------------------------------------------+----------------+------------
---+-------------+
```

Как видно, в этом случае операции `JOIN` выполняются по очереди.

Подзапросы в Phoenix

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
- IN
- EXISTS
- Операторы сравнения
  - Простые
  - Модифицированные
- Коррелирующие подзапросы
- Ветвление и вложенность
  - Ветвление AND и OR
  - Вложенность
- Производные таблицы
- Ограничения

Обзор

Phoenix поддерживает подзапросы в выражениях `WHERE` и `FROM`. При составлении подзапросов можно использовать логические операторы и операторы сравнения, с модификаторами и без.

Чтобы иметь возможность выполнять примеры, приведенные в этой статье, нужно создать эти таблицы в вашем окружении Phoenix, как указано в статье Оператор JOIN в Phoenix.

IN

Оператор `IN` позволяет выполнять поиск в результатах выполнения подзапроса. Подзапрос в скобках возвращает таблицу (возможно, пустую), и оператор `IN` воспринимает эту таблицу как множество значений, среди которых он будет искать совпадения по заданному условию.

Например, чтобы вывести на экран список людей, проживающих в штатах США, в которых расположены предприятия, нужно выполнить следующий запрос:

```sql
SELECT p.name, p.state, p.town
FROM people AS p
WHERE p.state IN
    (SELECT e.state
    FROM enterprises AS e
    WHERE e.country='USA')
LIMIT 20;
```

Результат:

```
+-------------------+---------+---------------+
|      P.NAME       | P.STATE |    P.TOWN     |
+-------------------+---------+---------------+
| Abbott Delia      | TX      | Dallas        |
| Abbott Howard     | LA      | Baton Rouge   |
| Adams Clyde       | LA      | Lafayette     |
| Aguilar Terry     | CA      | Fresno        |
| Alexander Derrick | TX      | Dallas        |
| Alexander Gregory | LA      | Lafayette     |
| Alexander Leon    | AZ      | Kingman       |
| Alvarado Melvin   | TX      | Austin        |
| Alvarado Timothy  | CA      | San Francisco |
| Alvarez Bruce     | CA      | Modesto       |
| Andrews Lucy      | AZ      | Kingman       |
| Armstrong Marie   | NM      | Santa Fe      |
| Atkins Gene       | TX      | Dallas        |
| Austin Eugene     | LA      | Lafayette     |
| Austin Travis     | TX      | El Paso       |
| Bailey Cameron    | TX      | Lubbock       |
| Baker Daniel      | LA      | Baton Rouge   |
| Ball Nelle        | NM      | Albuquerque   |
| Barber Minerva    | TX      | Beaumont      |
| Barnes Mollie     | LA      | Baton Rouge   |
+-------------------+---------+---------------+
```

Также можно выполнять поиск по нескольким столбцам. Количество столбцов в выражении `WHERE` должно совпадать с количеством столбцов в выражении `SELECT` в подзапросе. Если количество столбцов больше одного, то они должны быть заключены в скобки в выражении `WHERE`.

Например, чтобы вывести на экран предприятия, расположенные в столицах штатов, которые также являются крупнейшими городами, нужно выполнить следующий запрос:

```sql
SELECT e.name, e.state, e.town
FROM enterprises AS e
WHERE (e.state, e.town) IN
    (SELECT s.abbr, s.capitol
    FROM states AS s
    WHERE s.capitol=s.largest);
```

Результат:

```
+--------+---------+---------+
| E.NAME | E.STATE | E.TOWN  |
+--------+---------+---------+
| Avnet  | AZ      | Phoenix |
+--------+---------+---------+
```

Ключевое слово `NOT` инвертирует выражение `IN`. Оператор `NOT IN` выполняет поиск среди значений, не принадлежащих результату подзапроса.

Например, чтобы вывести на экран список людей, живущих в штатах, где нет 44-летних людей, нужно выполнить следующий запрос:

```sql
SELECT p.name, p.state, p.age
FROM people AS p
WHERE p.state NOT IN
    (SELECT pp.state
    FROM people AS pp
    WHERE pp.age=44)
LIMIT 20;
```

Результат:

```
+-------------------+---------+-------+
|      P.NAME       | P.STATE | P.AGE |
+-------------------+---------+-------+
| Alexander Leon    | AZ      | 42    |
| Andrews Lucy      | AZ      | 33    |
| Becker Marie      | AZ      | 19    |
| Bowman Randall    | AZ      | 46    |
| Bryant Marcus     | OR      | 30    |
| Buchanan Todd     | OR      | 25    |
| Burton Luella     | AZ      | 62    |
| Byrd Nannie       | OR      | 59    |
| Carpenter Carrie  | AZ      | 24    |
| Chambers Hettie   | AZ      | 52    |
| Clarke Joe        | AZ      | 26    |
| Cox Shawn         | AZ      | 25    |
| Cunningham Sallie | AZ      | 56    |
| Duncan Ollie      | AZ      | 45    |
| Elliott Myrtie    | OR      | 59    |
| Evans Sophie      | AZ      | 58    |
| Ferguson Josie    | AZ      | 28    |
| Fletcher Julia    | AZ      | 38    |
| Fowler Hettie     | OR      | 36    |
| Fowler May        | OR      | 64    |
+-------------------+---------+-------+
```

EXISTS

Оператор `EXISTS` является логическим. Он возвращает значение `true`, если результат подзапроса непуст, и значение `false` в противном случае. В процессе обработки запроса подзапрос в скобках выполняется для каждой итерации внешнего запроса. Также, чтобы подзапрос в выражении `EXISTS` имел смысл, нужно, чтобы он коррелировал с внешним запросом, то есть имел связь с его условиями. Если корреляции между запросами нет, то выражение `EXISTS` всегда будет иметь одно и то же значение независимо от условий внешнего запроса.

Например, чтобы вывести на экран штаты США, в которых живут люди в возрасте 65 лет, нужно выполнить следующий запрос:

```sql
SELECT s.name
FROM states AS s
WHERE EXISTS
    (SELECT p.state
    FROM people AS p
    WHERE p.age=65 AND p.state=s.abbr);
```

Как видно, в этом примере подзапрос содержит поиск по таблице `states`, как и внешний запрос.

Результат:

```
+------------+
|   S.NAME   |
+------------+
| Arizona    |
| California |
| Louisiana  |
| Nevada     |
| Oregon     |
| Texas      |
+------------+
```

Ключевое слово `NOT` инвертирует логический оператор. Оператор `NOT EXISTS` возвращает значение `true`, если результат подзапроса пуст, и значение `false` в противном случае.

Например, чтобы вывести на экран все штаты за пределами США из таблицы `people`, нужно выполнить следующий запрос:

```sql
SELECT p.state
FROM people AS p
WHERE NOT EXISTS
    (SELECT s.abbr
    FROM states AS s
    WHERE s.abbr=p.state)
GROUP BY p.state;
```

Результат:

```
+-----------------+
|     P.STATE     |
+-----------------+
| California Baja |
| Chihuahua       |
| Quintana Roo    |
| Sonora          |
| Yucatan         |
+-----------------+
```

Операторы сравнения

В выражении `WHERE` можно использовать операторы сравнения, при этом подзапрос является правым операндом. Phoenix использует следующие операторы сравнения:

- `=` — равно;
- `!=` или `<>` — не равно;
- `>` — больше;
- `<` — меньше;
- `>=` или `!<` — больше или равно;
- `<=` или `!>` — меньше или равно.

Простые

Простые операторы сравнения используются так же, как и в обычных операциях сравнения, без модификаторов.

Например, чтобы вывести на экран распределение возрастов людей, старших среднего возраста, нужно выполнить следующий запрос:

```sql
SELECT p.age, count(p.age)
FROM people AS p
WHERE p.age >
    (SELECT avg(p.age)
    FROM people AS p)
GROUP BY p.age;
```

Результат:

```
+-----+------------+
| AGE | COUNT(AGE) |
+-----+------------+
| 41  | 19         |
| 42  | 19         |
| 43  | 22         |
| 44  | 16         |
| 45  | 19         |
| 46  | 17         |
| 47  | 23         |
| 48  | 23         |
| 49  | 14         |
| 50  | 12         |
| 51  | 17         |
| 52  | 26         |
| 53  | 21         |
| 54  | 21         |
| 55  | 10         |
| 56  | 26         |
| 57  | 20         |
| 58  | 26         |
| 59  | 22         |
| 60  | 22         |
| 61  | 23         |
| 62  | 21         |
| 63  | 18         |
| 64  | 21         |
| 65  | 17         |
+-----+------------+
```

Модифицированные

Модифицированные операторы сравнения являются логическими и используют один из следующих модификаторов:

- `ANY` или `SOME` — оператор возвращает значение `true`, если хотя бы один результат подзапроса соответствует условию сравнения, и значение `false` в противном случае;
- `ALL` — оператор возвращает значение `true`, если все результаты подзапроса соответствуют условию сравнения, и значение `false` в противном случае.

Подзапрос должен возвращать значения только из одного столбца.

Например, чтобы вывести на экран штаты США, где расположены предприятия, а также их столицы, нужно выполнить следующий запрос:

```sql
SELECT s.name, s.capitol
FROM states AS s
WHERE s.abbr = ANY
    (SELECT e.state
    FROM enterprises AS e
    JOIN
    states AS s
    ON s.abbr=e.state);
```

Результат:

```
+------------+-------------+
|    NAME    |   CAPITOL   |
+------------+-------------+
| Arizona    | Phoenix     |
| California | Sacramento  |
| Louisiana  | Baton Rouge |
| Maryland   | Annapolis   |
| New Mexico | Santa Fe    |
| New York   | Albany      |
| Texas      | Austin      |
| Washington | Olympia     |
+------------+-------------+
```

Чтобы вывести на экран список людей, чей возраст не встречается среди жителей Невады, а также их штат и возраст, нужно выполнить следующий запрос:

```sql
SELECT p.name, p.state, p.age
FROM people AS p
WHERE p.age <> ALL
    (SELECT p.age
    FROM people AS p
    WHERE p.state='NV')
LIMIT 20;
```

Результат:

```
+--------------------+-----------------+-----+
|        NAME        |      STATE      | AGE |
+--------------------+-----------------+-----+
| Abbott Jack        | Chihuahua       | 29  |
| Adams Clyde        | LA              | 29  |
| Alvarado Dominic   | California Baja | 63  |
| Alvarez Jacob      | Yucatan         | 49  |
| Anderson Lucile    | Sonora          | 33  |
| Andrews Lucy       | AZ              | 33  |
| Andrews Noah       | California Baja | 63  |
| Armstrong Isabella | Sonora          | 49  |
| Atkins Gene        | TX              | 37  |
| Bailey Richard     | Yucatan         | 39  |
| Benson Hallie      | California Baja | 37  |
| Bowers Norman      | Chihuahua       | 29  |
| Boyd Travis        | Yucatan         | 49  |
| Brock Carlos       | Quintana Roo    | 49  |
| Burgess Nell       | TX              | 33  |
| Byrd Nell          | Quintana Roo    | 39  |
| Castillo Edna      | Quintana Roo    | 63  |
| Chapman Lilly      | Sonora          | 63  |
| Christensen Gavin  | Quintana Roo    | 63  |
| Clark Joshua       | Sonora          | 63  |
+--------------------+-----------------+-----+
```

Коррелирующие подзапросы

Подзапрос называется **коррелирующим** или **синхронизированным**, если он обращается к тем же таблицам, что и внешний запрос. Коррелирующий подзапрос выполняется каждый раз, когда внешний запрос обрабатывает очередную строку согласно собственным условиям. Это отличает коррелирующий запрос от независимого, который не связан с внешним запросом и выполняется лишь единожды.

Например, чтобы вывести на экран штаты США, в крупнейших городах которых расположены предприятия, а также их столицы, нужно выполнить следующий запрос:

```sql
SELECT s.name, s.capitol
FROM states AS s
WHERE s.abbr IN
    (SELECT e.state
    FROM enterprises AS e
    WHERE e.town=s.largest);
```

Результат:

```
+------------+------------+
|   S.NAME   | S.CAPITOL  |
+------------+------------+
| Arizona    | Phoenix    |
| California | Sacramento |
| New Mexico | Santa Fe   |
| New York   | Albany     |
+------------+------------+
```

Ветвление и вложенность

Подзапросы можно объединять с помощью операторов `AND` и `OR`. Также можно вкладывать подзапросы один в другой сколько угодно раз.

Ветвление AND и OR

Логический оператор `AND` означает, что оба операнда в выражении должны иметь значение `true`, чтобы все выражение имело значение `true`. Логический оператор `OR` означает, что любой из операндов в выражении может иметь значение `true`, чтобы все выражение имело значение `true`.

Например, чтобы вывести на экран жителей США в возрасте 65 лет, а также всех людей в возрасте 18 лет, нужно выполнить следующий запрос:

```sql
SELECT p.name, p.age, p.state
FROM people AS p
WHERE p.state IN
    (SELECT s.abbr
    FROM states AS s)
AND p.age>64
OR p.age IN
    (SELECT p.age
    FROM people AS p
    WHERE p.age=18);
```

Результат:

```
+--------------------+-------+-----------------+
|       P.NAME       | P.AGE |     P.STATE     |
+--------------------+-------+-----------------+
| Aguilar Terry      | 65    | CA              |
| Arnold Bettie      | 18    | NV              |
| Bailey Cameron     | 18    | TX              |
| Banks Catherine    | 18    | Sonora          |
| Campbell Todd      | 18    | NV              |
| Casey Louise       | 18    | California Baja |
| Davis Alan         | 18    | CA              |
| Farmer Dean        | 65    | LA              |
| Fitzgerald Timothy | 18    | Chihuahua       |
| Griffin Alexander  | 65    | NV              |
| Harper Alvin       | 65    | AZ              |
| Hill Katharine     | 18    | Quintana Roo    |
| Hodges Abbie       | 18    | Yucatan         |
| James Loretta      | 18    | TX              |
| Lopez Katharine    | 65    | AZ              |
| Love Dylan         | 65    | TX              |
| McCarthy Eunice    | 65    | LA              |
| Miller Francis     | 18    | Sonora          |
| Munoz Ricardo      | 18    | CA              |
| Myers Ricky        | 18    | LA              |
| Norton Bertha      | 18    | TX              |
| Phelps Rena        | 18    | Quintana Roo    |
| Riley Mollie       | 65    | OR              |
| Stokes Mittie      | 18    | OR              |
| Strickland Helena  | 65    | TX              |
| Thompson Warren    | 18    | AZ              |
| Torres Elsie       | 18    | LA              |
| Vaughn Belle       | 18    | CA              |
+--------------------+-------+-----------------+
```

Вложенность

Можно использовать подзапросы внутри подзапросов. Это называется **вложенность**.

Например, чтобы вывести на экран предприятия, расположенные в штатах, начинающихся на буквы `A`, `B` и `C`, для людей из таблицы `people`, нужно выполнить следующий запрос:

```sql
SELECT e.name
FROM enterprises AS e
WHERE e.state IN
    (SELECT p.state
    FROM people AS p
    WHERE p.state IN
        (SELECT s.abbr
        FROM states AS s
        WHERE s.name < 'D'));
```

Результат:

```
+----------+
|  E.NAME  |
+----------+
| Apple    |
| Avnet    |
| Intel    |
| Obsidian |
| Sierra   |
+----------+
```

Производные таблицы

Подзапрос можно задать в выражении `FROM`. Результатом такого подзапроса является таблица, по которой можно выполнять поиск значений.

Например, чтобы вывести на экран список людей не из США старше 63 лет, а также их возраст и город, нужно выполнить следующий запрос:

```sql
SELECT pp.name, pp.town, pp.age
FROM
    (SELECT p.name, p.town, p.age
    FROM people AS p
    WHERE p.country != 'USA')
AS pp
WHERE pp.age>63;
```

Результат:

```
+------------------+-----------+-----+
|       NAME       |   TOWN    | AGE |
+------------------+-----------+-----+
| Bryant Addie     | Izamal    | 64  |
| Chavez Don       | Chetumal  | 65  |
| Cohen Victor     | Chetumal  | 64  |
| Cox Lou          | Tijuana   | 64  |
| Craig Bruce      | Chihuahua | 65  |
| Elliott Callie   | Juarez    | 65  |
| Flores Lottie    | Uman      | 64  |
| Flores Nannie    | Chihuahua | 64  |
| Frank Chester    | Nogales   | 65  |
| Gill Bessie      | Nogales   | 64  |
| Hunter Curtis    | Mexicali  | 65  |
| Keller Loretta   | Tijuana   | 65  |
| Little Antonio   | Juarez    | 64  |
| Massey Hettie    | Tijuana   | 65  |
| McGee Lulu       | Cancun    | 64  |
| Mitchell Gary    | Uman      | 64  |
| Mitchell Jason   | Ensenada  | 65  |
| Riley Terry      | Juarez    | 64  |
| Torres Katharine | Juarez    | 64  |
| Wheeler Oscar    | Merida    | 64  |
| Wise Andre       | Cancun    | 64  |
+------------------+-----------+-----+
```

Ограничения

На текущий момент есть два ограничения на использование подзапросов в Phoenix:

- Нельзя использовать подзапрос как элемент выражения `SELECT`.
- Подзапрос не может иметь выражение `HAVING`.

Последовательности в Phoenix

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
- Создание последовательности
- Удаление последовательности
- Получение новых значений
  - Команда UPSERT VALUES
  - Команда SELECT
- Просмотр последнего значения
- Просмотр информации о последовательности
- Особенности использования последовательностей

Обзор

Последовательность — это объект, последовательно генерирующий монотонно возрастающий или убывающий ряд целых чисел согласно заданным условиям. Такие ряды в основном используются для создания идентификаторов, но могут использоваться и в иных целях. Последовательности могут принадлежать или не принадлежать схемам.

Создание последовательности

Для создания последовательности используется следующий синтаксис:

```
CREATE SEQUENCE [IF NOT EXISTS] {[<schema_name>.]<sequence_name>}
[START [WITH] <number>]
[INCREMENT [BY] <number>]
[MINVALUE <number>]
[MAXVALUE <number>]
[CYCLE]
[CACHE <number>];
```

Параметры и ключевые слова команды CREATE SEQUENCE

Параметр или ключевое слово	Описание
IF NOT EXISTS	Попытка создания последовательности с именем уже существующей последовательности без этого ключевого слова приведет к выводу на экран сообщения об ошибке. По умолчанию не используется
schema_name	Имя схемы, которой принадлежит последовательность
sequence_name	Имя последовательности
START (WITH)	Значение начала последовательности. Оно возвращается в качестве первого значения, генерируемого последовательностью. Это значение должно находиться в промежутке, заданном значениями ключевых слов `MINVALUE` и `MAXVALUE`, включая граничные значения. Ключевое слово `WITH` необязательно (ни на что не влияет). Значение по умолчанию: `1`
number	Численное значение, требуемое для ключевых слов `START WITH`, `INCREMENT BY`, `MINVALUE`, `MAXVALUE` и `CACHE`
INCREMENT (BY)	Значение шага возрастания или убывания последовательности. Оно добавляется к текущему значению последовательности для получения следующего значения. Ключевое слово `BY` необязательно (ни на что не влияет). Значение по умолчанию: `1`
MINVALUE	Минимальное значение последовательности. Используется в качестве предельной точки для убывающих последовательностей, а также начальной точки для возрастающих последовательностей, использующих зацикливание после достижения максимального значения. Значение по умолчанию: `-9223372036854775808`
MAXVALUE	Максимальное значение последовательности. Используется в качестве предельной точки для возрастающих последовательностей, а также начальной точки для убывающих последовательностей, использующих зацикливание после достижения минимального значения. Значение по умолчанию: `9223372036854775807`
CYCLE	Определяет, нужно ли при выходе за пределы заданного промежутка зациклить последовательность, то есть продолжать её от минимального значения (или от максимального для убывающих последовательностей), или выдать ошибку. По умолчанию не используется
CACHE	Размер порции последовательных значений, хранящихся на стороне клиента, по исчерпании которых он сможет запросить следующую порцию с сервера. Одну и ту же последовательность могут использовать несколько клиентов. Каждый раз, когда любой из клиентов запрашивает новую порцию значений, он будет получать следующую в очереди. Значение по умолчанию: `100`. Оно может быть изменено вручную при помощи добавления параметра `phoenix.sequence.cacheSize` в секцию ***Custom hbase-site.xml*** страницы конфигурации сервиса HBase

ПРИМЕЧАНИЕ

Последовательности в Phoenix используют только тип `BIGINT` для возвращаемых значений (от `-9223372036854775808` до `9223372036854775807`).

Пример:

```
CREATE SEQUENCE IF NOT EXISTS test_schema.test_sequence
START WITH 97 INCREMENT BY 2 MINVALUE 0 MAXVALUE 100 CYCLE CACHE 50;
```

Эта команда создает последовательность, которая вернет `97` в качестве первого значения, затем `99`, а после этого будет циклично возвращать четные числа от `0` до `100` в возрастающем порядке. Первые 50 значений будут храниться на стороне клиента и последовательно возвращаться согласно исполняемым запросам, а после их исчерпания будут запрошены следующие 50 значений с сервера.

Удаление последовательности

Для удаления последовательности используется следующий синтаксис:

```
DROP SEQUENCE [IF EXISTS] <sequence_ref>;
```

где `<sequence_ref>` — точное название последовательности, как оно было указано при изначальном её создании. Если последовательность принадлежит схеме, то её тоже нужно указать в этой команде.

Пример:

```
DROP SEQUENCE IF EXISTS test_schema.test_sequence;
```

Получение новых значений

Для получения нового значения последовательности используется выражение `NEXT VALUE FOR` с указанием имени последовательности в командах `SELECT` и `UPSERT VALUES`. Последовательность, вызываемая в первый раз, возвращает начальное значение (заданное ключевым словом или по умолчанию).

Команда UPSERT VALUES

Например, чтобы создать таблицу и наполнить её значениями, используя последовательность для генерации первичных ключей, нужно выполнить следующие команды:

```
CREATE TABLE weapon_types (
id INTEGER NOT NULL PRIMARY KEY,
wpn_type VARCHAR(50),
ammo_type VARCHAR(50));

CREATE SEQUENCE IF NOT EXISTS test_schema.test_sequence START 1 INCREMENT 1;

UPSERT INTO weapon_types(id, wpn_type, ammo_type) VALUES (NEXT VALUE FOR test_schema.test_sequence, 'handgun', 'bullet');
UPSERT INTO weapon_types(id, wpn_type, ammo_type) VALUES (NEXT VALUE FOR test_schema.test_sequence, 'shotgun', 'buckshot');
UPSERT INTO weapon_types(id, wpn_type, ammo_type) VALUES (NEXT VALUE FOR test_schema.test_sequence, 'shotgun', 'slug');
UPSERT INTO weapon_types(id, wpn_type, ammo_type) VALUES (NEXT VALUE FOR test_schema.test_sequence, 'assault rifle', 'bullet');
UPSERT INTO weapon_types(id, wpn_type, ammo_type) VALUES (NEXT VALUE FOR test_schema.test_sequence, 'sniper rifle', 'bullet');
UPSERT INTO weapon_types(id, wpn_type, ammo_type) VALUES (NEXT VALUE FOR test_schema.test_sequence, 'crossbow', 'arrow');

SELECT * FROM weapon_types;
```

Результат:

```
+----+---------------+-----------+
| ID |   WPN_TYPE    | AMMO_TYPE |
+----+---------------+-----------+
| 1  | handgun       | bullet    |
| 2  | shotgun       | buckshot  |
| 3  | shotgun       | slug      |
| 4  | assault rifle | bullet    |
| 5  | sniper rifle  | bullet    |
| 6  | crossbow      | arrow     |
+----+---------------+-----------+
```

Команда SELECT

Для загрузки значений из одной таблицы в другую с помощью одной команды `UPSERT` можно использовать команду `SELECT`. Например, чтобы создать таблицу, содержащую типы боеприпасов из ранее созданной таблицы, используя последовательность для генерации первичных ключей, нужно выполнить следующие команды:

```
CREATE TABLE ammo_types (
id INTEGER NOT NULL PRIMARY KEY,
ammo_type VARCHAR(50));

CREATE SEQUENCE IF NOT EXISTS test_schema.test_sequence2 START 1 INCREMENT 1;

UPSERT INTO ammo_types(id, ammo_type)
SELECT NEXT VALUE FOR test_schema.test_sequence2, ammo_type
FROM weapon_types
GROUP BY ammo_type;

SELECT * FROM ammo_types;
```

Результат:

```
+----+-----------+
| ID | AMMO_TYPE |
+----+-----------+
| 1  | arrow     |
| 2  | buckshot  |
| 3  | bullet    |
| 4  | slug      |
+----+-----------+
```

Просмотр последнего значения

Для просмотра последнего значения, сгенерированного последовательностью, используйте в команде `SELECT` выражение `CURRENT VALUE FOR` с указанием имени последовательности. Пример:

```
SELECT CURRENT VALUE FOR test_schema.test_sequence;
```

Результат:

```
+--------------------------------------------+
| CURRENT VALUE FOR TEST_SCHEMA.TEST_SEQUENCE |
+--------------------------------------------+
| 4                                          |
+--------------------------------------------+
```

ПРИМЕЧАНИЕ

Эта команда вернет сообщение об ошибке, если последовательность еще ни разу не вызывалась для генерации значений.

Если выражение `CURRENT VALUE FOR` используется совместно с выражением `NEXT VALUE FOR` для одной и той же последовательности в рамках одной команды, они оба вернут одно и то же вновь сгенерированное значение.

Просмотр информации о последовательности

Для просмотра информации о всех последовательностях выполните следующую команду:

```
SELECT * FROM SYSTEM."SEQUENCE";
```

Результат:

```
+-----------+-----------------+---------------+------------+---------------+--------------+-----------+----------------------+--
------------------+-----------+--------------------+
| TENANT_ID | SEQUENCE_SCHEMA | SEQUENCE_NAME | START_WITH | CURRENT_VALUE | INCREMENT_BY | CACHE_SIZE |      MIN_VALUE       |
MAX_VALUE       | CYCLE_FLAG | LIMIT_REACHED_FLAG |
+-----------+-----------------+---------------+------------+---------------+--------------+-----------+----------------------+--
------------------+-----------+--------------------+
|           | TEST_SCHEMA     | TEST_SEQUENCE | 1          | 101           | 1            | 100        | -9223372036854775808 |
9223372036854775807 | false      | false              |
+-----------+-----------------+---------------+------------+---------------+--------------+-----------+----------------------+--
------------------+-----------+--------------------+
```

Обратите внимание, что значение столбца `CURRENT_VALUE` для последовательности равно `101`. Причина этого в том, что размер порции значений равен 100 (столбец `CACHE_SIZE`) и первые 100 значений уже были отправлены клиенту. Если создать еще одну сессию подключения к Phoenix Query Server, она будет рассматриваться как новый клиент. Если в этом случае запросить следующее значение этой же последовательности, будет сгенерировано значение `101`, а значение столбца `CURRENT_VALUE` станет равным `201`. После исчерпания клиентом своей порции значений он получит следующую с сервера. Начальное значение этой порции будет зависеть от того, сколько порций было передано ранее другим клиентам.

Таблица `SYSTEM.SEQUENCE` доступна только для чтения.

Особенности использования последовательностей

При использовании последовательностей имейте в виду следующее:

- После создания последовательности изменить ее нельзя.
- Выражения `NEXT VALUE FOR` и `CURRENT VALUE FOR` могут использоваться только командами `SELECT` и `UPSERT VALUES`. Использование этих выражений в командах `WHERE`, `GROUP BY`, `HAVING` или `ORDER BY` приведет к ошибке.
- Выражение `NEXT VALUE FOR` генерирует новое значение только один раз для одной команды `SELECT` или `UPSERT`. Например, используется команда следующего вида:

```
UPSERT INTO test_table(c1,c2) VALUES (NEXT VALUE FOR some_sequence, NEXT VALUE FOR some_sequence);
```

В этом случае значения в обоих столбцах будут одинаковы.

Использование salted-таблиц в Phoenix

Владимир Адаменков

Содержание

Назад к содержанию

Обзор
Использование
Особенности

Обзор

HBase распределяет свои данные по регионам, которые располагаются на серверах регионов. Регионы определяются диапазонами ключей строк. В некоторых случаях регионы с данными, принадлежащими небольшому диапазону пространства ключей, могут испытывать высокую нагрузку (как на запись, так и на чтение), в то время как другие регионы могут оставаться относительно ненагруженными. Обычно это происходит, когда ключи строк представляют собой некоторую монотонную последовательность (целочисленные идентификаторы, метки времени). Это называется **hotspotting**. Он приводит к снижению производительности и ненужным разделениям регионов.

Во избежание таких ситуаций в Apache Phoenix используется механизм под названием **salting**. При использовании salting к ключам строк добавляется префикс, который называется **соль** — один байт в случае Apache Phoenix. Это делается для того, чтобы соседние ключи имели более высокий шанс попасть в разные регионы. Это приводит к более равномерному распределению данных и более высокому параллелизму записи. Производительность чтения также может вырасти, но необязательно (см. Особенности).

Использование

Чтобы создать salted-таблицу, нужно задать количество префиксов ключей строк (**salt buckets**) для использования в таблице, например:

```
CREATE TABLE people (
name VARCHAR(50) NOT NULL PRIMARY KEY,
age SMALLINT,
town VARCHAR(50),
state VARCHAR(50),
country VARCHAR(50))
SALT_BUCKETS = 10;
```

Эта команда создает пустую таблицу с десятью salt bucket. Теперь можно загружать данные в таблицу. Чтобы иметь возможность выполнять примеры, приведенные в этой статье, нужно импортировать данные из файла с тестовыми данными people.csv, как описано в статье Оператор JOIN в Phoenix. Используйте таблицу, созданную в примере выше.

Чтобы увидеть, как salting влияет на ключи строк, выполните следующий запрос:

```
SELECT * FROM people
WHERE people.name > 'X';
```

Этот запрос должен вывести на экран всех людей, чьи фамилии начинаются на `Y` или `Z`. Результат:

```
+-----------------+-----+-------------+-----------------+---------+
|      NAME       | AGE |    TOWN     |      STATE      | COUNTRY |
+-----------------+-----+-------------+-----------------+---------+
| Young Della     | 21  | Chetumal    | Quintana Roo    | MEX     |
| Yates Bill      | 48  | Juarez      | Chihuahua       | MEX     |
| Yates Francis   | 28  | Juarez      | Chihuahua       | MEX     |
| Yates Justin    | 47  | New Orleans | LA              | USA     |
| Yates Danny     | 47  | Merida      | Yucatan         | MEX     |
| Yates Douglas   | 35  | Phoenix     | AZ              | USA     |
| Zimmerman Madge | 46  | Nogales     | AZ              | USA     |
| Young Mattie    | 39  | San Jose    | CA              | USA     |
| Yates George    | 39  | Chihuahua   | Chihuahua       | MEX     |
| Yates Lucile    | 25  | Chihuahua   | Chihuahua       | MEX     |
| Zimmerman Gene  | 35  | Tijuana     | California Baja | MEX     |
| Yates Johnny    | 20  | Phoenix     | AZ              | USA     |
| Young Catherine | 60  | Juarez      | Chihuahua       | MEX     |
| Young Josephine | 29  | Redding     | OR              | USA     |
+-----------------+-----+-------------+-----------------+---------+
14 rows selected (0.019 seconds)
```

Как видно, фамилии перечислены не в алфавитном порядке, хотя HBase должен сортировать ключи строк. Это произошло потому, что фамилии имеют префиксы, не отображаемые в выводе, и сортировка происходит по этим префиксам. Вы можете добиться вывода фамилий в алфавитном порядке, выполнив явным образом сортировку значений в столбце. Пример:

```
SELECT * FROM people
WHERE people.name > 'X'
ORDER BY people.name;
```

Результат:

```
+-----------------+-----+-------------+-----------------+---------+
|      NAME       | AGE |    TOWN     |      STATE      | COUNTRY |
+-----------------+-----+-------------+-----------------+---------+
| Yates Bill      | 48  | Juarez      | Chihuahua       | MEX     |
| Yates Danny     | 47  | Merida      | Yucatan         | MEX     |
| Yates Douglas   | 35  | Phoenix     | AZ              | USA     |
| Yates Francis   | 28  | Juarez      | Chihuahua       | MEX     |
| Yates George    | 39  | Chihuahua   | Chihuahua       | MEX     |
| Yates Johnny    | 20  | Phoenix     | AZ              | USA     |
| Yates Justin    | 47  | New Orleans | LA              | USA     |
| Yates Lucile    | 25  | Chihuahua   | Chihuahua       | MEX     |
| Young Catherine | 60  | Juarez      | Chihuahua       | MEX     |
| Young Della     | 21  | Chetumal    | Quintana Roo    | MEX     |
| Young Josephine | 29  | Redding     | OR              | USA     |
| Young Mattie    | 39  | San Jose    | CA              | USA     |
| Zimmerman Gene  | 35  | Tijuana     | California Baja | MEX     |
| Zimmerman Madge | 46  | Nogales     | AZ              | USA     |
+-----------------+-----+-------------+-----------------+---------+
14 rows selected (0.024 seconds)
```

В этом случае, когда выражение `ORDER BY` используется для столбца ключей строк, соль игнорируется.

Для просмотра списка salted-таблиц выполните следующий запрос:

```
SELECT table_name, salt_buckets
FROM SYSTEM.CATALOG
WHERE salt_buckets IS NOT NULL;
```

Пример результата:

```
+---------------+--------------+
|  TABLE_NAME   | SALT_BUCKETS |
+---------------+--------------+
| PEOPLE        | 10           |
| PEOPLE_SALTED | 10           |
| LOG           | 32           |
+---------------+--------------+
```

Особенности

- Количество salt bucket (диапазон возможных значений соли) следует выбирать в зависимости от размера кластера и ожидаемого объема записи. Имеет смысл начать с количества из расчета одно значение соли на один сервер регионов. Далее можно экспериментировать для получения оптимального значения.
- Обычно salting улучшает производительность записи, но потенциально может привести к ухудшению производительности чтения, если запросы на чтение зависят от исходного порядка ключей строк. Phoenix будет автоматически выполнять сканирование по всем значениям соли, что может быть менее эффективно, чем сканирование конкретного региона.
- Salting привносит сложность. Полезно понимать распределение данных в кластере и паттерны осуществления доступа к ним для определения целесообразности использования salting.
- Возможно, придется внести изменения в тексты запросов для учета salting. Например, может потребоваться использовать оператор `LIKE` совместно с wildcard для маскировки соли в ключе.
- Salting может привести к нелокальности данных. Если необходим постоянный доступ к данным, связанным близкими оригинальными значениями ключей, salting вынужденно приведет к сканированию нескольких регионов.

Использование существующих таблиц HBase

Альберт Багдасарян

Назад к содержанию

Раздел Использование SQLLine для работы с Phoenix объясняет, как создавать новые таблицы в HBase с помощью Phoenix. Однако Phoenix позволяет также работать с существующими таблицами HBase в режиме чтения. Для этой цели создают представления (view) таблицы HBase. При их создании оригинальные метаданные HBase остаются неизменными.

Рассмотрим для примера таблицу `people_ages`. Используйте HBase shell для проверки наличия этой таблицы:

```
list
```

Пример вывода:

```
TABLE
BEST_BOOKS
SYSTEM.CATALOG
SYSTEM.FUNCTION
SYSTEM.LOG
SYSTEM.MUTEX
SYSTEM.SEQUENCE
SYSTEM.STATS
ns1:temp2
people_ages
t4
t5
t6
table10
temp1
temp10
wordcount
wordcount2
wordcount3
wordcount4
19 row(s)
Took 0.0428 seconds
=> ["BEST_BOOKS", "SYSTEM.CATALOG", "SYSTEM.FUNCTION", "SYSTEM.LOG", "SYSTEM.MUTEX", "SYSTEM.SEQUENCE", "SYSTEM.STATS",
"ns1:temp2", "people_ages", "t4", "t5", "t6", "table10", "temp1", "temp10", "wordcount", "wordcount2", "wordcount3",
"wordcount4"]
```

Эта таблица пока не доступна в Phoenix. Проверьте это с помощью соответствующей команды в SQLLine:

```
!tables
```

Вывод на экран не содержит `people_ages`:

```
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------+
| TABLE_CAT  | TABLE_SCHEM  | TABLE_NAME  |  TABLE_TYPE   | REMARKS  | TYPE_NAME  | SELF_REFERENCING_COL_NAME  | REF_GENERATION
| INDEX_STATE  | IMMUTABLE |
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------+
|            | SYSTEM       | CATALOG     | SYSTEM TABLE  |          |            |                            |
|              | false     |
|            | SYSTEM       | FUNCTION    | SYSTEM TABLE  |          |            |                            |
|              | false     |
|            | SYSTEM       | LOG         | SYSTEM TABLE  |          |            |                            |
|              | true      |
|            | SYSTEM       | SEQUENCE    | SYSTEM TABLE  |          |            |                            |
|              | false     |
|            | SYSTEM       | STATS       | SYSTEM TABLE  |          |            |                            |
|              | false     |
|            |              | BEST_BOOKS  | TABLE         |          |            |                            |
|              | false     |
+------------+--------------+-------------+---------------+----------+------------+----------------------------+-----------------
+--------------+-----------+
```

Для создания представления (view) в Phoenix выполните следующую команду в SQLLine:

```
CREATE VIEW "people_ages" ("pk" VARCHAR PRIMARY KEY, "basic"."age" UNSIGNED_SMALLINT);
```

Теперь с помощью Phoenix можно выбирать данные из оригинальной таблицы HBase:

```
SELECT * FROM "people_ages" LIMIT 10;
```

Вывод на экран:

```
+--------------------+--------+
|         pk         |  age   |
+--------------------+--------+
| Abbott Delia       | 13874  |
| Abbott Howard      | 12852  |
| Abbott Jack        | 12857  |
| Adams Clyde        | 12857  |
| Aguilar Myrtie     | 12851  |
| Aguilar Terry      | 13877  |
| Alexander Derrick  | 13366  |
| Alexander Gregory  | 13620  |
| Alexander Leon     | 13362  |
| Allen Austin       | 13108  |
+--------------------+--------+
10 rows selected (0.029 seconds)
```

Определите количество строк в таблице:

```
SELECT COUNT(*) FROM "people_ages";
```

Вывод на экран:

```
+-----------+
| COUNT(1)  |
+-----------+
| 997       |
+-----------+
1 row selected (0.039 seconds)
```

Теперь в списке таблиц Phoenix вы увидите представление `people_ages`. Вы можете удалить его позднее, используя команду DROP VIEW.

```
!tables
```

Вывод на экран:

```
+------------+--------------+--------------+---------------+----------+------------+----------------------------+----------------
-+--------------+----------+
| TABLE_CAT  | TABLE_SCHEM  |  TABLE_NAME  |  TABLE_TYPE   | REMARKS  | TYPE_NAME  | SELF_REFERENCING_COL_NAME  | REF_GENERATION
| INDEX_STATE  | IMMUTABL |
+------------+--------------+--------------+---------------+----------+------------+----------------------------+----------------
-+--------------+----------+
|            | SYSTEM       | CATALOG      | SYSTEM TABLE  |          |            |                            |
|              | false    |
|            | SYSTEM       | FUNCTION     | SYSTEM TABLE  |          |            |                            |
|              | false    |
|            | SYSTEM       | LOG          | SYSTEM TABLE  |          |            |                            |
|              | true     |
|            | SYSTEM       | SEQUENCE     | SYSTEM TABLE  |          |            |                            |
|              | false    |
|            | SYSTEM       | STATS        | SYSTEM TABLE  |          |            |                            |
|              | false    |
|            |              | BEST_BOOKS   | TABLE         |          |            |                            |
|              | false    |
|            |              | people_ages  | VIEW          |          |            |                            |
|              | false    |
+------------+--------------+--------------+---------------+----------+------------+----------------------------+----------------
-+--------------+----------+
```

Для представлений в Phoenix поддерживается только чтение, поэтому при попытке изменить данные в них вы получите сообщение об ошибке, как показано ниже:

Сообщение об ошибке

```
0: jdbc:phoenix:> UPSERT INTO "people_ages" values('Test Test', 100);
Error: ERROR 505 (42000): Table is read only. (state=42000,code=505)
org.apache.phoenix.schema.ReadOnlyTableException: ERROR 505 (42000): Table is read only.
        at org.apache.phoenix.compile.UpsertCompiler.compile(UpsertCompiler.java:359)
        at org.apache.phoenix.jdbc.PhoenixStatement$ExecutableUpsertStatement.compilePlan(PhoenixStatement.java:784)
        at org.apache.phoenix.jdbc.PhoenixStatement$ExecutableUpsertStatement.compilePlan(PhoenixStatement.java:770)
        at org.apache.phoenix.jdbc.PhoenixStatement$2.call(PhoenixStatement.java:401)
        at org.apache.phoenix.jdbc.PhoenixStatement$2.call(PhoenixStatement.java:391)
        at org.apache.phoenix.call.CallRunner.run(CallRunner.java:53)
        at org.apache.phoenix.jdbc.PhoenixStatement.executeMutation(PhoenixStatement.java:390)
        at org.apache.phoenix.jdbc.PhoenixStatement.executeMutation(PhoenixStatement.java:378)
        at org.apache.phoenix.jdbc.PhoenixStatement.execute(PhoenixStatement.java:1825)
        at sqlline.Commands.execute(Commands.java:822)
        at sqlline.Commands.sql(Commands.java:732)
        at sqlline.SqlLine.dispatch(SqlLine.java:813)
        at sqlline.SqlLine.begin(SqlLine.java:686)
        at sqlline.SqlLine.start(SqlLine.java:398)
        at sqlline.SqlLine.main(SqlLine.java:291)
```

Массовая загрузка в Phoenix

Альберт Багдасарян

Содержание

Назад к содержанию

Метод 1. Использование psql

Метод 2. Использование MapReduce

Массовая загрузка данных доступна не только в HBase, но также и в Phoenix. Существуют два метода:

- **Использование psql**. Psql — однопоточный инструмент, хорошо подходящий для загрузки нескольких мегабайт или гигабайт данных. Он основан на типовой последовательности записи в HBase без применения прямой записи в HFiles.
  Названия загружаемых файлов должны иметь расширение *.sql* или *.csv*. Первая группа файлов используется для формирования запросов SQL, а другая — для загрузки данных.
- **Использование MapReduce**. Этот метод основан на обработке вида ETL (Extract, Transform, Load). Файлы загружают в HDFS и затем преобразуют с помощью встроенного задания MapReduce CsvBulkLoadTool. Преобразованные данные загружают напрямую в HFiles с помощью той же утилиты. Этот метод позволяет обрабатывать гораздо больший объем данных по сравнению с psql, благодаря многопоточности процесса MapReduce. Таким образом, применение MapReduce рекомендуется для рабочих (production) процессов.
  Поддерживаемые расширения названий файлов: *.csv* и *.json*.

ПРИМЕЧАНИЕ

Более подробная информация о массовой загрузке данных приведена в документации Phoenix.

Метод 1. Использование psql

1. Подготовьте SQL-файл *best_books.sql*, содержащий запрос на создание таблицы `BEST_BOOKS` со следующей структурой:

```
CREATE TABLE IF NOT EXISTS best_books
        (author VARCHAR not null,
        title VARCHAR not null,
        public_year UNSIGNED_SMALLINT,
        CONSTRAINT pk PRIMARY KEY (author, title));
```

   Таблица будет содержать данные об известных книгах, включая их авторов, заголовки и даты их первых публикаций. Загрузите файл по ссылке best_books.sql.

ВНИМАНИЕ

По некоторым причинам psql не позволяет использовать названия таблиц с использованием букв нижнего регистра. Если вы попытаетесь дать таблице название `"best_books"`, это приведет к ошибкам в процессе загрузки данных с помощью psql, хотя ручная загрузка без этой утилиты проходит нормально. Поэтому мы рекомендуем не заключать в двойные кавычки названия таблиц. Отсутствие кавычек позволяет утилите конвертировать буквы в верхний регистр. Это ограничение не затрагивает названия колонок.

2. Подготовьте два CSV-файла для загрузки данных в таблицу `BEST_BOOKS`: best_books1.csv и best_books2.csv.
3. Подготовьте файл *best_books_query.sql* для запроса данных из таблицы `BEST_BOOKS`:

```
SELECT
        public_year as "Year",
        count(*) as "Books count"
FROM best_books
GROUP BY public_year
ORDER BY public_year;
```

   Этот запрос вернет список, каждая строка которого будет содержать год и количество книг, опубликованных в этом году. Вы можете загрузить файл по ссылке best_books_query.sql.

4. Поместите все подготовленные файлы в локальную файловую систему одного из узлов кластера HBase. С помощью следующей команды убедитесь, что файлы находятся там:

```
$ ls -la ~
```

   Пример вывода на экран:

```
total 96
drwx------. 6 dasha dasha  4096 Dec  1 14:33 .
drwxr-xr-x. 3 root  root     19 Aug 31 11:54 ..
drwx------. 3 dasha dasha    17 Aug 31 15:06 .ansible
-rw-------. 1 dasha dasha 23386 Dec  1 12:45 .bash_history
-rw-r--r--. 1 dasha dasha    18 Apr  1  2020 .bash_logout
-rw-r--r--. 1 dasha dasha   193 Apr  1  2020 .bash_profile
-rw-r--r--. 1 dasha dasha   231 Apr  1  2020 .bashrc
-rw-rw-r--. 1 dasha dasha  2736 Dec  1 14:33 best_books1.csv
-rw-rw-r--. 1 dasha dasha  1140 Dec  1 14:33 best_books2.csv
-rw-rw-r--. 1 dasha dasha   122 Dec  1 14:33 best_books_query.sql
-rw-rw-r--. 1 dasha dasha   176 Dec  1 14:33 best_books.sql
drwxrwxrwx. 2 dasha dasha    64 Nov 26 07:45 dasha
-rw-rw-r--. 1 dasha dasha 17651 Nov 26 07:46 people_ages.csv
drwxrwxr-x. 2 dasha dasha    21 Nov 30 14:04 .sqlline
drwx------. 2 dasha dasha    29 Sep 23 17:35 .ssh
```

5. Перейдите в каталог с исполняемым файлом:

```
$ cd /usr/lib/phoenix/bin
```

6. Используйте приведенную ниже команду запуска psql. Аргумент `-t` задает название таблицы. Его следует задавать явно, если оно не совпадает с названием CSV-файла (которое используется по умолчанию). Аргумент `-d` указывает на разделитель, используемый в CSV-файле (по умолчанию это запятая `,`). Другие возможные аргументы приведены в документации Phoenix.
   Можно задать одновременно несколько файлов, учитывая при этом, что они обрабатываются последовательно, и, следовательно, записывать их надо в правильном порядке. В данном примере мы сначала создаем таблицу (файл *best_books.sql*), затем загружаем в нее данные (*best_books1.csv*) и в конце делаем запрос SQL (*best_books_query.sql*):

```
$ ./psql.py -t BEST_BOOKS -d '|' ~/best_books.sql ~/best_books1.csv ~/best_books_query.sql
```

Вывод на экран

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-client.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
21/12/01 16:13:18 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
no rows upserted
Time: 1.381 sec(s)

csv columns from database.
CSV Upsert complete. 70 rows upserted
Time: 0.068 sec(s)
                                    Year                              Books count
---------------------------------------- ----------------------------------------
                                    1924                                        1
                                    1925                                        2
                                    1926                                        1
                                    1927                                        3
                                    1929                                        2
                                    1932                                        1
                                    1934                                        1
                                    1936                                        1
                                    1937                                        1
                                    1939                                        2
                                    1940                                        3
                                    1945                                        2
                                    1946                                        1
                                    1947                                        1
                                    1949                                        1
                                    1950                                        1
                                    1951                                        1
                                    1952                                        1
                                    1953                                        1
                                    1954                                        2
                                    1955                                        2
                                    1957                                        1
                                    1958                                        1
                                    1959                                        1
                                    1960                                        2
                                    1961                                        4
                                    1962                                        3
                                    1963                                        1
                                    1965                                        1
                                    1966                                        2
                                    1969                                        4
                                    1970                                        2
                                    1973                                        1
                                    1975                                        1
                                    1981                                        1
                                    1984                                        1
                                    1985                                        2
                                    1987                                        2
                                    1990                                        1
                                    1992                                        1
                                    1996                                        1
                                    1997                                        1
                                    2000                                        2
                                    2001                                        2
                                    2005                                        1
Time: 0.027 sec(s)
```

   Таким образом, команда выполняет три операции последовательно: создает таблицу, вставляет данные и запрашивает результат.

7. Проверьте результат выполнения команды. Для этого в среде SQLLine выполните запрос на вывод первых десяти строк из созданной таблицы:

```
SELECT * FROM best_books LIMIT 10;
```

   Вывод на экран:

```
+-----------------------+-----------------------------------------------------+--------------+
|        AUTHOR         |                        TITLE                        | PUBLIC_YEAR  |
+-----------------------+-----------------------------------------------------+--------------+
| A.S. Byatt            | Possession                                          | 1990         |
| Alan Moore            | Watchmen                                            | 1987         |
| Anthony Burgess       | A Clockwork Orange                                  | 1962         |
| C.S. Lewis            | The Lion, the Witch and the Wardrobe                | 1950         |
| Carson McCullers      | The Heart Is a Lonely Hunter                        | 1940         |
| Chinua Achebe         | Chinua Achebe                                       | 1958         |
| Cormac McCarthy       | Blood Meridian, or the Evening Redness in the West  | 1985         |
| Dashiell Hammett      | Red Harvest                                         | 1929         |
| David Foster Wallace  | Infinite Jest                                       | 1996         |
| Don DeLillo           | White Noise                                         | 1985         |
+-----------------------+-----------------------------------------------------+--------------+
10 rows selected (0.059 seconds)
```

   Используйте HBase shell для сканирования созданной таблицы. Первые десять значений приведены ниже. Обратите внимание на комплексную структуру первичного ключа таблицы. Два поля, из которых он состоит, записаны в виде единого ключа строки в таблице HBase (`AUTHOR` + `TITLE`):

```
scan 'BEST_BOOKS'
```

   Вывод на экран:

```
ROW                                                               COLUMN+CELL
 A.S. Byatt\x00Possession                                         column=0:\x00\x00\x00\x00,
timestamp=1638375201460, value=x
 A.S. Byatt\x00Possession                                         column=0:\x80\x0B,
timestamp=1638375201460, value=\x07\xC6
 Alan Moore\x00Watchmen                                           column=0:\x00\x00\x00\x00,
timestamp=1638375201460, value=x
 Alan Moore\x00Watchmen                                           column=0:\x80\x0B,
timestamp=1638375201460, value=\x07\xC3
 Anthony Burgess\x00A Clockwork Orange                            column=0:\x00\x00\x00\x00,
timestamp=1638375201460, value=x
 Anthony Burgess\x00A Clockwork Orange                            column=0:\x80\x0B,
timestamp=1638375201460, value=\x07\xAA
 C.S. Lewis\x00The Lion, the Witch and the Wardrobe               column=0:\x00\x00\x00\x00,
timestamp=1638375201460, value=x
 C.S. Lewis\x00The Lion, the Witch and the Wardrobe               column=0:\x80\x0B,
timestamp=1638375201460, value=\x07\x9E
 Carson McCullers\x00The Heart Is a Lonely Hunter                 column=0:\x00\x00\x00\x00,
timestamp=1638375201460, value=x
 Carson McCullers\x00The Heart Is a Lonely Hunter                 column=0:\x80\x0B,
timestamp=1638375201460, value=\x07\x94
```

8. После создания таблицы запустите psql еще раз для загрузки второй части данных. На этот раз указываем только два файла (первый файл со скриптом более не нужен):

```
$ ./psql.py -t BEST_BOOKS -d '|' ~/best_books2.csv ~/best_books_query.sql
```

Вывод на экран

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-client.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
21/12/01 16:16:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
csv columns from database.
CSV Upsert complete. 30 rows upserted
Time: 0.065 sec(s)

                                    Year                              Books count
---------------------------------------- ----------------------------------------
                                    1924                                        1
                                    1925                                        3
                                    1926                                        1
                                    1927                                        3
                                    1929                                        2
                                    1932                                        1
                                    1934                                        5
                                    1936                                        1
                                    1937                                        1
                                    1938                                        1
                                    1939                                        4
                                    1940                                        4
                                    1945                                        4
                                    1946                                        1
                                    1947                                        1
                                    1948                                        1
                                    1949                                        2
                                    1950                                        1
                                    1951                                        1
                                    1952                                        1
                                    1953                                        2
                                    1954                                        3
                                    1955                                        4
                                    1957                                        3
                                    1958                                        1
                                    1959                                        1
                                    1960                                        3
                                    1961                                        5
                                    1962                                        4
                                    1963                                        1
                                    1964                                        1
                                    1965                                        1
                                    1966                                        2
                                    1967                                        1
                                    1969                                        4
                                    1970                                        3
                                    1973                                        1
                                    1974                                        1
                                    1975                                        1
                                    1977                                        1
                                    1980                                        1
                                    1981                                        1
                                    1984                                        2
                                    1985                                        2
                                    1986                                        1
                                    1987                                        2
                                    1990                                        1
                                    1992                                        1
                                    1996                                        1
                                    1997                                        1
                                    2000                                        2
                                    2001                                        2
                                    2005                                        1
Time: 0.033 sec(s)
```

9. В завершение убедитесь, что данные, загруженные в HBase через psql, пока не содержатся в HFiles. Для этого выполните команду `list_regions`. Значение `0` в колонке `LOCALITY` доказывает это:

```
list_regions 'BEST_BOOKS'
```

   Вывод на экран:

```
                                 SERVER_NAME |                                                  REGION_NAME |  START_KEY |
END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | ------------------------------------------------------------ | ---------- |
---------- | ----- | ----- | ---------- |
 bds-adh-2.ru-central1.internal,16020,1638426590048 | BEST_BOOKS,,1638430003108.36ee4f69dc1ab9d4a075091ff9ea3b80. |            |
|     0 |   140 |        0.0 |
 1 rows
Took 0.5920 seconds
```

Метод 2. Использование MapReduce

1. С помощью HBase shell очистите таблицу. Обратите внимание на то, что таблица должна быть создана через Phoenix до применения MapReduce.

```
truncate 'BEST_BOOKS'
```

   Вывод на экран:

```
Truncating 'BEST_BOOKS' table (it may take a while):
Disabling table...
Truncating table...
Took 2.1019 seconds
```

ПРИМЕЧАНИЕ

Шаги 2 и 3 необходимы, если вы не выполняли пример с массовой загрузкой в HBase, как описано в разделе Массовая загрузка данных в HBase.

2. Создайте в HDFS каталог для своего пользователя, если такого каталога еще нет:

```
$ sudo -u hdfs hdfs dfs -mkdir /user/dasha
$ hdfs dfs -ls /user
```

   Пример вывода на экран:

```
Found 5 items
drwxr-xr-x   - hdfs   hadoop          0 2021-11-26 09:56 /user/dasha
drwx------   - hdfs   hadoop          0 2021-08-31 16:15 /user/hdfs
drwxr-xr-x   - mapred hadoop          0 2021-08-31 16:22 /user/history
drwxr-xr-x   - mapred mapred          0 2021-08-31 16:21 /user/mapred
drwxr-xr-x   - yarn   yarn            0 2021-09-01 06:57 /user/yarn
```

3. Расширьте права доступа на ваш каталог с помощью следующей команды (более подробная информация о защите файлов приведена в разделе Защита файлов в HDFS):

```
$ sudo -u hdfs hdfs dfs -chmod 777 /user/dasha
$ hdfs dfs -ls /user
```

   Пример вывода на экран:

```
Found 5 items
drwxrwxrwx   - hdfs   hadoop          0 2021-11-26 09:57 /user/dasha
drwx------   - hdfs   hadoop          0 2021-08-31 16:15 /user/hdfs
drwxr-xr-x   - mapred hadoop          0 2021-08-31 16:22 /user/history
drwxr-xr-x   - mapred mapred          0 2021-08-31 16:21 /user/mapred
drwxr-xr-x   - yarn   yarn            0 2021-09-01 06:57 /user/yarn
```

4. Скопируйте файлы *best_books1.csv* и *best_books2.csv* из локальной файловой системы в HDFS:

```
$ hdfs dfs -copyFromLocal ~/best_books1.csv /user/dasha
$ hdfs dfs -copyFromLocal ~/best_books2.csv /user/dasha
```

5. Убедитесь в наличии файлов в HDFS:

```
$ hdfs dfs -ls /user/dasha
```

   Вывод на экран:

```
Found 6 items
drwx------   - dasha hadoop          0 2021-11-26 09:59 /user/dasha/.staging
-rw-r--r--   3 dasha hadoop       2736 2021-12-02 06:55 /user/dasha/best_books1.csv
-rw-r--r--   3 dasha hadoop       1140 2021-12-02 06:55 /user/dasha/best_books2.csv
drwxrwxrwx   - dasha hadoop          0 2021-11-26 09:59 /user/dasha/hbase-staging
-rw-r--r--   3 dasha hadoop      17651 2021-11-26 09:57 /user/dasha/people_ages.csv
drwxr-xr-x   - hbase hbase           0 2021-11-26 09:59 /user/dasha/test_output
```

6. С помощью описанной ниже команды запустите задание MapReduce CsvBulkLoadTool. Оно трансформирует данные и загрузит их в HFiles.
   Введите `hadoop jar` и затем определите полный путь к JAR-файлу *phoenix-<phoenix_version>-HBase-<hbase_version>client.jar*. В команде использованы следующие аргументы:

Аргументы

-t	Название таблицы (обязательный)
-d	Разделитель, используемый в CSV-файле (необязательный). Значением по умолчанию является запятая `,`
-i	Путь к CSV-файлу (обязательный)

   Другие возможные аргументы приведены в документации Phoenix.

```
$ hadoop jar /usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-client.jar org.apache.phoenix.mapreduce.CsvBulkLoadTool -d '|' -t BEST_BOOKS -
i /user/dasha/best_books1.csv
```

Вывод на экран

```
2021-12-02 07:18:15,214 INFO util.QueryUtil: Creating connection with the jdbc url: jdbc:phoenix:localhost:2181:/hbase;
2021-12-02 07:18:15,543 INFO log.QueryLoggerDisruptor: Starting  QueryLoggerDisruptor for with ringbufferSize=8192,
waitStrategy=BlockingWaitStrategy, exceptionHandler=org.apache.phoenix.log.QueryLoggerDefaultExceptionHandler@1ee563dc...
2021-12-02 07:18:15,560 INFO query.ConnectionQueryServicesImpl: An instance of ConnectionQueryServices was created.
2021-12-02 07:18:15,690 INFO zookeeper.ReadOnlyZKClient: Connect 0x5ad1da90 to localhost:2181 with session timeout=90000ms,
retries 30, retry interval 1000ms, keepAlive=60000ms
2021-12-02 07:18:15,709 INFO zookeeper.ZooKeeper: Client environment:zookeeper.version=3.4.10-
39d3a4f269333c922ed3db283be479f9deacaa0f, built on 03/23/2017 10:13 GMT
2021-12-02 07:18:15,709 INFO zookeeper.ZooKeeper: Client environment:host.name=bds-adh-1.ru-central1.internal
2021-12-02 07:18:15,709 INFO zookeeper.ZooKeeper: Client environment:java.version=1.8.0_312
2021-12-02 07:18:15,709 INFO zookeeper.ZooKeeper: Client environment:java.vendor=Red Hat, Inc.
2021-12-02 07:18:15,709 INFO zookeeper.ZooKeeper: Client environment:java.home=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.312.b07-1.
el7_9.x86_64/jre
2021-12-02 07:18:15,709 INFO zookeeper.ZooKeeper: Client environment:java.class.path=/etc/hadoop/conf:/usr/lib/hadoop/lib/jsch-
0.1.54.jar:/usr/lib/hadoop/lib/accessors-smart-1.2.jar:/usr/lib/hadoop/lib/stax2-api.jar:/usr/lib/hadoop/lib/asm-5.0.4.
jar:/usr/lib/hadoop/lib/jetty-io-9.3.24.v20180605.jar:/usr/lib/hadoop/lib/audience-annotations-0.5.0.
jar:/usr/lib/hadoop/lib/token-provider-1.0.1.jar:/usr/lib/hadoop/lib/avro-1.7.7.jar:/usr/lib/hadoop/lib/json-smart-2.3.
jar:/usr/lib/hadoop/lib/commons-beanutils-1.9.3.jar:/usr/lib/hadoop/lib/woodstox-core-5.0.3.jar:/usr/lib/hadoop/lib/commons-
cli-1.2.jar:/usr/lib/hadoop/lib/jsp-api-2.1.jar:/usr/lib/hadoop/lib/commons-codec-1.11.jar:/usr/lib/hadoop/lib/jetty-security-
9.3.24.v20180605.jar:/usr/lib/hadoop/lib/commons-collections-3.2.2.jar:/usr/lib/hadoop/lib/jsr305-3.0.0.
jar:/usr/lib/hadoop/lib/commons-compress-1.18.jar:/usr/lib/hadoop/lib/jetty-server-9.3.24.v20180605.
jar:/usr/lib/hadoop/lib/commons-configuration2-2.1.1.jar:/usr/lib/hadoop/lib/woodstox-core.jar:/usr/lib/hadoop/lib/commons-io-
2.5.jar:/usr/lib/hadoop/lib/jsr311-api-1.1.1.jar:/usr/lib/hadoop/lib/commons-lang-2.6.jar:/usr/lib/hadoop/lib/jul-to-slf4j-1.7.
25.jar:/usr/lib/hadoop/lib/commons-lang3-3.4.jar:/usr/lib/hadoop/lib/kerb-server-1.0.1.jar:/usr/lib/hadoop/lib/commons-logging-
1.1.3.jar:/usr/lib/hadoop/lib/kerb-simplekdc-1.0.1.jar:/usr/lib/hadoop/lib/commons-math3-3.1.1.
jar:/usr/lib/hadoop/lib/zookeeper-3.4.13.jar:/usr/lib/hadoop/lib/commons-net-3.6.jar:/usr/lib/hadoop/lib/kerby-config-1.0.1.
jar:/usr/lib/hadoop/lib/curator-client-2.13.0.jar:/usr/lib/hadoop/lib/jetty-servlet-9.3.24.v20180605.
jar:/usr/lib/hadoop/lib/curator-framework-2.13.0.jar:/usr/lib/hadoop/lib/kerby-asn1-1.0.1.jar:/usr/lib/hadoop/lib/curator-
recipes-2.13.0.jar:/usr/lib/hadoop/lib/gson-2.2.4.jar:/usr/lib/hadoop/lib/guava-11.0.2.jar:/usr/lib/hadoop/lib/jetty-util-9.3.
24.v20180605.jar:/usr/lib/hadoop/lib/htrace-core4-4.1.0-incubating.jar:/usr/lib/hadoop/lib/kerby-pkix-1.0.1.
jar:/usr/lib/hadoop/lib/httpclient-4.5.2.jar:/usr/lib/hadoop/lib/httpcore-4.4.4.jar:/usr/lib/hadoop/lib/jetty-webapp-9.3.24.
v20180605.jar:/usr/lib/hadoop/lib/jackson-annotations-2.7.8.jar:/usr/lib/hadoop/lib/kerby-util-1.0.1.
jar:/usr/lib/hadoop/lib/jackson-core-2.7.8.jar:/usr/lib/hadoop/lib/kerby-xdr-1.0.1.jar:/usr/lib/hadoop/lib/jackson-core-asl-1.9.
13.jar:/usr/lib/hadoop/lib/log4j-1.2.17.jar:/usr/lib/hadoop/lib/jackson-databind-2.7.8.jar:/usr/lib/hadoop/lib/metrics-core-3.2.
4.jar:/usr/lib/hadoop/lib/jackson-jaxrs-1.9.13.jar:/usr/lib/hadoop/lib/kerb-identity-1.0.1.jar:/usr/lib/hadoop/lib/jackson-
mapper-asl-1.9.13.jar:/usr/lib/hadoop/lib/nimbus-jose-jwt-4.41.1.jar:/usr/lib/hadoop/lib/jackson-xc-1.9.13.
jar:/usr/lib/hadoop/lib/javax.servlet-api-3.1.0.jar:/usr/lib/hadoop/lib/jaxb-api-2.2.11.jar:/usr/lib/hadoop/lib/netty-3.10.5.
Final.jar:/usr/lib/hadoop/lib/jaxb-impl-2.2.3-1.jar:/usr/lib/hadoop/lib/paranamer-2.3.jar:/usr/lib/hadoop/lib/jcip-annotations-
1.0-1.jar:/usr/lib/hadoop/lib/protobuf-java-2.5.0.jar:/usr/lib/hadoop/lib/jersey-core-1.19.jar:/usr/lib/hadoop/lib/re2j-1.1.
jar:/usr/lib/hadoop/lib/jersey-json-1.19.jar:/usr/lib/hadoop/lib/slf4j-api-1.7.25.jar:/usr/lib/hadoop/lib/jersey-server-1.19.
jar:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar:/usr/lib/hadoop/lib/jersey-servlet-1.19.jar:/usr/lib/hadoop/lib/jettison-1.1.
jar:/usr/lib/hadoop/lib/jetty-xml-9.3.24.v20180605.jar:/usr/lib/hadoop/lib/jetty-http-9.3.24.v20180605.
jar:/usr/lib/hadoop/lib/kerb-admin-1.0.1.jar:/usr/lib/hadoop/lib/snappy-java-1.0.5.jar:/usr/lib/hadoop/lib/kerb-client-1.0.1.
jar:/usr/lib/hadoop/lib/stax2-api-3.1.4.jar:/usr/lib/hadoop/lib/kerb-common-1.0.1.jar:/usr/lib/hadoop/lib/kerb-core-1.0.1.
jar:/usr/lib/hadoop/lib/kerb-crypto-1.0.1.jar:/usr/lib/hadoop/lib/kerb-util-1.0.1.jar:/usr/lib/hadoop/.//hadoop-annotations-3.1.
2.jar:/usr/lib/hadoop/.//hadoop-annotations.jar:/usr/lib/hadoop/.//hadoop-auth-3.1.2.jar:/usr/lib/hadoop/.//hadoop-auth.
jar:/usr/lib/hadoop/.//hadoop-common-3.1.2-tests.jar:/usr/lib/hadoop/.//hadoop-common-3.1.2.jar:/usr/lib/hadoop/.//hadoop-
common.jar:/usr/lib/hadoop/.//hadoop-kms-3.1.2.jar:/usr/lib/hadoop/.//hadoop-kms.jar:/usr/lib/hadoop/.//hadoop-nfs-3.1.2.
jar:/usr/lib/hadoop/.//hadoop-nfs.jar:/usr/lib/hadoop-hdfs/./:/usr/lib/hadoop-hdfs/lib/kerb-client-1.0.1.jar:/usr/lib/hadoop-
hdfs/lib/accessors-smart-1.2.jar:/usr/lib/hadoop-hdfs/lib/stax2-api-3.1.4.jar:/usr/lib/hadoop-hdfs/lib/asm-5.0.4.
jar:/usr/lib/hadoop-hdfs/lib/jetty-util-ajax-9.3.24.v20180605.jar:/usr/lib/hadoop-hdfs/lib/audience-annotations-0.5.0.
jar:/usr/lib/hadoop-hdfs/lib/token-provider-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/avro-1.7.7.jar:/usr/lib/hadoop-hdfs/lib/kerb-
common-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/commons-beanutils-1.9.3.jar:/usr/lib/hadoop-hdfs/lib/woodstox-core-5.0.3.
jar:/usr/lib/hadoop-hdfs/lib/commons-cli-1.2.jar:/usr/lib/hadoop-hdfs/lib/kerb-core-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/commons-
codec-1.11.jar:/usr/lib/hadoop-hdfs/lib/jetty-io-9.3.24.v20180605.jar:/usr/lib/hadoop-hdfs/lib/commons-collections-3.2.2.
jar:/usr/lib/hadoop-hdfs/lib/kerb-crypto-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/commons-compress-1.18.jar:/usr/lib/hadoop-
hdfs/lib/jetty-xml-9.3.24.v20180605.jar:/usr/lib/hadoop-hdfs/lib/commons-configuration2-2.1.1.jar:/usr/lib/hadoop-
hdfs/lib/kerb-identity-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/commons-daemon-1.0.13.jar:/usr/lib/hadoop-hdfs/lib/zookeeper-3.4.13.
jar:/usr/lib/hadoop-hdfs/lib/commons-io-2.5.jar:/usr/lib/hadoop-hdfs/lib/kerb-server-1.0.1.jar:/usr/lib/hadoop-
hdfs/lib/commons-lang-2.6.jar:/usr/lib/hadoop-hdfs/lib/kerb-simplekdc-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/commons-lang3-3.4.
jar:/usr/lib/hadoop-hdfs/lib/kerb-util-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/commons-logging-1.1.3.jar:/usr/lib/hadoop-
hdfs/lib/kerby-asn1-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/commons-math3-3.1.1.jar:/usr/lib/hadoop-hdfs/lib/commons-net-3.6.
jar:/usr/lib/hadoop-hdfs/lib/kerby-config-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/curator-client-2.13.0.jar:/usr/lib/hadoop-
hdfs/lib/jetty-webapp-9.3.24.v20180605.jar:/usr/lib/hadoop-hdfs/lib/curator-framework-2.13.0.jar:/usr/lib/hadoop-
hdfs/lib/kerby-pkix-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/curator-recipes-2.13.0.jar:/usr/lib/hadoop-hdfs/lib/gson-2.2.4.
jar:/usr/lib/hadoop-hdfs/lib/guava-11.0.2.jar:/usr/lib/hadoop-hdfs/lib/jsch-0.1.54.jar:/usr/lib/hadoop-hdfs/lib/htrace-core4-4.
1.0-incubating.jar:/usr/lib/hadoop-hdfs/lib/kerby-util-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/httpclient-4.5.2.jar:/usr/lib/hadoop-
hdfs/lib/httpcore-4.4.4.jar:/usr/lib/hadoop-hdfs/lib/json-simple-1.1.1.jar:/usr/lib/hadoop-hdfs/lib/jackson-annotations-2.7.8.
jar:/usr/lib/hadoop-hdfs/lib/kerby-xdr-1.0.1.jar:/usr/lib/hadoop-hdfs/lib/jackson-core-2.7.8.jar:/usr/lib/hadoop-
hdfs/lib/leveldbjni-all-1.8.jar:/usr/lib/hadoop-hdfs/lib/jackson-core-asl-1.9.13.jar:/usr/lib/hadoop-hdfs/lib/log4j-1.2.17.
jar:/usr/lib/hadoop-hdfs/lib/jackson-databind-2.7.8.jar:/usr/lib/hadoop-hdfs/lib/netty-3.10.5.Final.jar:/usr/lib/hadoop-
hdfs/lib/jackson-jaxrs-1.9.13.jar:/usr/lib/hadoop-hdfs/lib/json-smart-2.3.jar:/usr/lib/hadoop-hdfs/lib/jackson-mapper-asl-1.9.
13.jar:/usr/lib/hadoop-hdfs/lib/netty-all-4.0.52.Final.jar:/usr/lib/hadoop-hdfs/lib/jackson-xc-1.9.13.jar:/usr/lib/hadoop-
hdfs/lib/nimbus-jose-jwt-4.41.1.jar:/usr/lib/hadoop-hdfs/lib/javax.servlet-api-3.1.0.jar:/usr/lib/hadoop-hdfs/lib/jaxb-api-2.2.
11.jar:/usr/lib/hadoop-hdfs/lib/okhttp-2.7.5.jar:/usr/lib/hadoop-hdfs/lib/jaxb-impl-2.2.3-1.jar:/usr/lib/hadoop-hdfs/lib/okio-1.
6.0.jar:/usr/lib/hadoop-hdfs/lib/jcip-annotations-1.0-1.jar:/usr/lib/hadoop-hdfs/lib/paranamer-2.3.jar:/usr/lib/hadoop-
hdfs/lib/jersey-core-1.19.jar:/usr/lib/hadoop-hdfs/lib/protobuf-java-2.5.0.jar:/usr/lib/hadoop-hdfs/lib/jersey-json-1.19.
jar:/usr/lib/hadoop-hdfs/lib/re2j-1.1.jar:/usr/lib/hadoop-hdfs/lib/jersey-server-1.19.jar:/usr/lib/hadoop-hdfs/lib/snappy-java-
1.0.5.jar:/usr/lib/hadoop-hdfs/lib/jersey-servlet-1.19.jar:/usr/lib/hadoop-hdfs/lib/jettison-1.1.jar:/usr/lib/hadoop-
hdfs/lib/jsr305-3.0.0.jar:/usr/lib/hadoop-hdfs/lib/jetty-http-9.3.24.v20180605.jar:/usr/lib/hadoop-hdfs/lib/jetty-security-9.3.
24.v20180605.jar:/usr/lib/hadoop-hdfs/lib/jetty-server-9.3.24.v20180605.jar:/usr/lib/hadoop-hdfs/lib/jsr311-api-1.1.1.
jar:/usr/lib/hadoop-hdfs/lib/jetty-servlet-9.3.24.v20180605.jar:/usr/lib/hadoop-hdfs/lib/kerb-admin-1.0.1.jar:/usr/lib/hadoop-
hdfs/lib/jetty-util-9.3.24.v20180605.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-3.1.2-tests.jar:/usr/lib/hadoop-hdfs/.//hadoop-
hdfs-3.1.2.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-client-3.1.2-tests.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-client-3.1.2.
jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-client.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-httpfs-3.1.2.jar:/usr/lib/hadoop-
hdfs/.//hadoop-hdfs-httpfs.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-native-client-3.1.2-tests.jar:/usr/lib/hadoop-
hdfs/.//hadoop-hdfs-native-client-3.1.2.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-native-client.jar:/usr/lib/hadoop-
hdfs/.//hadoop-hdfs-nfs-3.1.2.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-nfs.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-rbf-3.1.2-
tests.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-rbf-3.1.2.jar:/usr/lib/hadoop-hdfs/.//hadoop-hdfs-rbf.jar:/usr/lib/hadoop-
hdfs/.//hadoop-hdfs.jar:/usr/lib/hadoop-mapreduce/.//hadoop-streaming.jar:/usr/lib/hadoop-mapreduce/.//aliyun-sdk-oss-2.8.3.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-examples.jar:/usr/lib/hadoop-mapreduce/.//aws-java-sdk-bundle-1.11.271.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-openstack-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//azure-data-lake-store-sdk-2.2.7.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-openstack.jar:/usr/lib/hadoop-mapreduce/.//azure-keyvault-core-1.0.0.
jar:/usr/lib/hadoop-mapreduce/.//jdom-1.1.jar:/usr/lib/hadoop-mapreduce/.//azure-storage-7.0.0.jar:/usr/lib/hadoop-
mapreduce/.//kafka-clients-0.8.2.1.jar:/usr/lib/hadoop-mapreduce/.//hadoop-aliyun-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-
aliyun.jar:/usr/lib/hadoop-mapreduce/.//hadoop-resourceestimator-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-archive-logs-3.1.
2.jar:/usr/lib/hadoop-mapreduce/.//lz4-1.2.0.jar:/usr/lib/hadoop-mapreduce/.//hadoop-archive-logs.jar:/usr/lib/hadoop-
mapreduce/.//netty-buffer-4.1.17.Final.jar:/usr/lib/hadoop-mapreduce/.//hadoop-archives-3.1.2.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-archives.jar:/usr/lib/hadoop-mapreduce/.//netty-codec-4.1.17.Final.jar:/usr/lib/hadoop-mapreduce/.//hadoop-
aws-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-aws.jar:/usr/lib/hadoop-mapreduce/.//netty-codec-http-4.1.17.Final.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-azure-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-resourceestimator.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-azure-datalake-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//netty-common-4.1.17.Final.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-azure-datalake.jar:/usr/lib/hadoop-mapreduce/.//hadoop-azure.jar:/usr/lib/hadoop-
mapreduce/.//netty-handler-4.1.17.Final.jar:/usr/lib/hadoop-mapreduce/.//hadoop-datajoin-3.1.2.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-datajoin.jar:/usr/lib/hadoop-mapreduce/.//netty-resolver-4.1.17.Final.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-distcp-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-distcp.jar:/usr/lib/hadoop-mapreduce/.//netty-
transport-4.1.17.Final.jar:/usr/lib/hadoop-mapreduce/.//hadoop-extras-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-extras.
jar:/usr/lib/hadoop-mapreduce/.//ojalgo-43.0.jar:/usr/lib/hadoop-mapreduce/.//hadoop-fs2img-3.1.2.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-fs2img.jar:/usr/lib/hadoop-mapreduce/.//hadoop-gridmix-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-
gridmix.jar:/usr/lib/hadoop-mapreduce/.//hadoop-kafka-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-kafka.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-mapreduce-client-jobclient-3.1.2-tests.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-app-3.1.2.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-rumen-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-app.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-jobclient-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-
client-common-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-common.jar:/usr/lib/hadoop-mapreduce/.//hadoop-
mapreduce-client-jobclient.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-core-3.1.2.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-sls-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-core.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-mapreduce-client-nativetask-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-hs-3.1.2.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-uploader-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-
client-hs-plugins-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-uploader.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-mapreduce-client-hs-plugins.jar:/usr/lib/hadoop-mapreduce/.//hadoop-sls.jar:/usr/lib/hadoop-
mapreduce/.//hadoop-mapreduce-client-hs.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-nativetask.
jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-examples-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-
shuffle-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-streaming-3.1.2.jar:/usr/lib/hadoop-mapreduce/.//hadoop-mapreduce-client-
shuffle.jar:/usr/lib/hadoop-mapreduce/.//hadoop-rumen.jar:/usr/lib/hadoop-yarn/./:/usr/lib/hadoop-yarn/lib/HikariCP-java7-2.4.
12.jar:/usr/lib/hadoop-yarn/lib/aopalliance-1.0.jar:/usr/lib/hadoop-yarn/lib/dnsjava-2.1.7.jar:/usr/lib/hadoop-
yarn/lib/ehcache-3.3.1.jar:/usr/lib/hadoop-yarn/lib/fst-2.50.jar:/usr/lib/hadoop-yarn/lib/geronimo-jcache_1.0_spec-1.0-alpha-1.
jar:/usr/lib/hadoop-yarn/lib/guice-4.0.jar:/usr/lib/hadoop-yarn/lib/guice-servlet-4.0.jar:/usr/lib/hadoop-yarn/lib/jackson-
jaxrs-base-2.7.8.jar:/usr/lib/hadoop-yarn/lib/jackson-jaxrs-json-provider-2.7.8.jar:/usr/lib/hadoop-yarn/lib/jackson-module-
jaxb-annotations-2.7.8.jar:/usr/lib/hadoop-yarn/lib/java-util-1.9.0.jar:/usr/lib/hadoop-yarn/lib/javax.inject-1.
jar:/usr/lib/hadoop-yarn/lib/jersey-client-1.19.jar:/usr/lib/hadoop-yarn/lib/jersey-guice-1.19.jar:/usr/lib/hadoop-
yarn/lib/json-io-2.5.1.jar:/usr/lib/hadoop-yarn/lib/metrics-core-3.2.4.jar:/usr/lib/hadoop-yarn/lib/mssql-jdbc-6.2.1.jre7.
jar:/usr/lib/hadoop-yarn/lib/objenesis-1.0.jar:/usr/lib/hadoop-yarn/lib/snakeyaml-1.16.jar:/usr/lib/hadoop-yarn/lib/swagger-
annotations-1.5.4.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-api-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-api.
jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-services-core.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-applications-distributedshell-3.
1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-applications-distributedshell.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-applications-
unmanaged-am-launcher-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-applications-unmanaged-am-launcher.jar:/usr/lib/hadoop-
yarn/.//hadoop-yarn-client-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-client.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-common-
3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-common.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-registry-3.1.2.jar:/usr/lib/hadoop-
yarn/.//hadoop-yarn-registry.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-services-core-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-
yarn-server-applicationhistoryservice-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-applicationhistoryservice.
jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-common-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-common.
jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-nodemanager-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-nodemanager.
jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-resourcemanager-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-
resourcemanager.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-router-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-
router.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-sharedcachemanager-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-
sharedcachemanager.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-tests-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-
tests.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-timeline-pluginstorage-3.1.2.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-
server-timeline-pluginstorage.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-server-web-proxy-3.1.2.jar:/usr/lib/hadoop-
yarn/.//hadoop-yarn-server-web-proxy.jar:/usr/lib/hadoop-yarn/.//hadoop-yarn-services-api-3.1.2.jar:/usr/lib/hadoop-
yarn/.//hadoop-yarn-services-api.jar
2021-12-02 07:18:15,710 INFO zookeeper.ZooKeeper: Client environment:java.library.path=/usr/lib/hadoop/lib/native
2021-12-02 07:18:15,710 INFO zookeeper.ZooKeeper: Client environment:java.io.tmpdir=/tmp
2021-12-02 07:18:15,710 INFO zookeeper.ZooKeeper: Client environment:java.compiler=<NA>
2021-12-02 07:18:15,711 INFO zookeeper.ZooKeeper: Client environment:os.name=Linux
2021-12-02 07:18:15,711 INFO zookeeper.ZooKeeper: Client environment:os.arch=amd64
2021-12-02 07:18:15,711 INFO zookeeper.ZooKeeper: Client environment:os.version=3.10.0-1160.11.1.el7.x86_64
2021-12-02 07:18:15,711 INFO zookeeper.ZooKeeper: Client environment:user.name=dasha
2021-12-02 07:18:15,711 INFO zookeeper.ZooKeeper: Client environment:user.home=/home/dasha
2021-12-02 07:18:15,711 INFO zookeeper.ZooKeeper: Client environment:user.dir=/usr/lib/phoenix/bin
2021-12-02 07:18:15,714 INFO zookeeper.ZooKeeper: Initiating client connection, connectString=localhost:2181
sessionTimeout=90000 watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$13/1224047103@df7fe85
2021-12-02 07:18:15,736 INFO zookeeper.ClientCnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not
attempt to authenticate using SASL (unknown error)
2021-12-02 07:18:15,741 INFO zookeeper.ClientCnxn: Socket connection established to localhost/127.0.0.1:2181, initiating
session
2021-12-02 07:18:15,750 INFO zookeeper.ClientCnxn: Session establishment complete on server localhost/127.0.0.1:2181, sessionid
= 0x100000335ca0024, negotiated timeout = 40000
2021-12-02 07:18:15,915 INFO query.ConnectionQueryServicesImpl: HConnection established. Stacktrace for informational purposes:
hconnection-0xe487ec java.lang.Thread.getStackTrace(Thread.java:1559)
org.apache.phoenix.util.LogUtil.getCallerStackTrace(LogUtil.java:55)
org.apache.phoenix.query.ConnectionQueryServicesImpl.openConnection(ConnectionQueryServicesImpl.java:432)
org.apache.phoenix.query.ConnectionQueryServicesImpl.access$400(ConnectionQueryServicesImpl.java:272)
org.apache.phoenix.query.ConnectionQueryServicesImpl$12.call(ConnectionQueryServicesImpl.java:2556)
org.apache.phoenix.query.ConnectionQueryServicesImpl$12.call(ConnectionQueryServicesImpl.java:2532)
org.apache.phoenix.util.PhoenixContextExecutor.call(PhoenixContextExecutor.java:76)
org.apache.phoenix.query.ConnectionQueryServicesImpl.init(ConnectionQueryServicesImpl.java:2532)
org.apache.phoenix.jdbc.PhoenixDriver.getConnectionQueryServices(PhoenixDriver.java:255)
org.apache.phoenix.jdbc.PhoenixEmbeddedDriver.createConnection(PhoenixEmbeddedDriver.java:150)
org.apache.phoenix.jdbc.PhoenixDriver.connect(PhoenixDriver.java:221)
java.sql.DriverManager.getConnection(DriverManager.java:664)
java.sql.DriverManager.getConnection(DriverManager.java:208)
org.apache.phoenix.util.QueryUtil.getConnection(QueryUtil.java:400)
org.apache.phoenix.util.QueryUtil.getConnection(QueryUtil.java:392)
org.apache.phoenix.mapreduce.AbstractBulkLoadTool.loadData(AbstractBulkLoadTool.java:206)
org.apache.phoenix.mapreduce.AbstractBulkLoadTool.run(AbstractBulkLoadTool.java:180)
org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:76)
org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:90)
org.apache.phoenix.mapreduce.CsvBulkLoadTool.main(CsvBulkLoadTool.java:109)
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
java.lang.reflect.Method.invoke(Method.java:498)
org.apache.hadoop.util.RunJar.run(RunJar.java:318)
org.apache.hadoop.util.RunJar.main(RunJar.java:232)

2021-12-02 07:18:18,132 INFO zookeeper.ReadOnlyZKClient: Connect 0x4ecebaad to localhost:2181 with session timeout=90000ms,
retries 30, retry interval 1000ms, keepAlive=60000ms
2021-12-02 07:18:18,133 INFO zookeeper.ZooKeeper: Initiating client connection, connectString=localhost:2181
sessionTimeout=90000 watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$13/1224047103@df7fe85
2021-12-02 07:18:18,135 INFO zookeeper.ClientCnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not
attempt to authenticate using SASL (unknown error)
2021-12-02 07:18:18,135 INFO zookeeper.ClientCnxn: Socket connection established to localhost/127.0.0.1:2181, initiating
session
2021-12-02 07:18:18,139 INFO zookeeper.ClientCnxn: Session establishment complete on server localhost/127.0.0.1:2181, sessionid
= 0x100000335ca0025, negotiated timeout = 40000
2021-12-02 07:18:18,145 INFO zookeeper.ReadOnlyZKClient: Connect 0x0aecb893 to localhost:2181 with session timeout=90000ms,
retries 30, retry interval 1000ms, keepAlive=60000ms
2021-12-02 07:18:18,146 INFO zookeeper.ZooKeeper: Initiating client connection, connectString=localhost:2181
sessionTimeout=90000 watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$13/1224047103@df7fe85
2021-12-02 07:18:18,147 INFO zookeeper.ClientCnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not
attempt to authenticate using SASL (unknown error)
2021-12-02 07:18:18,147 INFO zookeeper.ClientCnxn: Socket connection established to localhost/127.0.0.1:2181, initiating
session
2021-12-02 07:18:18,150 INFO zookeeper.ClientCnxn: Session establishment complete on server localhost/127.0.0.1:2181, sessionid
= 0x100000335ca0026, negotiated timeout = 40000
2021-12-02 07:18:18,362 INFO mapreduce.MultiHfileOutputFormat:  the table logical name is BEST_BOOKS
2021-12-02 07:18:18,362 INFO client.ConnectionImplementation: Closing master protocol: MasterService
2021-12-02 07:18:18,363 INFO zookeeper.ReadOnlyZKClient: Close zookeeper connection 0x0aecb893 to localhost:2181
2021-12-02 07:18:18,365 INFO mapreduce.MultiHfileOutputFormat: Configuring 1 reduce partitions to match current region count
2021-12-02 07:18:18,365 INFO mapreduce.MultiHfileOutputFormat: Writing partition information to /tmp/hadoop-
dasha/partitions_dfe383e6-d920-4829-8340-a86b2926db27
2021-12-02 07:18:18,366 INFO zookeeper.ZooKeeper: Session: 0x100000335ca0026 closed
2021-12-02 07:18:18,367 INFO zookeeper.ClientCnxn: EventThread shut down for session: 0x100000335ca0026
2021-12-02 07:18:18,489 INFO zlib.ZlibFactory: Successfully loaded & initialized native-zlib library
2021-12-02 07:18:18,490 INFO compress.CodecPool: Got brand-new compressor [.deflate]
2021-12-02 07:18:18,643 INFO mapreduce.AbstractBulkLoadTool: Running MapReduce import job from /user/dasha/best_books1.csv to
/tmp/2533c78e-2a96-47f4-9e06-aaec783f979f
2021-12-02 07:18:18,836 INFO client.RMProxy: Connecting to ResourceManager at bds-adh-2.ru-central1.internal/10.92.6.9:8032
2021-12-02 07:18:18,965 INFO client.AHSProxy: Connecting to Application History server at bds-adh-3.ru-central1.internal/10.92.
6.90:10200
2021-12-02 07:18:19,083 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/dasha/.
staging/job_1638426545038_0003
2021-12-02 07:18:20,061 INFO input.FileInputFormat: Total input files to process : 1
2021-12-02 07:18:20,159 INFO mapreduce.JobSubmitter: number of splits:1
2021-12-02 07:18:20,198 INFO Configuration.deprecation: io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-
checksum
2021-12-02 07:18:20,291 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1638426545038_0003
2021-12-02 07:18:20,291 INFO mapreduce.JobSubmitter: Executing with tokens: []
2021-12-02 07:18:20,445 INFO conf.Configuration: resource-types.xml not found
2021-12-02 07:18:20,445 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2021-12-02 07:18:20,506 INFO impl.YarnClientImpl: Submitted application application_1638426545038_0003
2021-12-02 07:18:20,642 INFO mapreduce.Job: The url to track the job: http://bds-adh-2.ru-central1.
internal:8088/proxy/application_1638426545038_0003/
2021-12-02 07:18:20,642 INFO mapreduce.Job: Running job: job_1638426545038_0003
2021-12-02 07:18:29,739 INFO mapreduce.Job: Job job_1638426545038_0003 running in uber mode : false
2021-12-02 07:18:29,741 INFO mapreduce.Job:  map 0% reduce 0%
2021-12-02 07:18:37,797 INFO mapreduce.Job:  map 100% reduce 0%
2021-12-02 07:18:46,833 INFO mapreduce.Job:  map 100% reduce 100%
2021-12-02 07:18:46,838 INFO mapreduce.Job: Job job_1638426545038_0003 completed successfully
2021-12-02 07:18:46,920 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=5824
                FILE: Number of bytes written=534283
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=2839
                HDFS: Number of bytes written=8614
                HDFS: Number of read operations=14
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=5
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=6406
                Total time spent by all reduces in occupied slots (ms)=6027
                Total time spent by all map tasks (ms)=6406
                Total time spent by all reduce tasks (ms)=6027
                Total vcore-milliseconds taken by all map tasks=6406
                Total vcore-milliseconds taken by all reduce tasks=6027
                Total megabyte-milliseconds taken by all map tasks=6559744
                Total megabyte-milliseconds taken by all reduce tasks=6171648
        Map-Reduce Framework
                Map input records=70
                Map output records=70
                Map output bytes=5678
                Map output materialized bytes=5824
                Input split bytes=103
                Combine input records=0
                Combine output records=0
                Reduce input groups=70
                Reduce shuffle bytes=5824
                Reduce input records=70
                Reduce output records=140
                Spilled Records=140
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=230
                CPU time spent (ms)=8510
                Physical memory (bytes) snapshot=1084018688
                Virtual memory (bytes) snapshot=5900730368
                Total committed heap usage (bytes)=815267840
                Peak Map Physical memory (bytes)=597655552
                Peak Map Virtual memory (bytes)=2946670592
                Peak Reduce Physical memory (bytes)=486363136
                Peak Reduce Virtual memory (bytes)=2954059776
        Phoenix MapReduce Import
                Upserts Done=70
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=2736
        File Output Format Counters
                Bytes Written=8614
2021-12-02 07:18:46,922 INFO mapreduce.AbstractBulkLoadTool: Loading HFiles from /tmp/2533c78e-2a96-47f4-9e06-aaec783f979f
2021-12-02 07:18:46,950 INFO Configuration.deprecation: io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-
checksum
2021-12-02 07:18:46,953 INFO zookeeper.ReadOnlyZKClient: Connect 0x59a2153e to localhost:2181 with session timeout=90000ms,
retries 30, retry interval 1000ms, keepAlive=60000ms
2021-12-02 07:18:46,953 INFO zookeeper.ZooKeeper: Initiating client connection, connectString=localhost:2181
sessionTimeout=90000 watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$13/1224047103@df7fe85
2021-12-02 07:18:46,955 INFO zookeeper.ClientCnxn: Opening socket connection to server localhost/127.0.0.1:2181. Will not
attempt to authenticate using SASL (unknown error)
2021-12-02 07:18:46,955 INFO zookeeper.ClientCnxn: Socket connection established to localhost/127.0.0.1:2181, initiating
session
2021-12-02 07:18:46,958 INFO zookeeper.ClientCnxn: Session establishment complete on server localhost/127.0.0.1:2181, sessionid
= 0x100000335ca0027, negotiated timeout = 40000
2021-12-02 07:18:46,959 INFO mapreduce.AbstractBulkLoadTool: Loading HFiles for BEST_BOOKS from /tmp/2533c78e-2a96-47f4-9e06-
aaec783f979f/BEST_BOOKS
2021-12-02 07:18:47,126 INFO impl.MetricsConfig: loaded properties from hadoop-metrics2-hbase.properties
2021-12-02 07:18:47,132 WARN impl.MetricsSystemImpl: Error creating sink 'tracing'
org.apache.hadoop.metrics2.impl.MetricsConfigException: Error creating plugin: org.apache.phoenix.trace.PhoenixMetricsSink
        at org.apache.hadoop.metrics2.impl.MetricsConfig.getPlugin(MetricsConfig.java:211)
        at org.apache.hadoop.metrics2.impl.MetricsSystemImpl.newSink(MetricsSystemImpl.java:531)
        at org.apache.hadoop.metrics2.impl.MetricsSystemImpl.configureSinks(MetricsSystemImpl.java:503)
        at org.apache.hadoop.metrics2.impl.MetricsSystemImpl.configure(MetricsSystemImpl.java:479)
        at org.apache.hadoop.metrics2.impl.MetricsSystemImpl.start(MetricsSystemImpl.java:188)
        at org.apache.hadoop.metrics2.impl.MetricsSystemImpl.init(MetricsSystemImpl.java:163)
        at org.apache.hadoop.metrics2.lib.DefaultMetricsSystem.init(DefaultMetricsSystem.java:62)
        at org.apache.hadoop.metrics2.lib.DefaultMetricsSystem.initialize(DefaultMetricsSystem.java:58)
        at org.apache.hadoop.hbase.metrics.BaseSourceImpl$DefaultMetricsSystemInitializer.init(BaseSourceImpl.java:54)
        at org.apache.hadoop.hbase.metrics.BaseSourceImpl.<init>(BaseSourceImpl.java:116)
        at org.apache.hadoop.hbase.io.MetricsIOSourceImpl.<init>(MetricsIOSourceImpl.java:46)
        at org.apache.hadoop.hbase.io.MetricsIOSourceImpl.<init>(MetricsIOSourceImpl.java:38)
        at org.apache.hadoop.hbase.regionserver.MetricsRegionServerSourceFactoryImpl.
createIO(MetricsRegionServerSourceFactoryImpl.java:84)
        at org.apache.hadoop.hbase.io.MetricsIO.<init>(MetricsIO.java:35)
        at org.apache.hadoop.hbase.io.hfile.HFile.<clinit>(HFile.java:195)
        at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.visitBulkHFiles(LoadIncrementalHFiles.java:1025)
        at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.discoverLoadQueue(LoadIncrementalHFiles.java:941)
        at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.prepareHFileQueue(LoadIncrementalHFiles.java:224)
        at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.doBulkLoad(LoadIncrementalHFiles.java:331)
        at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.doBulkLoad(LoadIncrementalHFiles.java:256)
        at org.apache.phoenix.mapreduce.AbstractBulkLoadTool.completebulkload(AbstractBulkLoadTool.java:355)
        at org.apache.phoenix.mapreduce.AbstractBulkLoadTool.submitJob(AbstractBulkLoadTool.java:331)
        at org.apache.phoenix.mapreduce.AbstractBulkLoadTool.loadData(AbstractBulkLoadTool.java:267)
        at org.apache.phoenix.mapreduce.AbstractBulkLoadTool.run(AbstractBulkLoadTool.java:180)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:76)
        at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:90)
        at org.apache.phoenix.mapreduce.CsvBulkLoadTool.main(CsvBulkLoadTool.java:109)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at org.apache.hadoop.util.RunJar.run(RunJar.java:318)
        at org.apache.hadoop.util.RunJar.main(RunJar.java:232)
Caused by: java.lang.ClassNotFoundException: org.apache.phoenix.trace.PhoenixMetricsSink
        at java.net.URLClassLoader.findClass(URLClassLoader.java:387)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:418)
        at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:352)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:351)
        at java.lang.Class.forName0(Native Method)
        at java.lang.Class.forName(Class.java:348)
        at org.apache.hadoop.metrics2.impl.MetricsConfig.getPlugin(MetricsConfig.java:205)
        ... 32 more
2021-12-02 07:18:47,203 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2021-12-02 07:18:47,203 INFO impl.MetricsSystemImpl: HBase metrics system started
2021-12-02 07:18:47,220 INFO metrics.MetricRegistries: Loaded MetricRegistries class org.apache.hadoop.hbase.metrics.impl.
MetricRegistriesImpl
2021-12-02 07:18:47,346 INFO hfile.CacheConfig: Created cacheConfig: CacheConfig:disabled
2021-12-02 07:18:47,412 INFO tool.LoadIncrementalHFiles: Trying to load hfile=hdfs://adh1/tmp/2533c78e-2a96-47f4-9e06-
aaec783f979f/BEST_BOOKS/0/f544e36fd2044f07bce46e920d6c4c60 first=Optional[A.S. Byatt\x00Possession] last=Optional[Zora Neale
Hurston\x00Their Eyes Were Watching God]
2021-12-02 07:18:47,526 INFO mapreduce.AbstractBulkLoadTool: Incremental load complete for table=BEST_BOOKS
2021-12-02 07:18:47,526 INFO mapreduce.AbstractBulkLoadTool: Removing output directory /tmp/2533c78e-2a96-47f4-9e06-
aaec783f979f
```

7. Проверьте результат выполнения команды путем выполнения следующего запроса в среде SQLLine для отображения первых десяти строк из созданной таблицы:

```
SELECT * FROM BEST_BOOKS LIMIT 10;
```

   Вывод на экран:

```
+-----------------------+-----------------------------------------------------+--------------+
|        AUTHOR         |                        TITLE                        | PUBLIC_YEAR  |
+-----------------------+-----------------------------------------------------+--------------+
| A.S. Byatt            | Possession                                          | 1990         |
| Alan Moore            | Watchmen                                            | 1987         |
| Anthony Burgess       | A Clockwork Orange                                  | 1962         |
| C.S. Lewis            | The Lion, the Witch and the Wardrobe                | 1950         |
| Carson McCullers      | The Heart Is a Lonely Hunter                        | 1940         |
| Chinua Achebe         | Chinua Achebe                                       | 1958         |
| Cormac McCarthy       | Blood Meridian, or the Evening Redness in the West  | 1985         |
| Dashiell Hammett      | Red Harvest                                         | 1929         |
| David Foster Wallace  | Infinite Jest                                       | 1996         |
| Don DeLillo           | White Noise                                         | 1985         |
+-----------------------+-----------------------------------------------------+--------------+
10 rows selected (0.092 seconds)
```

   Определите количество строк в таблице:

```
SELECT COUNT(*) FROM BEST_BOOKS;
```

   Вывод на экран:

```
+-----------+
| COUNT(1)  |
+-----------+
| 70        |
+-----------+
1 row selected (0.048 seconds)
```

8. В заключение выполните команду `list_regions` в HBase shell. Обратите внимание на значение `1` в колонке `LOCALITY`. Оно означает, что второй метод загружает данные напрямую в HFiles.

```
list_regions 'BEST_BOOKS'
```

   Вывод на экран:

```
                                 SERVER_NAME |                                                  REGION_NAME |  START_KEY |
END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | ------------------------------------------------------------ | ---------- |
---------- | ----- | ----- | ---------- |
 bds-adh-1.ru-central1.internal,16020,1638426590054 | BEST_BOOKS,,1638427992879.8a84103a2b4ed50e718587bcc8a1e63b. |            |
|     0 |    81 |        1.0 |
 1 rows
Took 0.6029 seconds
```

Оптимизация производительности в Phoenix

Владимир Адаменков

Содержание

Назад к содержанию

- Настройка конфигурации
  - Соединение
  - Запросы
  - JDBC URL
  - Серверы регионов
- Настройка БД
  - Ключ строки
  - Семейства столбцов
  - Стратегии индексирования
  - Сжатие данных
  - Локальность данных
- Методы оптимизации запросов
  - План выполнения запроса
  - Эффективность SQL
  - Оптимизация JOIN
  - Подсказки
  - Сервер запросов

В этой статье приведены рекомендации по оптимизации производительности Phoenix в ADH.

Настройка конфигурации

Соединение

Описанные ниже параметры управляют настройкой подключений в Phoenix.

Параметры подключений

Параметр	Значение по умолчанию и описание	Рекомендация
phoenix.query.timeoutMs	Значение по умолчанию: `600000` (в миллисекундах). Определяет время, в течение которого запрос может выполняться, прежде чем он будет прерван на стороне клиента. Это предотвращает чрезмерное потребление ресурсов длительными или неконтролируемыми запросами и потенциальное снижение производительности	Установите относительно небольшое значение (`30000` или `60000`) и отслеживайте фактическое время выполнения запросов. Используйте для этого веб-интерфейс HBase, команду Phoenix `EXPLAIN PLAN` или логирование на уровне приложения. Определите запросы, которые постоянно превышают текущее значение тайм-аута. Если обнаружится, что тайм-аут слишком мал, и запросы, которые должны быть выполнены успешно, постоянно завершаются с ошибкой, постепенно увеличивайте значение на небольшую величину (например, на 30 секунд или на 1 минуту) и отслеживайте результаты. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
phoenix.query.threadPoolSize	Значение по умолчанию: `128`. Этот параметр определяет максимальное количество потоков в пуле, выделенных для выполнения входящих запросов Phoenix. Каждый поток обрабатывает один запрос (или часть сложного запроса)	Начните со значения по умолчанию и запустите типичную рабочую нагрузку. Отслеживайте ключевые метрики: - использование ЦП; - использование ОЗУ; - задержки запросов; - метрики пула потоков. Если процессоры загружены недостаточно, задержка запросов высока или глубина очереди запросов постоянно велика, увеличьте параметр на 25%. Если процессоры загружены на максимум, а оперативная память исчерпана, уменьшите параметр на 25%. После каждой корректировки перезапускайте рабочую нагрузку и контролируйте метрики. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
hbase.client.scanner.timeout.period	Значение по умолчанию: `60000` (в миллисекундах). Этот параметр определяет тайм-аут бездействия сканера, по истечении которого сервер региона закроет его	Начните со значения по умолчанию и запустите типичную рабочую нагрузку. Проверьте логи на наличие исключений типа `scanner timed out` и/или прерываний длительных запросов. Если таких случаев много, увеличьте значение параметра. Перезапустите рабочую нагрузку и проверьте логи еще раз. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase
zookeeper.session.timeout	Значение по умолчанию: `90000` (в миллисекундах). Этот параметр определяет тайм-аут для сеансов ZooKeeper. HBase (и, следовательно, Phoenix) использует ZooKeeper для координации. Потеря сеанса ZooKeeper сервером региона может привести к проблемам с подключением и нестабильной работе	В средах с нестабильными сетями или высокой нагрузкой на ZooKeeper рекомендуется увеличить это значение. Однако чем больше значение, тем дольше будет определяться действительно неисправный сервер региона. Не устанавливайте слишком низкое значение, так как временные сбои в сети могут привести к ненужным отключениям. Этот параметр необходимо согласовывать с конфигурацией ZooKeeper. Этот параметр находится в секции ***hbase-site.xml*** на конфигурационной странице сервиса HBase

Запросы

Описанные ниже параметры связаны с производительностью запросов в Phoenix.

Параметры запросов

Параметр	Значение по умолчанию и описание	Рекомендация
phoenix.query.spoolThresholdBytes	Значение по умолчанию: `20971520` (в байтах). Ограничивает объем памяти, который запрос может использовать для сортировки или группировки результатов, не прибегая к переброску памяти на диск для временного хранения. Это критически важный параметр для запросов, использующих предложения `ORDER BY`, `GROUP BY` или `JOIN`, особенно при работе с большими наборами данных. Недостаток памяти может привести к чрезмерному количеству операций ввода-вывода на диске, что существенно скажется на производительности запросов	Установите для этого параметра значение `10485760` (10 МБ) и запустите типичную рабочую нагрузку. Контролируйте ключевые метрики: - Задержки запросов — измерьте среднюю задержку запроса и 95-й/99-й процентиль. - Использование операций ввода-вывода на диске. - Активность сборщика мусора. Если какая-либо из этих метрик слишком высокая — удвойте значение параметра, перезапустите рабочую нагрузку и снова отслеживайте метрики. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
phoenix.query.maxRows	Значение по умолчанию: `-1` (без ограничения). Этот параметр ограничивает максимальное количество строк, которое может вернуть один запрос. Если запрос попытается вернуть больше строк, чем это ограничение, Phoenix выбросит исключение (`org.apache.phoenix.exception.PhoenixIOException`) и завершит выполнение запроса	Начните со значения по умолчанию и запустите типичную рабочую нагрузку. Отслеживайте выполнение запросов и выявляйте те из них, которые возвращают неожиданно большие наборы результатов или вызывают проблемы с производительностью. Проверьте логи запросов, чтобы понять типичные размеры наборов результатов для различных типов запросов. Определите запросы, которые постоянно возвращают очень большое количество строк. Основываясь на результатах, установите значение параметра, достаточно большое для обработки большинства корректных запросов, но достаточно низкое, чтобы неконтролируемые запросы не перегружали систему. Установите относительно консервативное значение (например, `10000` или `100000`) и перезапустите рабочую нагрузку. Отслеживайте выполнение запросов и выявляйте запросы, которые были прерваны из-за превышения значения параметра `phoenix.query.maxRows`. Если вы обнаружите корректные запросы, которые прерываются, немного увеличьте лимит и перезапустите рабочую нагрузку снова. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
phoenix.query.resultset.cache.enabled	Значение по умолчанию: `false`. Этот параметр определяет, включено ли кеширование результатов запросов на стороне клиента. Если включено, Phoenix будет кешировать результаты запросов, позволяя последующим идентичным запросам возвращать результаты из кеша вместо повторного выполнения на сервере	Включите этот параметр, если у вас часто выполняются запросы с большим объемом чтения и стабильными данными. Это может значительно повысить производительность чтения. Однако это увеличивает потребление клиентской памяти и создает риск устаревания данных, если базовые данные часто меняются, а кеш не аннулируется. Следите за актуальностью клиентской памяти и данных. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
phoenix.query.optimizeMaxParallelScans	Значение по умолчанию: `10`. Этот параметр определяет максимальное количество параллельных сканирований, которые Phoenix может инициировать для определенных типов запросов (например, запросов, которые включают несколько параллельных сканирований по разным регионам или таблицам)	Если ваши таблицы распределены по многим регионам или имеется сильно распределенный набор данных, увеличение этого значения может повысить производительность сложных аналитических запросов, которым может помочь параллелизм. Увеличивайте значение с осторожностью, так как слишком большое значение может создать значительную нагрузку на службы метаданных кластера и на сеть. Следите за активностью серверов регионов и ZooKeeper. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
phoenix.mutate.batchSize	Значение по умолчанию: `1000`. Этот параметр определяет количество операций изменения (`INSERT`, `UPDATE` или `DELETE`), которые накапливаются на стороне клиента перед отправкой на сервер HBase одной партией. Больший размер партии сокращает количество обращений к серверу, что может повысить производительность записи	Увеличьте это значение для повышения пропускной способности записи при выполнении массовой загрузки или массовых обновлений. Начните с удвоения значения по умолчанию и следите за производительностью. Если на стороне клиента или на серверах региона возникает ошибка `OutOfMemoryError` или если сервер регионов перегружен из-за больших пакетов, уменьшите значение. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
phoenix.mutate.maxSizeBytes	Значение по умолчанию: `10485760` (в байтах). Определяет максимальный размер одной партии операций изменения. Это предотвращает перегрузку серверов регионов очень большими партиями	Если вы вставляете строки с очень большими значениями (например, большими двоичными данными), и лимит по умолчанию постоянно превышается, увеличьте это значение. Не устанавливайте слишком большое значение, так как это может привести к проблемам с серверами регионов. При появлении ошибок, связанных с изменениями слишком большого размера, уменьшите это значение. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

ВНИМАНИЕ

Последние два параметра (`phoenix.mutate.batchSize` и `phoenix.mutate.maxSizeBytes`) работают в связке. Партия данных будет отправлена при достижении любого из ограничений.

JDBC URL

Описанные ниже параметры можно использовать в строке JDBC URL при подключении к Phoenix.

Параметры JDBC URL

Параметр	Значение по умолчанию и описание	Рекомендация
phoenix.query.client.fetchSize	Значение по умолчанию: `1000`. Этот параметр определяет количество строк, которые клиент запрашивает у сервера в каждой операции выборки во время выполнения запроса. Больший размер выборки сокращает количество сетевых циклов, потенциально повышая производительность запросов, возвращающих много строк	Увеличьте значение для запросов, извлекающих большое количество строк, чтобы снизить задержку в сети. Уменьшите значение, если памяти на клиенте становится недостаточно или при плохих сетевых условиях, которые могут приводить к тайм-ауту при выборке больших объемов данных. Часто бывает полезно сделать это значение равным значению параметра `phoenix.statement.prefetchRows`, если он используется (описан далее)
phoenix.statement.prefetchRows	Значение по умолчанию: `0` (disabled). Этот параметр задает количество строк, предварительно загружаемых в локальный буфер клиента. Если установлено значение больше `0`, Phoenix попытается загрузить указанное количество строк раньше, чем те, которые в данный момент потребляются приложением, сглаживая выполнение вызовов `next()`	Укажите значение больше `0` (например, `1000` или `5000`) для запросов, в которых клиент перебирает результаты с помощью функции `next()`. Это может значительно улучшить воспринимаемую производительность запроса за счет сокращения задержки между выборками строк. Слишком большое значение может привести к чрезмерному потреблению клиентской памяти
phoenix.schema.isNamespaceMappingEnabled	Значение по умолчанию: `false`. Этот параметр определяет, должен ли Phoenix использовать пространства имен HBase для представления схем Phoenix	Установите этот параметр в значение `true`, если вы хотите использовать пространства имен HBase для управления схемами. Это обеспечивает лучшее разделение задач и может повысить производительность в мультиарендных средах за счет использования безопасности и управления ресурсами на основе пространств имен HBase. Для этого требуется правильная настройка и конфигурация пространств имен HBase. Если пространства имен HBase не используются, оставьте значение `false`
phoenix.use.stats	Значение по умолчанию: `true`. Этот параметр определяет, включено ли использование статистики при планировании запросов	Оставьте это значение равным `true`, чтобы Phoenix мог принимать обоснованные решения о выполнении запросов на основе статистики таблицы. Если у вас очень небольшой набор данных или известно, что статистика значительно устарела, можно временно установить значение `false`, чтобы Phoenix использовал другой план запроса. Однако в большинстве случаев рекомендуется оставить статистику включенной и обеспечить ее регулярное обновление. Вы можете принудительно обновить статистику с помощью запроса `UPDATE STATISTICS <table_name>`
async	Значение по умолчанию: `false`. Этот параметр определяет, включены ли асинхронные запросы. Это позволяет клиенту отправлять запросы и получать результаты позже	Установите этот параметр в значение `true`, если хотите отправлять запросы без блокировки. Результаты можно обрабатывать позже. Для обработки асинхронных результатов требуется дополнительный код и инфраструктура

Серверы регионов

Описанные ниже параметры относятся к работе серверов регионов в контексте работы Phoenix.

Параметры серверов регионов

Параметр	Значение по умолчанию и описание	Рекомендация
hbase.rpc.timeout	Значение по умолчанию: `60000` (в миллисекундах). Этот параметр определяет общее время ожидания для RPC-вызовов к серверам регионов HBase. Если клиент не получает ответ от сервера регионов в течение этого времени, RPC-вызов считается неудавшимся. Phoenix использует RPC-вызовы практически для всех операций	При частом появлении ошибок `RpcTimeoutException` или `CallTimeoutException`, что может происходить при высокой нагрузке на сервер регионов или задержке в сети, увеличьте это значение. Увеличивайте его постепенно (например, по 10000 за раз) и следите за логами. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
phoenix.coprocessor.maxServerCacheBytes	Значение по умолчанию: `0` (выключено). Этот параметр определяет размер кеша на стороне сервера для результатов запросов в сопроцессоре на серверах регионов. Это может значительно повысить производительность часто выполняемых запросов с относительно небольшими наборами результатов. Значение определяется в байтах	Если у вас много повторяющихся запросов, начните с небольшого значения (например, `104857600` для 100 МБ). Внимательно следите за использованием памяти на серверах регионов. Слишком большое значение может привести к нехватке памяти и проблемам с производительностью. Также необходимо следить за уровнем использования кеша. Если он слишком низкий, память тратится впустую. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase
phoenix.index.builder.threads	Значение по умолчанию: `1`. Этот параметр определяет количество потоков, которые будут использоваться для построения глобальных индексов. Глобальные индексы строятся асинхронно в фоновом режиме, и этот параметр управляет уровнем параллелизма, используемым при построении индексов	Если нужно индексировать большое количество строк и вы хотите ускорить процесс построения индекса, увеличьте это значение до числа доступных ядер ЦП на серверах регионов. Если процесс построения индекса создает слишком большую нагрузку на серверы регионов, уменьшите это значение. Этот параметр нужно добавлять вручную в секции ***Custom hbase-site.xml*** на конфигурационной странице сервиса HBase

Настройка БД

Ключ строки

Ключ строки — важнейший фактор при планировании БД в HBase и, следовательно, в Phoenix. Правильно устроенный ключ строки может значительно повысить производительность запросов и локальность данных. Неправильно устроенный ключ строки может привести к перегрузкам, медленным запросам и общей неэффективности системы.

Ниже представлен список ключевых факторов при конструировании ключа строки:

- **Паттерны запросов**. Ключ строки следует выбирать в зависимости от того, как вы планируете запрашивать данные. Определите наиболее распространенные и критичные к производительности запросы и спроектируйте ключ строки для их оптимизации. Примите во внимание следующее:
  - **Сканирование диапазона**. Если вам часто требуется извлекать данные в определенном диапазоне, ключ строки должен включать дату или ее производную в качестве префикса.
  - **Точечные запросы**. Если вам нужно быстро извлекать отдельные строки по уникальному идентификатору, сделайте этот идентификатор ключом строки.
  - **Запросы на основе префикса**. Если вы часто используете запросы, основанные на префиксе определенного атрибута, включите этот атрибут в начало ключа строки.
- **Hotspotting**. Избегайте конструкций ключей строк, которые приводят к hotspotting. Это происходит, когда непропорционально большой объем данных записывается или считывается с одного сервера регионов, создавая узкое место. Распространенные причины:
  - **Монотонно возрастающие ключи**. Использование временных меток, порядковых номеров или других монотонно возрастающих значений в качестве ключа строки направит все новые операции записи в один регион, что приведет к перегрузке сервера.
  - **Последовательные ключи**. Подобно монотонно возрастающим ключам, последовательные ключи также могут вызывать hotspotting, если они сосредоточены в небольшом диапазоне.
- Salting. Это метод избегания hotspotting путем добавления префикса к ключу строки для распределения записи по нескольким регионам.
- **Предварительное разделение (pre-splitting)**. Подразумевает предварительное создание нескольких регионов для более равномерного распределения начальной нагрузки. Это особенно важно для больших наборов данных.
- **Длина ключа**. Старайтесь использовать как можно более короткие ключи строк, не жертвуя при этом уникальностью или производительностью запросов. Более короткие ключи уменьшают объем хранилища и повышают производительность сканирования.
- **Составные ключи**. Phoenix поддерживает составные ключи строк, которые объединяют несколько столбцов в один ключ. Это полезно, когда нужно выполнять запросы на основе комбинации атрибутов.
  - **Порядок столбцов**. Порядок столбцов в составном ключе имеет значение. Размещайте наиболее часто запрашиваемые столбцы в начале.
  - **Типы данных**. Выберите подходящие типы данных для каждого столбца в составном ключе, чтобы оптимизировать дисковое пространство и производительность.

Пример

Рассмотрим таблицу, хранящую данные датчиков, со следующими столбцами:

- `sensor_id`
- `timestamp`
- `temperature`
- `humidity`

Если вы часто запрашиваете данные датчиков в определенном временном диапазоне, можно спроектировать составной ключ строки как комбинацию `sensor_id` и `timestamp`. Однако, если количество датчиков невелико, а запись в определенные временные диапазоны достаточно интенсивна, возникнет проблема hotspotting. В этом случае вам потребуется добавить к `sensor_id` несколько salt-регионов.

Семейства столбцов

В HBase и Phoenix данные хранятся в семействах столбцов. Выбор правильного дизайна семейства столбцов имеет решающее значение для оптимизации эффективности хранения, производительности запросов и общей масштабируемости системы.

Ниже приведен список ключевых факторов, которые следует учитывать при дизайне семейства столбцов:

- **Локальность обращения**. Столбцы, к которым часто обращаются одновременно, следует сгруппировать в одно семейство столбцов. Это повышает производительность чтения, поскольку HBase извлекает данные из целых семейств столбцов сразу. И наоборот, столбцы, к которым редко обращаются одновременно, следует помещать в разные семейства.
- **Количество семейств столбцов**. Минимизируйте количество семейств столбцов в таблице. Каждое семейство столбцов имеет отдельные требования к хранилищу и памяти. Небольшое количество хорошо спроектированных семейств столбцов, как правило, предпочтительнее большого количества фрагментированных семейств. Идеальное количество семейств столбцов — 2 или 3.
- **Характеристики хранения**. Учитывайте характеристики хранения данных в каждом семействе столбцов:
  - **Тип данных**. Столбцы со схожими типами данных (например, строковые, числовые, двоичные) можно сгруппировать для оптимизации хранения и сжатия.
  - **Размер данных**. Столбцы, содержащие большие объекты (например, изображения, документы), следует размещать в отдельных семействах столбцов, чтобы избежать влияния на производительность других часто используемых данных.
- **Эволюция схемы**. Семейства столбцов обеспечивают определенную гибкость для эволюции схемы. Вы можете добавлять новые столбцы в существующее семейство столбцов, не влияя на другие семейства. Однако следует учитывать влияние добавления большого количества столбцов в одно семейство, поскольку это может повлиять на производительность.
- **Именование семейств столбцов**. Выбирайте описательные и содержательные имена для семейств столбцов, но делайте их максимально короткими.
- **Сжатие**. Для семейств столбцов можно настроить различные алгоритмы сжатия. Выберите подходящий алгоритм сжатия в зависимости от типа данных и паттернов доступа в каждом семействе.
- **Хранение в памяти**. Вы можете настроить хранение семейства столбцов в памяти для более быстрого доступа. Это полезно для часто используемых данных относительно небольшого объема. Однако следует учитывать использование памяти на серверах регионов.

Пример

Рассмотрим таблицу, хранящую информацию о профиле пользователя, со следующими столбцами: `user_id`, `username`, `email`, `first_name`, `last_name`, `address`, `city`, `state`, `zip` и `profile_picture`. Семейства столбцов можно распределить следующим образом:

- `info`: `username`, `email`, `first_name`, `last_name`
- `address`: `address`, `city`, `state`, `zip`
- `media`: `profile_picture`

Такое устройство позволяет группировать часто используемую информацию о пользователе в семейство столбцов `info`, выделяет адресную информацию в другое семейство и помещает большие фотографии профиля в третье семейство, чтобы не влиять на производительность других запросов.

Стратегии индексирования

Phoenix предоставляет мощные возможности индексирования для ускорения выполнения запросов к данным HBase. В отличие от традиционных реляционных баз данных, где индексы обычно создаются для небольшого количества столбцов, Phoenix позволяет создавать индексы для любой комбинации столбцов, включая выражения и функции. Выбор правильной стратегии индексирования критически важен для оптимизации производительности запросов и минимизации затрат на обслуживание индексов.

В Phoenix существует два основных типа индексов: глобальные и локальные.

Глобальные индексы

Глобальные индексы хранятся в отдельной таблице HBase и охватывают все строки исходной таблицы. Когда запрос использует глобальный индекс, Phoenix извлекает данные из таблицы индекса, а не из исходной таблицы.

Глобальные индексы лучше всего подходят для рабочих нагрузок с высокой интенсивностью чтения, когда запросы часто фильтруют данные по индексированным столбцам. Они обновляются синхронно во время записи данных. Это может повлиять на производительность записи, особенно для больших наборов данных.

Глобальные индексы непригодны для использования, если базовая таблица недоступна.

Локальные индексы

Локальные индексы хранятся на тех же серверах регионов, что и исходные данные таблиц. Когда запрос использует локальный индекс, Phoenix извлекает данные из таблицы индексов и исходной таблицы на одном сервере регионов.

Локальные индексы идеально подходят для рабочих нагрузок с высокой интенсивностью записи, когда запросам требуется фильтрация по неведущим столбцам ключа строки или столбцам, не входящим в глобальный индекс. Они обновляются асинхронно после записи данных. Это минимизирует влияние на производительность записи.

Локальные индексы доступны, даже если базовая таблица недоступна.

Примеры синтаксиса для создания глобального и локального индексов:

```sql
CREATE INDEX my_index ON my_table (column1, column2);
CREATE LOCAL INDEX my_index ON my_table (column1, column2);
```

Сжатие данных

Сжатие данных — важнейший метод экономии дискового пространства и повышения производительности ввода-вывода в HBase и Phoenix. Сжатие данных перед сохранением позволяет значительно сократить объем занимаемого дискового пространства и время чтения и записи данных. Однако сжатие также приводит к дополнительным затратам ресурсов процессора, связанным со сжатием и распаковкой данных. Выбор правильного алгоритма сжатия — это компромисс между экономией дискового пространства, производительностью ввода-вывода и загрузкой процессора.

HBase поддерживает несколько алгоритмов сжатия, перечисленных ниже, каждый из которых имеет свои особенности:

- **NONE**. Сжатие не используется. Это алгоритм по умолчанию. Он обеспечивает максимальную скорость чтения и записи, но требует больше всего дискового пространства.
- **LZO**. Быстрый и широко используемый алгоритм сжатия, обеспечивающий хороший баланс между степенью сжатия и загрузкой процессора. Требует отдельной установки и настройки.
- **GZIP**. Популярный алгоритм сжатия, обеспечивающий высокую степень сжатия, но потребляющий больше процессорных циклов, чем LZO. Подходит для данных, к которым редко обращаются.
- **Snappy**. Быстрый алгоритм сжатия, ориентированный на скорость, а не на высокую степень сжатия. Идеально подходит для данных, к которым часто обращаются и которые требуют малой задержки.
- **LZ4**. Чрезвычайно быстрый алгоритм сжатия, но обеспечивающий более низкую степень сжатия, чем другие алгоритмы.

Оптимальный алгоритм сжатия для ваших данных зависит от нескольких факторов:

- **Тип данных**. Некоторые алгоритмы сжатия лучше подходят для определенных типов данных, чем другие. Например, GZIP часто подходит для текстовых данных, а Snappy — для двоичных.
- **Паттерны доступа**. Если к данным обращаются часто, следует выбрать алгоритм сжатия с низкой задержкой, например, Snappy или LZ4. Если к данным обращаются редко, можно выбрать алгоритм сжатия с высокой степенью сжатия, например, GZIP.
- **Использование процессора**. Учитывайте нагрузку на процессор при сжатии. Если ваша система уже ограничена ресурсами процессора, возможно, стоит выбрать менее ресурсоемкий алгоритм сжатия, например, LZO или Snappy.
- **Объем хранилища**. Если объем хранилища ограничен, следует выбрать алгоритм сжатия с высокой степенью сжатия.

Локальность данных

Локальность данных — ключевая концепция HBase и Phoenix, означающая близость данных к вычислительным ресурсам, которым требуется доступ к этим данным. Максимизация локальности данных критически важна для достижения высокой производительности, поскольку сокращает объем данных, передаваемых по сети, что может стать серьезным узким местом.

HBase хранит данные в регионах, распределенных по серверам регионов в кластере. Каждый сервер регионов отвечает за управление подмножеством данных. Когда клиентское приложение запрашивает данные, HBase пытается обработать запрос с сервера регионов, ближайшего к данным.

На локальность данных влияют следующие факторы:

- **Конструкция ключа строки**. Ключ строки — важнейший фактор, влияющий на локальность данных. Правильно спроектированный ключ строки группирует связанные данные на одном сервере регионов, что упрощает доступ к ним.
- **Разделение и объединение регионов**. HBase автоматически разделяет и объединяет регионы в зависимости от размера данных и нагрузки. Если регионы разделяются и объединяются некорректно, локальность данных может пострадать. Важно отслеживать размеры регионов и соответствующим образом настраивать политики разделения и объединения регионов.
- **Размещение данных**. HBase пытается разместить данные на серверах регионов, расположенных близко к потребителям данных. Однако такие факторы, как топология сети и rack awareness, могут влиять на размещение данных.
- **Сжатие**. Сжатие — это процесс слияния и перезаписи файлов данных в HBase. Сжатие может улучшить локальность данных за счет реорганизации данных на диске.
- **Массовая загрузка**. Использование массовой загрузки данных в HBase позволяет предварительно сортировать данные, что может улучшить локальность данных.

Для улучшения локальности данных можно использовать следующие стратегии:

- **Оптимальная конструкция ключа строки**. Правильно спроектированный ключ строки крайне важен для локальности данных. Выберите ключ строки, который группирует связанные данные на одном сервере регионов. Рассмотрите возможность использования salting для предотвращения перегрузки, сохраняя при этом локальность связанных данных в пределах одного региона с salting.
- **Предварительное разделение**. Предварительное разделение таблицы на несколько регионов может улучшить локальность данных за счет распределения данных по нескольким серверам регионов с самого начала.
- **Rack awareness**. Настройте HBase с учетом топологии стоек физических серверов вашего кластера. Это позволит HBase размещать данные на серверах регионов, находящихся в той же стойке, где расположены потребители данных.
- **Конфигурация HDFS**. Правильно настройте HDFS, чтобы обеспечить хранение данных в непосредственной близости от серверов регионов.

Методы оптимизации запросов

План выполнения запроса

Понимание плана выполнения запроса имеет основополагающее значение для оптимизации SQL-запросов в Phoenix. План выполнения описывает шаги, которые Phoenix предпримет для извлечения и обработки данных. Анализируя этот план, вы можете выявить потенциальные узкие места и области для улучшения.

Вы можете получить план выполнения запроса с помощью оператора `EXPLAIN PLAN` в Phoenix. Например:

```sql
EXPLAIN PLAN SELECT * FROM users WHERE age > 30 AND city = 'New York';
```

Пример результата:

```
CLIENT 1-WAY PARALLEL 1/1 [SERVER FILTER]
    SERVER FILTER BY "AGE" > 30
        CLIENT 1-WAY FULL SCAN OVER USERS
```

План выполнения обычно представлен в виде древовидной структуры, где корневой узел представляет результат, а конечные узлы — источники данных. Каждый узел дерева описывает операцию, которую выполнит Phoenix. Ниже представлена таблица с описанием операций плана выполнения.

Операции плана выполнения

Операция	Описание
SERIAL	Указывает, что операция будет выполняться последовательно на одном сервере регионов. Это, как правило, менее эффективно, чем параллельное выполнение
PARALLEL <n>-WAY	Указывает, что операция будет выполняться параллельно на нескольких серверах регионов, где `<n>` определяет количество серверов (степень параллелизма). Ищите возможности для повышения параллелизма там, где это необходимо, но помните об ограничениях ресурсов
FULL SCAN OVER <table_name>	Обозначает полное сканирование таблицы, то есть Phoenix прочитает каждую строку в таблице, чтобы найти нужные. Полное сканирование таблицы чаще всего неэффективно, особенно для больших таблиц. Полное сканирование индексной таблицы бывает вполне эффективно
RANGE SCAN OVER <table_name> <startRow, stopRow>	Обозначает сканирование диапазона, при котором Phoenix будет читать только подмножество строк таблицы на основе диапазона ключей строк. Это эффективнее, чем полное сканирование таблицы, но все еще может быть неэффективным, если диапазон слишком велик. Индексная таблица будет определять свои диапазоны иначе, чем исходная таблица
POINT LOOKUP OVER <table_name> <rowKey>	Обозначает точечный поиск, при котором Phoenix напрямую извлекает одну строку по ее ключу. Это наиболее эффективный тип операции чтения
CLIENT <op type>	Указывает, что выполнение запроса происходит на стороне клиента
JOIN <join type>	Указывает тип выполненной операции `JOIN` (например, `STAR JOIN`). Анализ операций объединения имеет решающее значение для оптимизации
FILTER <predicate>	Указывает, что Phoenix фильтрует данные на основе предиката (предложения `WHERE`). Убедитесь, что фильтр максимально избирателен
AGGREGATE <aggregation function>	Указывает, что Phoenix выполняет функцию агрегации (например, `SUM`, `AVG`, `COUNT`)
SORT	Указывает, что результаты сортируются. Сортировка может быть затратной, поэтому по возможности избегайте ее. Если вы можете использовать индекс для обеспечения требуемого порядка сортировки, сделайте это

На основе анализа плана выполнения вы можете предпринять шаги для повышения производительности запросов. В приведенном выше примере вы можете создать индекс по столбцам `age` и `city` подобным образом:

```sql
CREATE INDEX idx_users_age_city ON users (age, city);
```

После создания индекса план выполнения может выглядеть следующим образом:

```
CLIENT 1-WAY RANGE SCAN OVER IDX_USERS_AGE_CITY [30,]
    SERVER FILTER BY "CITY" = 'New York'
```

Эффективность SQL

Написание эффективных SQL-запросов имеет первостепенное значение для оптимальной производительности Phoenix. Хотя Phoenix использует HBase для хранения данных, неэффективное применение SQL может привести к медленному выполнению запросов даже при правильной индексации таблиц. В этом разделе рассматривается создание SQL-запросов, которые минимизируют нагрузку на Phoenix и максимально используют ресурсы.

Ключевые принципы написания эффективного SQL-запроса:

- **Выбирайте только необходимые столбцы**. Явно указывайте нужные столбцы в предложении `SELECT`. Использование предложения `SELECT *` позволяет извлечь все столбцы из таблицы, даже если вам нужны лишь некоторые из них. Это увеличивает объем данных, которые необходимо прочитать, передать и обработать.
- **Выполняйте фильтрацию эффективно и на ранней стадии**. Применение фильтров на поздних этапах выполнения запроса (например, после полного сканирования таблицы) может быть крайне неэффективным. Используйте предложение `WHERE` для фильтрации данных на максимально ранней стадии. Убедитесь, что предложение `WHERE` использует доступные индексы. Наиболее строгие условия следует размещать в предложении `WHERE` в первую очередь.
- **Используйте правильные типы данных**. Несоответствие типов данных в сравнениях и предикатах может помешать Phoenix эффективно использовать индексы и привести к дополнительным расходам ресурсов на преобразование типов. Убедитесь, что типы данных в ваших SQL-запросах соответствуют типам данных соответствующих столбцов в схеме таблицы.
- **Используйте встроенные функции эффективно**. Некоторые встроенные функции могут быть эффективнее других. Изучите характеристики производительности различных встроенных функций. Используйте функции, оптимизированные для HBase/Phoenix.
- **Избегайте ненужных подзапросов**. Подзапросы часто можно переписать как объединения, что может быть более эффективно в Phoenix. Коррелированные подзапросы (подзапросы, зависящие от внешнего запроса) могут быть особенно проблемными. Старайтесь переписывать подзапросы как объединения, когда это возможно. Рассмотрите денормализацию в качестве альтернативы, если подзапрос используется часто.
- **Ограничьте набор результатов**. Получение больших наборов результатов может создать нагрузку на клиент и сеть. Используйте предложение `LIMIT` для ограничения количества возвращаемых строк. Если требуется разбиение на страницы, используйте `LIMIT` и `OFFSET` вместе.
- **Используйте COUNT(*) с осторожностью**. Функция `COUNT(*)` для большой таблицы без предложения `WHERE` может быть очень затратной, так как требует сканирования всей таблицы. Если нужно подсчитать все строки в таблице, рассмотрите возможность использования предварительного агрегированного подсчета или облегченного подхода, если таблица не слишком большая. По возможности добавьте предложение `WHERE`, чтобы ограничить подсчет меньшим подмножеством данных.
- **Используйте UPSERT SELECT для эффективного обновления и вставки данных**. Вставка данных построчно может быть очень неэффективной. Используйте `UPSERT SELECT` для пакетной вставки или обновления данных. Это особенно полезно при загрузке данных из других таблиц или внешних источников.

Оптимизация JOIN

Объединения в Phoenix могут быть относительно затратными операциями по сравнению с точечным поиском или сканированием диапазонов. Поэтому оптимизация объединений критически важна для достижения высокой производительности при работе со связанными данными в нескольких таблицах. Распределенная природа HBase делает производительность объединений более важной, чем в традиционных реляционных базах данных. В этом разделе представлены стратегии оптимизации объединений в Phoenix, включая выбор правильного типа объединения, использование локальности данных и, при необходимости, полный отказ от объединений посредством денормализации.

Phoenix поддерживает несколько типов объединений и связанных с ними стратегий, каждая из которых обладает своими характеристиками производительности:

- **Star join**. Оптимизированы для сценариев, когда одна большая таблица (таблица фактов) объединяется с несколькими меньшими таблицами измерений. Phoenix пытается транслировать только те части таблиц измерений, которые требуются запросу, на все соответствующие серверы регионов (содержащие данные из таблицы фактов в объединении), что обеспечивает эффективные локальные объединения. Это часто предпочтительная стратегия объединения в Phoenix, если модель данных и паттерны запросов ее поддерживают. Phoenix обычно может определить целесообразность выполнения star join на основе доступной ему статистики данных.
- **Broadcast join**. Оптимизированы для сценариев, когда одна таблица действительно небольшая (с запасом помещается в памяти на каждом сервере регионов) и объединяется с одной или несколькими более крупными таблицами. Phoenix передает всю небольшую таблицу на все серверы регионов. Это отличается от star join, при котором отправляются только необходимые части таблиц измерений. Такой подход может обеспечить отличную производительность, поскольку операции объединения выполняются локально на каждом сервере регионов, что минимизирует передачу данных. Phoenix автоматически использует эту оптимизацию, когда это применимо.
- **Sort-merge join**. Хотя Phoenix поддерживает объединения типа sort-merge, они, как правило, менее эффективны, чем объединения типа star или broadcast, особенно в распределенных средах, таких как HBase. Они предполагают сортировку входных таблиц по ключам объединения с последующим слиянием отсортированных результатов. Узкое место производительности возникает из-за дополнительных расходов ресурсов на сортировку и потенциального перемешивания данных по сети. Phoenix обычно прибегает к объединениям типа sort-merge только в тех случаях, когда невозможно применить другие, более эффективные стратегии объединения. Это может произойти, если таблицы очень большие и не могут быть эффективно переданы, или если ключи объединения не подходят для объединения на основе хеша.

Возможные стратегии оптимизации объединений:

- **Денормализация**. Часто наилучшей оптимизацией для объединений является их полное исключение. Рассмотрите возможность денормализации данных путем встраивания связанной информации непосредственно в таблицу фактов или создания предобъединенных представлений. Например, вместо объединения таблиц `orders` и `customers` для получения информации о клиенте по каждому заказу, можно включить соответствующие данные о клиенте (например, имя клиента, город) непосредственно в таблицу `orders`. Однако денормализация может привести к избыточности данных и потенциальным проблемам с согласованностью, поэтому тщательно взвесьте все за и против, исходя из вашего конкретного варианта использования.
- **Локальность данных**. Размещайте связанные таблицы на одних серверах регионов, чтобы минимизировать сетевой трафик во время объединений. Этого можно добиться с помощью стратегий совместного размещения в HBase.
- **Выбор правильного порядка объединений**. Порядок объединения таблиц может существенно влиять на производительность. Как правило, лучше всего начинать с таблицы с наиболее строгим фильтром и объединять ее с другими таблицами по возрастанию их размера. Ранняя фильтрация сокращает объем данных, необходимых для обработки в последующих операциях объединения.
- **Фильтрация перед объединением**. Применяйте предложения `WHERE` для фильтрации данных из каждой таблицы перед операцией `JOIN`. Уменьшение размера объединяемых таблиц повышает производительность объединения.
- **Объединения типа broadcast для небольших таблиц**. Для объединений, включающих небольшую таблицу измерений и большую таблицу фактов, используйте объединение типа broadcast. Такие объединения копируют меньшую таблицу на все серверы регионов, что позволяет выполнять локальные объединения с большей таблицей. Phoenix может автоматически выбирать объединение типа broadcast при обнаружении небольшой таблицы. Вы часто можете повлиять на это решение с помощью подсказок.
- **Используйте оптимизацию объединения типа star**. Если ваша модель данных соответствует схеме star (одна большая таблица фактов, объединенная с несколькими меньшими таблицами измерений), воспользуйтесь оптимизацией объединения типа star в Phoenix.

Подсказки

Подсказки — это директивы, которые можно добавлять в SQL-запросы для управления планом выполнения запросов в Phoenix. Они позволяют направлять работу оптимизатора и потенциально повышать производительность, когда план выполнения по умолчанию неоптимален. Хотя Phoenix автоматически оптимизирует запросы, подсказки позволяют переопределять его решения в определенных ситуациях. Используйте подсказки с осторожностью, так как они могут сделать ваши запросы менее приспособленными к портированию и более сложными в дальнейшей поддержке.

Подсказки размещаются после ключевых слов `SELECT`, `INSERT`, `UPDATE` или `DELETE` и имеют следующий синтаксис:

```sql
SELECT /*+ hint1, hint2, ... */ column1, column2 FROM table_name WHERE condition;
```

При использовании подсказок учитывайте следующее:

- **Используйте с осторожностью**. Подсказки переопределяют оптимизатор запросов по умолчанию, что может скрывать изначально существующие проблемы с производительностью или приводить к неоптимальным планам при неправильном использовании.
- **Тщательно тестируйте**. Всегда проверяйте производительность запросов с подсказками и без них, чтобы убедиться, что использование подсказки действительно улучшает производительность.
- **Обеспечивайте удобство поддержки**. Подсказки могут сделать ваши запросы менее портативными и более сложными в обслуживании. Задокументируйте причины использования подсказки и отслеживайте ее эффективность с течением времени.
- **Помните об изменениях в оптимизаторе**. Оптимизатор запросов Phoenix может измениться в будущих версиях, сделав ваши подсказки неэффективными или даже вредными. Регулярно проверяйте и обновляйте подсказки по мере необходимости.
- **Понимайте план выполнения**. Используйте функцию `EXPLAIN PLAN`, чтобы понять, как ваши подсказки влияют на план выполнения запроса.

Сервер запросов

Сервер запросов Phoenix (Phoenix Query Server, PQS) предоставляет доступ к Phoenix клиентам, не имеющим прямого доступа к кластеру HBase. Он действует как прокси-сервер, позволяя клиентам отправлять SQL-запросы через тонкий драйвер JDBC, абстрагируясь от сложностей прямого взаимодействия с HBase. Такой подход полезен в ситуациях, когда ограничения безопасности, сетевые конфигурации или зависимости на стороне клиента препятствуют прямому подключению к кластеру HBase.

Phoenix Query Server имеет клиент-серверную архитектуру:

1. **Клиент**. Клиентское приложение использует тонкий драйвер JDBC для подключения к PQS. Этот драйвер часто менее ресурсоемкий, чем стандартный драйвер Phoenix JDBC, поскольку ему не нужно обрабатывать прямые соединения с HBase.
2. **Сервер запросов Phoenix**. PQS — это автономный процесс, работающий вне кластера HBase (хотя часто размещается на узлах HBase). Он получает SQL-запросы от клиентов, преобразует их в запросы Phoenix, выполняет их в кластере HBase и возвращает результаты клиенту. PQS выполняет такие задачи, как:
   - Аутентификация и авторизация.
   - Анализ и оптимизация запросов (с помощью внутреннего оптимизатора Phoenix).
   - Организация пула подключений к HBase.
   - Управление наборами результатов.
3. **Кластер HBase**. Кластер HBase хранит данные и выполняет операции доступа к ним. PQS подключается к кластеру HBase с помощью стандартного драйвера Phoenix JDBC.

Преимущества использования Phoenix Query Server:

- **Упрощенный клиентский доступ**. Клиенты могут получать доступ к данным Phoenix через стандартный интерфейс JDBC, без необходимости установки полных клиентских библиотек Phoenix или прямого доступа к кластеру HBase.
- **Безопасность**. PQS предоставляет центральную точку аутентификации и авторизации, позволяя контролировать доступ к данным Phoenix. Вы можете ограничить прямой доступ к кластеру HBase, повышая таким образом уровень безопасности.
- **Сетевая изоляция**. PQS может быть развернут в DMZ или другой защищенной сетевой зоне, изолируя кластер HBase от прямого доступа клиентов. Это критически важно для безопасности во многих организациях.
- **Управление ресурсами**. PQS может управлять подключениями к кластеру HBase, предотвращая перегрузку кластера клиентами со слишком большим количеством подключений. Это обеспечивает множественность подключений и управление ресурсами.
- **Независимость от языка**. Клиенты могут быть написаны на любом языке, поддерживающем JDBC, что обеспечивает гибкость в разработке приложений.
- **Тонкий драйвер JDBC**. Клиент JDBC имеет небольшой размер, что упрощает его развертывание на различных платформах.

Логирование в Phoenix

Владимир Адаменков

Содержание

Назад к содержанию

Обзор

Настройка логирования запросов

Просмотр лога запросов

Настройка логирования Phoenix Query Server

Обзор

Apache Phoenix позволяет выполнять логирование выполненных запросов `SELECT`, сохраняя их в таблицу `SYSTEM.LOG`. Можно указывать уровень подробности и среднее количество запросов для логирования. Кроме того, можно добавить лог компонента Phoenix Query Server к логу сервиса HBase.

Настройка логирования запросов

Чтобы включить логирование запросов в Phoenix, перейдите в веб-интерфейс ADCM и выполните следующие действия:

1. Перейдите в раздел ***Clusters*** и выберите ваш кластер.
2. Откройте вкладку ***Services*** и выберите сервис ***HBase***.
3. Активируйте переключатель ***Show advanced*** и раскройте секцию параметров ***Custom hbase-site.xml***.
4. Нажмите ***Add property*** и в качестве названия параметра введите `phoenix.log.level`. В качестве значения параметра введите одно из следующих значений:
   - `OFF` — запросы логироваться не будут (значение по умолчанию);
   - `INFO` — будут логироваться основные сведения о запросах;
   - `DEBUG` — будут логироваться подробные сведения о запросах;
   - `TRACE` — будут логироваться подробные сведения о запросах, включая bind-параметры.
5. Нажмите ***Add property*** и в качестве названия параметра введите `phoenix.log.sample.rate`. В качестве значения параметра введите численное представление вероятности логирования запроса от `0.0` до `1.0`. Например, значение `0.5` означает, что примерно половина выполненных запросов `SELECT` будет логироваться. Значение по умолчанию: `1.0` (все запросы).
6. Сохраните конфигурацию, нажав ***Save*** → ***Create***, и перезапустите сервис, выполнив действие ***Actions*** → ***Reconfig and graceful restart***.
7. Если на текущий момент была загружена командная оболочка Phoenix, выйдите из неё и снова загрузите, чтобы изменения вступили в силу.
8. Если необходимо, с помощью команды `GRANT` назначьте права на запись в таблицу `SYSTEM.LOG` пользователям, чьи запросы необходимо логировать.

> ВАЖНО
>
> Лог запросов смогут увидеть все имеющие доступ к таблице `SYSTEM.LOG` пользователи. Используйте логирование запросов с осторожностью.

Просмотр лога запросов

Для просмотра лога запросов нужно составить и выполнить соответствующий запрос к таблице `SYSTEM.LOG`. Пример:

```
SELECT * FROM SYSTEM.LOG;
```

Результат:

```
+---------------+--------------+--------------------------------------+-----------+------------------+--------------+-------------
----------------------------------------+
|  START_TIME   |  TABLE_NAME  |               QUERY_ID               | TENANT_ID |       USER       |  CLIENT_IP   |
QUERY                      |
+---------------+--------------+--------------------------------------+-----------+------------------+--------------+-------------
----------------------------------------+
| 1743693276315 | WEAPON_TYPES | efa9397d-56b4-4766-940f-b2dca9ed88ac |           | rsa-key-20240513 | 10.92.40.104 | SELECT
ammo_type FROM weapon_types GROUP BY ammo_t |
| 1744124555613 |              | a380c6f4-cffc-4fd8-b35c-1ef02aa52632 |           | rsa-key-20240513 | 10.92.40.104 | SELECT
ammo_type FROM weapon_types GROUP BY ammo_t |
+---------------+--------------+--------------------------------------+-----------+------------------+--------------+-------------
----------------------------------------+
```

Таблица `SYSTEM.LOG` имеет гораздо больше столбцов, чем может поместиться в окне терминала какой бы то ни было ширины. Можно использовать команду `!outputformat vertical` перед выполнением запроса и/или сузить его условия.

Для очистки таблицы `SYSTEM.LOG` используйте выражение `DELETE` с условием `WHERE`. Например, чтобы удалить все записи из этой таблицы, выполните следующий запрос:

```
DELETE FROM SYSTEM.LOG WHERE START_TIME > 0;
```

Настройка логирования Phoenix Query Server

Для настройки логирования Phoenix Query Server перейдите в веб-интерфейс ADCM и выполните следующие действия:

1. Перейдите в раздел ***Clusters*** и выберите ваш кластер.
2. Откройте вкладку ***Services*** и выберите сервис ***HBase***.
3. Активируйте переключатель ***Show advanced*** и выберите параметр ***Custom log4j.properties***.
4. Добавьте следующие строки к содержимому параметра:

```
log4j.logger.org.apache.phoenix.queryserver=DEBUG
log4j.logger.org.apache.phoenix.queryserver.server=DEBUG
log4j.logger.org.apache.phoenix=DEBUG
log4j.logger.org.eclipse.jetty.io.AbstractEndPoint=DEBUG
```

5. Замените значение `log4j.threshold` на `DEBUG` и нажмите ***Apply***.
6. Сохраните конфигурацию, нажав ***Save*** → ***Create*** и перезапустите сервис, выполнив действие ***Actions*** → ***Reconfig and graceful restart***.

Подключение к Solr через CLI

Константин Алпашкин

Содержание

Назад к содержанию

- Команды управления: start, stop, restart
- Команды вывода системной информации
  - Параметры команд
  - Примеры
- Команды управления коллекциями и ядрами
  - Параметры команд
  - Примеры
  - Ядра
- Аутентификация
  - Параметры команд
  - Примеры
- Команды ZooKeeper
  - Параметры команд
  - Примеры
- Команды экспорта данных
  - Параметры команд
  - Примеры

Компоненты Solr Server, установленные на хостах ADH, предоставляют скрипт */usr/lib/solr/bin/solr*, который позволяет управлять Solr-кластером из командной строки. Несмотря на то, что всегда рекомендуется использовать ADCM для управления Solr-кластером, некоторые CLI-команды Solr могут быть полезны для выполнения действий, недоступных в веб-интерфейсе Solr. Примеры этих команд описаны далее в статье.

Использование скрипта */usr/lib/solr/bin/solr* показано ниже:

```
$ /usr/lib/solr/bin/solr <COMMAND> [<PARAMETERS>]
```

Подробная информация о всех поддерживаемых значениях `COMMAND` и `PARAMETERS` доступна в документации Solr. Для получения справочной информации по конкретной команде используйте синтаксис:

```
$ /usr/lib/solr/bin/solr <COMMAND> -help
```

Для более подробного вывода используйте флаг `-V`.

> **ПРИМЕЧАНИЕ**
>
> Являясь сервисом ADH, Solr всегда запускается в режиме SolrCloud. Поэтому некоторые команды, относящиеся к standalone-режиму, могут не иметь никакого эффекта.

Команды управления: start, stop, restart

Несмотря на то, что скрипт */usr/lib/solr/bin/solr* поддерживает команды типа `start`, `stop` и `restart`, настоятельно рекомендуется использовать ADCM для выполнения этих действий. Под капотом ADCM вызывает эти же команды для запуска и остановки Solr-процессов, однако ADCM осуществляет дополнительные манипуляции для корректного выполнения этих команд. Например, установка переменных окружения, запуск команд от имени пользователя с достаточными правами доступа, обработка ошибок и так далее.

Выполнение команд `start`, `stop` и `restart` с помощью */usr/lib/solr/bin/solr* может привести к проблемам с запуском Solr и сбоям в работе всего Solr-кластера.

Команды вывода системной информации

Solr предоставляет несколько команд, которые могут быть полезны для получения диагностической информации о состоянии вашего Solr-кластера. Некоторые из этих команд представлены ниже:

```
$ /usr/lib/solr/bin/solr version
$ /usr/lib/solr/bin/solr status
$ /usr/lib/solr/bin/solr healthcheck [<PARAMETERS>]
```

Параметры команд

Команда: healthcheck

Параметр	Описание
-c <collection_name>	Имя коллекции
-z <zk_connection_string>	Опциональный параметр. Строка подключения ZooKeeper. Можно не указывать, если строка подключения к ZooKeeper не изменялась вручную

Примеры

Вывод версии Solr

```
$ /usr/lib/solr/bin/solr version
```

Пример вывода:

```
8.11.2
```

Получение статуса узла Solr

```
$ /usr/lib/solr/bin/solr status
```

Пример вывода:

```
Found 1 Solr nodes:

Solr process 23658 running on port 8983
{
  "solr_home":"/srv/solr/server",
  "version":"8.11.2 3cb85f40e394dcfb50fe77310908cce385381ba2 - jenkins - 2024-03-20 17:52:09",
  "startTime":"2024-05-23T22:52:07.271Z",
  "uptime":"0 days, 0 hours, 0 minutes, 6 seconds",
  "memory":"105.9 MB (%21.6) of 491 MB",
  "cloud":{
    "ZooKeeper":"ka-adh-1.ru-central1.internal:2181,ka-adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.
internal:2181/Arenadata.Hadoop-7.solr.server",
    "liveNodes":"3",
    "collections":"0"}}
```

Проверка состояния коллекции

```
$ /usr/lib/solr/bin/solr healthcheck -c test_collection
```

Пример вывода:

```
{
  "collection":"test_collection",
  "status":"healthy",
  "numDocs":0,
  "numShards":2,
  "shards":[
    {
      "shard":"shard1",
      "status":"healthy",
      "replicas":[
        {
          "name":"core_node3",
          "url":"http://ka-adh-3.ru-central1.internal:8983/solr/test_collection_shard1_replica_n1/",
          "numDocs":0,
          "status":"active",
          "uptime":"0 days, 11 hours, 21 minutes, 7 seconds",
          "memory":"143.6 MB (%29.4) of 489 MB"},
        {
          "name":"core_node5",
          "url":"http://ka-adh-1.ru-central1.internal:8983/solr/test_collection_shard1_replica_n2/",
          "numDocs":0,
          "status":"active",
          "uptime":"0 days, 11 hours, 21 minutes, 7 seconds",
          "memory":"67.9 MB (%14) of 485 MB",
          "leader":true},
        {
          "name":"core_node7",
          "url":"http://ka-adh-2.ru-central1.internal:8983/solr/test_collection_shard1_replica_n4/",
          "numDocs":0,
          "status":"active",
          "uptime":"0 days, 11 hours, 21 minutes, 7 seconds",
          "memory":"63 MB (%13) of 485 MB"}]},
    {
      "shard":"shard2",
      "status":"healthy",
      "replicas":[
        {
          "name":"core_node9",
          "url":"http://ka-adh-3.ru-central1.internal:8983/solr/test_collection_shard2_replica_n6/",
          "numDocs":0,
          "status":"active",
          "uptime":"0 days, 11 hours, 21 minutes, 7 seconds",
          "memory":"144.8 MB (%29.6) of 489 MB"},
        {
          "name":"core_node11",
          "url":"http://ka-adh-1.ru-central1.internal:8983/solr/test_collection_shard2_replica_n8/",
          "numDocs":0,
          "status":"active",
          "uptime":"0 days, 11 hours, 21 minutes, 7 seconds",
          "memory":"69.3 MB (%14.3) of 485 MB"},
        {
          "name":"core_node12",
          "url":"http://ka-adh-2.ru-central1.internal:8983/solr/test_collection_shard2_replica_n10/",
          "numDocs":0,
          "status":"active",
          "uptime":"0 days, 11 hours, 21 minutes, 7 seconds",
          "memory":"65.1 MB (%13.4) of 485 MB",
          "leader":true}]}]}
```

Команды управления коллекциями и ядрами

Вы можете использовать скрипт */usr/lib/solr/bin/solr* для создания и удаления Solr-коллекций.

Использование команд показано ниже:

```
$ /usr/lib/solr/bin/solr create [<PARAMETERS>]
$ /usr/lib/solr/bin/solr create_collection [<PARAMETERS>]
$ /usr/lib/solr/bin/solr delete [<PARAMETERS>]
```

> **ПРИМЕЧАНИЕ**
>
> В ADH-кластере команды `create` и `create_collection` эквивалентны. При запуске `create` определяется текущий режим Solr (в ADH это всегда SolrCloud) и вызывается команда `create_collection` с тем же набором параметров.

Параметры команд

Команды: create/create_collection

Параметр	Описание
-c <collection>	Имя создаваемой коллекции
-d <conf_dir>	Указывает директорию конфигурации (configset) для создания коллекции. Если параметр не указан, используется конфигсет */usr/lib/solr/server/solr/configsets/_default*
-n <config_name>	Имя, используемое для загрузки конфигсета в ZooKeeper. По умолчанию используется имя коллекции
-s <shards>, -shards <shards>	Количество шардов для разбиения коллекции. Значение по умолчанию — `1`
-rf <replicas>, -replicationFactor <replicas>	Количество реплик каждого документа в коллекции. Значение по умолчанию — `1`, указывающее на отсутствие репликации

Команда: delete

Параметр	Описание
-c <collection>	Имя удаляемой коллекции
-deleteConfig <true|false>	Указывает, удалять ли директорию конфигурации из ZooKeeper. По умолчанию используется значение `true`. Если директория конфигурации ZooKeeper уже используется другой коллекцией, она не будет удалена, даже если указать `-deleteConfig true`

Примеры

Создание базовой коллекции

```
$ /usr/lib/solr/bin/solr create_collection -c test_collection1
```

Пример вывода:

```
Created collection 'test_collection1' with 1 shard(s), 1 replica(s) with config-set 'test_collection1'
```

Создание коллекции с несколькими шардами и репликами

```
$ /usr/lib/solr/bin/solr create_collection -c test_collection -shards 2 -replicationFactor 3 -V
```

Пример вывода:

```
Connecting to ZooKeeper at ka-adh-1.ru-central1.internal:2181,ka-adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.
internal:2181/Arenadata.Hadoop-8.solr.server ...
INFO  - 2024-05-28 08:23:31.128; org.apache.solr.common.cloud.ConnectionManager; Waiting for client to connect to ZooKeeper
INFO  - 2024-05-28 08:23:31.145; org.apache.solr.common.cloud.ConnectionManager; zkClient has connected
INFO  - 2024-05-28 08:23:31.145; org.apache.solr.common.cloud.ConnectionManager; Client is connected to ZooKeeper
INFO  - 2024-05-28 08:23:31.158; org.apache.solr.common.cloud.ZkStateReader; Updated live nodes from ZooKeeper... (0) -> (3)
INFO  - 2024-05-28 08:23:31.269; org.apache.solr.client.solrj.impl.ZkClientClusterStateProvider; Cluster at ka-adh-1.ru-
central1.internal:2181,ka-adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.internal:2181/Arenadata.Hadoop-8.solr.server
ready
Uploading /usr/lib/solr/server/solr/configsets/_default/conf for config test_collection to ZooKeeper at ka-adh-1.ru-central1.
internal:2181,ka-adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.internal:2181/Arenadata.Hadoop-8.solr.server

Creating new collection 'test_collection' using command:
http://ka-adh-1.ru-central1.internal:8983/solr/admin/collections?
action=CREATE&name=test_collection&numShards=2&replicationFactor=3&maxShardsPerNode=-1&collection.configName=test_collection

{
  "responseHeader":{
    "status":0,
    "QTime":2325},
  "success":{
    "ka-adh-1.ru-central1.internal:8983_solr":{
      "responseHeader":{
        "status":0,
        "QTime":1199},
      "core":"test_collection_shard2_replica_n8"},
    "ka-adh-3.ru-central1.internal:8983_solr":{
      "responseHeader":{
        "status":0,
        "QTime":1610},
      "core":"test_collection_shard1_replica_n1"},
    "ka-adh-2.ru-central1.internal:8983_solr":{
      "responseHeader":{
        "status":0,
        "QTime":1822},
      "core":"test_collection_shard2_replica_n10"}}}
```

Удаление коллекции

```
$ /usr/lib/solr/bin/solr delete -c test_collection1
```

Пример вывода:

```
{
  "responseHeader":{
    "status":0,
    "QTime":297},
  "success":{
    "ka-adh-1.ru-central1.internal:8983_solr":{"responseHeader":{
        "status":0,
        "QTime":50}},
    "ka-adh-3.ru-central1.internal:8983_solr":{"responseHeader":{
        "status":0,
        "QTime":119}},
    "ka-adh-2.ru-central1.internal:8983_solr":{"responseHeader":{
        "status":0,
        "QTime":130}}}}


Deleted collection 'test_collection1' using command:
http://ka-adh-1.ru-central1.internal:8983/solr/admin/collections?action=DELETE&name=test_collection1
```

Ядра

Поскольку компоненты Solr Server всегда запускаются в режиме SolrCloud, в этом режиме Solr оперирует коллекциями (поисковый индекс, распределенный по нескольким узлам кластера), а не ядрами. По этой причине команды, связанные с ядрами, не имеют эффекта, а их описание выходит за рамки данной статьи.

Аутентификация

С помощью */usr/lib/solr/bin/solr* можно настраивать аутентификацию в Solr. Использование соответствующих команд показано ниже:

```
$ /usr/lib/solr/bin/solr auth enable [<PARAMETERS>]
$ /usr/lib/solr/bin/solr auth disable
```

Параметры команд

Команда: auth enable

Параметр	Описание
-authConfDir <dir>	Указывает директорию для хранения файлов конфигурации, связанных с аутентификацией
-blockUnknown <true|false>	Блокирует доступ ко всем Solr-эндпойнтам для неаутентифицированных пользователей
-credentials <username:password>	Учетные данные в формате `username:password`
-prompt <true|false>	Запрашивает ввод учетных данных. Необходимо указать либо параметр `-prompt`, либо `-credentials`
-type <type>	Указывает механизм аутентификации (`basicAuth` или `kerberos`). По умолчанию используется `basicAuth`
-updateIncludeFileOnly <true|false>	Обновляет только файл *solr.in.sh*, не включая/отключая аутентификацию
-z <zkString>	Строка подключения к ZooKeeper. Можно не указывать, если строка подключения к ZooKeeper не изменялась вручную
-d <dir>	Указывает директорию сервера Solr (по умолчанию */usr/lib/solr/server/*). Используйте этот параметр, только если вы изменяли путь к домашней директории Solr
-s <dir>	Указывает домашнюю директорию Solr для хранения учетных данных и файлов конфигурации аутентификации (например, *basicAuth.conf*)

Примеры

Включение basic-аутентификации

Перед выполнением следующей команды убедитесь, что файл security.json загружен в ZooKeeper. Загрузить файл можно с помощью ZooKeeper CLI или с помощью скрипта */usr/lib/solr/server/scripts/cloud-scripts/zkcli.sh*.

```
$ /usr/lib/solr/bin/solr auth enable \
    -type basicAuth \
    -prompt true \
    -z ka-adh-1.ru-central1.internal:2181,ka-adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.internal:2181/Arenadata.Hadoop-
6.solr.server \
    -blockUnknown true
```

Пример вывода:

```
Security is enabled. You can disable it with 'bin/solr auth disable'. Existing security.json:
{"authentication":{"blockUnknown":true,"class":"solr.BasicAuthPlugin","realm":"My Solr
users","forwardCredentials":false},"authorization":{"class":"solr.RuleBasedAuthorizationPlugin","permissions":
[{"name":"security-edit","role":"admin"}],"user-role":{"solr":"admin"}}}
```

Отключение аутентификации

```
$ /usr/lib/solr/bin/solr auth disable
```

Пример вывода:

```
Security has been disabled.
Please remove any SOLR_AUTH_TYPE or SOLR_AUTHENTICATION_OPTS configuration from solr.in.sh/solr.in.cmd.
```

Команды ZooKeeper

Скрипт */usr/lib/solr/bin/solr* поддерживает команды для взаимодействия с ZooKeeper. Например, команды `zk upconfig` и `zk downconfig` используются для загрузки/выгрузки конфигсетов Solr из ZooKeeper. Команды используются следующим образом:

```
$ /usr/lib/solr/bin/solr zk upconfig [<PARAMETERS>]
$ /usr/lib/solr/bin/solr zk downconfig [<PARAMETERS>]
```

Параметры команд

Команды: zk upconfig/zk downconfig

Параметр	Описание
-n <name>	Имя конфигсета, который должен быть загружен или выгружен из ZooKeeper. Все загруженные конфигсеты доступны в веб-интерфейсе Solr. Указание уже существующего имени перезаписывает конфигсет в ZooKeeper
-d <configset_dir>	Путь к директории конфигсета для загрузки/выгрузки. Директория конфигсета должна содержать поддиректорию *conf* с файлом *solrconfig.xml* внутри
-z <zk_host>	Опциональная строка подключения к ZooKeeper. Можно не указывать, если строка подключения к ZooKeeper не изменялась вручную

> **ПРИМЕЧАНИЕ**
>
> Загрузка конфигсета в ZooKeeper не обновляет коллекции автоматически. При обновлении конфигсета, который уже используется коллекцией, необходимо выполнить команду RELOAD для перезагрузки всех коллекций, использующих данный конфигсет.

Также существует несколько команд `zk …`, которые могут быть полезны для взаимодействия с ZooKeeper. Использование некоторых из них показано ниже:

```
$ /usr/lib/solr/bin/solr zk cp <src> <dest> [-r] [-z <zk_string>] ①
$ /usr/lib/solr/bin/solr zk rm <path> [-r] [-z <zk_string>] ②
$ /usr/lib/solr/bin/solr zk mv <src> <dst> [-z <zk_string>] ③
$ /usr/lib/solr/bin/solr zk ls <path> [-r] [-z <zk_string>] ④
$ /usr/lib/solr/bin/solr zk mkroot <path> [-z <zk_string>] ⑤
```

① Перенос данных между локальной файловой системой и ZooKeeper Znodes. Использование префикса `zk:` указывает на ZooKeeper-локацию, а `file:` указывает на локальную файловую систему. Одно из значений `<src>` или `<dest>` должно иметь префикс `zk:`.

② Удаление Znode из ZooKeeper. Поскольку `<path>` относится к ZooKeeper, указание префикса `zk:` не требуется.

③ Перемещение (переименование) Znode.

④ Вывод дочерних элементов Znode.

⑤ Создание новой Znode. Промежуточные Znodes создаются при необходимости.

Примеры

Загрузка конфигсета из ZooKepper

```
$ /usr/lib/solr/bin/solr zk downconfig -n test_collection -d ~/my_test_collection_config
```

Пример вывода:

```
Downloading configset test_collection from ZooKeeper at ka-adh-1.ru-central1.internal:2181,ka-adh-2.ru-central1.
internal:2181,ka-adh-3.ru-central1.internal:2181/Arenadata.Hadoop-8.solr.server to directory
/home/admin/my_test_collection_config/conf
```

Загрузка директории в ZooKeeper

```
$ /usr/lib/solr/bin/solr zk cp -r file:/home/admin/test_dir/ zk:/Arenadata.Hadoop-8.solr.server/test_upload
```

Пример вывода:

```
Copying from 'file:/home/admin/test_dir/' to 'zk:/Arenadata.Hadoop-8.solr.server/test_upload'. ZooKeeper at ka-adh-1.ru-
central1.internal:2181,ka-adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.internal:2181/Arenadata.Hadoop-8.solr.server
```

Удаление Znode с дочерними элементами

```
$ /usr/lib/solr/bin/solr zk rm -r zk:/Arenadata.Hadoop-8.solr.server/test_upload
```

Пример вывода:

```
Removing ZooKeeper node /Arenadata.Hadoop-8.solr.server/test_upload from ZooKeeper at ka-adh-1.ru-central1.internal:2181,ka-
adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.internal:2181/Arenadata.Hadoop-8.solr.server recurse: true
```

Команды экспорта данных

Используя команду `export`, вы можете экспортировать документы из Solr-коллекции в формате JSON или Javabin. Использование команды показано ниже:

```
$ /usr/lib/solr/bin/solr export [<PARAMETERS>]
```

Параметры команд

Команда: export

Параметр	Описание
-fields <f1,f2,fn>	Список полей для экспорта, разделенных запятыми. По умолчанию экспортируются все поля
-format <fmt>	Устанавливает формат экспортируемых документов (JSON или javabin). По умолчанию используется JSON
-limit <i>	Максимальное количество документов для экспорта. Значение по умолчанию — `100`. Используйте `-1` для экспорта всех документов без ограничений
-out	Указывает имя файла для сохранения экспортированных данных. По умолчанию используется шаблон `<collection-name>.<format>`
-url <collection_url>	Адрес URL, указывающий на коллекцию

Примеры

Экспорт всех данных коллекции

```
$ sudo bin/solr export -url http://ka-adh-1.ru-central1.internal:8983/solr/test_collection
```

Пример вывода:

```
NO: of shards : 2

Export complete for : test_collection_shard1_replica_n2, docs : 0

Export complete for : test_collection_shard2_replica_n8, docs : 1

Total Docs exported: 0. Time taken: 0secs
```

Выполнение этой команды создает в локальной файловой системе файл *test_collection.json*. В этом файле хранятся документы из запрошенной коллекции, например:

```
{"key1":"foo","key2":"bar","id":"71b51c87-142c-45a6-8ea0-82ae924fd3c4"}
```

Экспорт определенных полей в файл с кастомным именем

```
$ sudo bin/solr export -url http://ka-adh-1.ru-central1.internal:8983/solr/test_collection -fields "key1"
```

Пример вывода:

```
NO: of shards : 2
.
Export complete for : test_collection_shard1_replica_n2, docs : 0

Export complete for : test_collection_shard2_replica_n8, docs : 1

Total Docs exported: 0. Time taken: 0secs
```

Выполнение этой команды создает в локальной файловой системе файл *test_export_file.json*. В этом файле хранятся документы, содержащие только запрошенные поля, например:

```
{"key1":"foo"}
```

Обзор работы с индексами в Solr

Константин Алпашкин

Содержание

Назад к содержанию

- Инструмент Post
- Индексирование данных
- Обновление индексированных данных
- Удаление данных из индекса
- Поиск данных
  - Парсеры запросов Solr
  - Примеры поиска

Данная статья описывает процедуру индексирования в Solr. Индексирование — это процесс добавления нового содержимого в индекс Solr, что делает содержимое доступным для поиска. Также под индексированием стоит понимать изменение и удаление существующего содержимого.

Существует несколько способов доставки данных в Solr для индексирования:

- Загрузка XML/JSON/CSV-файла в теле запроса HTTP, отправленного на REST-эндпойнт сервера Solr. ADH-компоненты Solr Server поставляются с Post-инструментом, который облегчает загрузку документов через REST-эндпойнт. Этот метод используется в примерах в статье.
- Использование Solr Admin UI для загрузки документов через веб-интерфейс.
- Создание клиентского приложения, которое взаимодействует с Solr по HTTP, используя Client API. Например, библиотека SolrJ для приложений на базе JVM.
- Использование фреймворков, таких как Solr Cell, разработанных для эффективной загрузки данных в Solr.

Независимо от способа доставки данных в Solr существует общая структура для всех данных, загружаемых в индекс. Это должен быть документ, содержащий одно или несколько именованных полей, в каждом из которых хранятся данные. Структура документов определяется файлом схемы (schema file), однако Solr также может принимать документы, которые не соответствуют строгой структуре (schemaless mode). Обычно одно из полей выбирается уникальным, подобно первичному ключу в традиционных базах данных, хотя использование уникального ключа не является обязательным требованием Solr.

Помимо форматов JSON, XML и CSV, Solr может работать с форматами обогащенного текста (rich text format), например HTML, PDF и так далее. В большинстве случаев Solr автоматически определяет тип файла, извлекает данные и не требует указания типа файла вручную.

Больше информации о процедуре индексирования Solr доступно в документации Solr.

Инструмент Post

Каждый компонент Solr Server, развернутый в кластере ADH, поставляется с инструментом Post, который облегчает загрузку документов на сервер Solr. Для работы с инструментом используйте скрипт */usr/lib/solr/bin/post*, который доступен на хостах с компонентом Solr Server.

Ниже показан синтаксис запуска скрипта:

```
$ /usr/lib/solr/post -c <collection_name> [OPTIONS] <file|directory|url|-d ["...",...]>
```

Поддерживаемые параметры `OPTIONS` описаны в таблице ниже.

-c <collection_name>	Коллекция Solr для загрузки данных. Необходимо указать либо имя коллекции, либо URL-адрес
-url <URL>	URL REST-эндпойнта сервера Solr. Если значение не указано, URL генерируется автоматически, например *http://localhost:<solr_port>/solr/<collection_name>/update*. Параметры, указанные в URL, имеют приоритет над параметрами `-c`, `-host` и `-port`
-host <host_name>	Имя хоста или IP-адрес сервера Solr
-port <p>, -p	Номер порта, используемого для подключения
-user <user:pass>, -u <user:pass>	Учетные данные для Basic-аутентификации
-type <content-type>	Значение HTTP-заголовка Content-Type. По умолчанию используется `application/xml`
-filetypes <type1,...>	Список типов файлов, принимаемых Solr. По умолчанию Solr принимает файлы следующих типов: xml, json, csv, pdf, doc, docx, ppt, pptx, xls, xlsx, odt, odp, ods, ott, otp, ots, rtf, htm, html, txt, log
-out <yes|no>	Выводит ответ Solr-сервера в консоль

Несколько примеров запуска скрипта */usr/lib/solr/bin/post* приведены ниже.

```
/usr/lib/solr/bin/post -c demo_index_collection transactions.json
/usr/lib/solr/bin/post -c demo_index_collection test-*.xml
/usr/lib/solr/bin/post -c demo_index_collection -out yes test.csv
/usr/lib/solr/bin/post -c demo_index_collection ~/test/solr-docs
```

Индексирование данных

Далее описаны шаги для индексирования данных в Solr. Данная инструкция показывает, как индексировать простой JSON-документ с помощью инструмента Post.

1. Подключитесь к хосту ADH, на котором установлен компонент Solr Server.
2. Создайте тестовую коллекцию Solr с помощью следующей команды:

```
$ /usr/lib/solr/bin/solr create -c demo_index_collection -s 2 -rf 2
```

Команда создает новую коллекцию `demo_index_collection` с двумя шардами (shards) и фактором репликации, равным `2`. Коллекция создается на основе стандартной конфигурации `_default`, которая поддерживает бессхемный режим (schemaless mode).

Вывод:

```
Created collection 'demo_index_collection' with 2 shard(s), 2 replica(s) with config-set 'demo_index_collection'
```

РЕКОМЕНДАЦИЯ

Для создания новой коллекции можно также использовать веб-интерфейс Solr.

3. Создайте тестовый документ в формате JSON, как показано ниже. Можно создать аналогичный документ, используя форматы XML или CSV. Solr автоматически определит тип файла и обработает содержимое документа соответствующим образом.

```
{
  "id": 1,
  "txn_id": "1",
  "acc_id": 1001,
  "txn_value": 75.0,
  "txn_date": "2024-01-02",
  "comment": "The first transaction."
}
```

В данном примере поле `id` выступает в качестве первичного ключа, позволяющего идентифицировать документ среди других проиндексированных документов. Используя этот идентификатор, можно обновить или удалить документ в будущем. Поле `id` выбрано в качестве уникального ключа, так как это поле указано в файле схемы по умолчанию (определение `uniqueKey`). Если удалить поле `id` из документа выше, Solr сгенерирует идентификатор автоматически; однако в таком случае необходимо отслеживать сгенерированные идентификаторы, чтобы иметь возможность обращаться к документам в индексе в будущем.

4. Используя инструмент Post, загрузите тестовый документ для индексирования с помощью следующей команды:

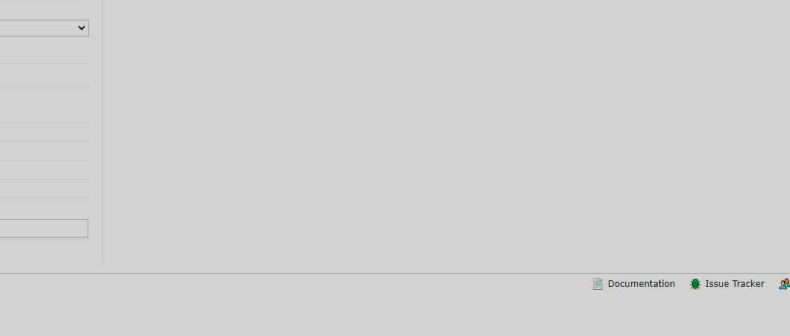
```
$ /usr/lib/solr/bin/post -c demo_index_collection transactions.json
```

Запуск скрипта выводит следующее:

```
java -classpath /usr/lib/solr/dist/solr-core-8.11.2.jar -Dauto=yes -Dc=demo_index_collection -Ddata=files org.apache.solr.util.
SimplePostTool transactions.json
SimplePostTool version 5.0.0
Posting files to [base] url http://localhost:8983/solr/demo_index_collection/update...
Entering auto mode. File endings considered are
xml,json,jsonl,csv,pdf,doc,docx,ppt,pptx,xls,xlsx,odt,odp,ods,ott,otp,ots,rtf,htm,html,txt,log
POSTing file transactions.json (application/json) to [base]/json/docs
1 files indexed.
COMMITting Solr index changes to http://localhost:8983/solr/demo_index_collection/update...
Time spent: 0:00:01.341
```

Чтобы убедиться, что Solr успешно проиндексировал документ, выполните следующие действия:

1. В Solr Admin UI выберите нужную коллекцию (`demo_index_collection` в данном примере).
2. Перейдите на страницу ***Query***.
3. Кликните ***Execute Query*** для выполнения поискового запроса с параметрами по умолчанию. Результаты запроса показаны ниже.



Результаты запроса

Обновление индексированных данных

Чтобы обновить проиндексированный документ, необходимо повторно загрузить измененную версию документа в Solr, убедившись, что уникальный ключ в обновленном документе соответствует документу, который нужно обновить. Если уникальный ключ не указан или указан некорректно, Solr создаст новый документ вместо обновления нужного документа. В стандартной Solr-схеме (схема, принадлежащая конфигурации `_default`) в качестве уникального ключа используется поле `id`.

Ниже описаны шаги, необходимые для обновления проиндексированного документа:

1. Создайте обновленную версию документа, который уже присутствует в индексе Solr. Например, содержимое файла *transactions_updated.json*:

```
{
  "id": 1,
  "txn_id": "1",
  "acc_id": 1001,
  "txn_value": 75.0,
  "txn_date": "2024-01-02",
  "comment": "Updated transaction description."
}
```

2. Загрузите обновленный документ в Solr с помощью инструмента Post:

```
$ /usr/lib/solr/bin/post -c demo_index_collection transactions_updated.json
```

Вывод:

```
java -classpath /usr/lib/solr/dist/solr-core-8.11.2.jar -Dauto=yes -Dc=demo_index_collection -Ddata=files org.apache.solr.util.
SimplePostTool transactions_updated.json
SimplePostTool version 5.0.0
Posting files to [base] url http://localhost:8983/solr/demo_index_collection/update...
Entering auto mode. File endings considered are
xml,json,jsonl,csv,pdf,doc,docx,ppt,pptx,xls,xlsx,odt,odp,ods,ott,otp,ots,rtf,htm,html,txt,log
POSTing file transactions_updated.json (application/json) to [base]/json/docs
1 files indexed.
COMMITting Solr index changes to http://localhost:8983/solr/demo_index_collection/update...
Time spent: 0:00:00.307
```

3. В Solr Admin UI запросите обновленный документ, указав `txn_id:1` в качестве строки запроса (поле `q`). Такой запрос эквивалентен SQL-запросу `SELECT * FROM <collection_name> WHERE txn_id=1;`. В ответе содержатся обновленные поля документа, как показано в примере ниже.

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":8,
    "params":{
      "q":"txn_id:1",
      "indent":"true",
      "q.op":"OR",
      "_":"1723478346726"}},
  "response":{"numFound":1,"start":0,"maxScore":1.0,"numFoundExact":true,"docs":[
      {
        "id":"1",
        "txn_id":[1],
        "acc_id":[1001],
        "txn_value":[75.0],
        "txn_date":["2024-01-02T00:00:00Z"],
        "comment":["Updated transaction description."],
        "_version_":1807198162248531968}]
  }}
```

Удаление данных из индекса

Для удаления данных из индекса Solr необходимо отправить delete-команду на REST-эндпойнт обновления, указав уникальный ключ документа либо строку запроса для удаления нескольких документов. Например, чтобы удалить документ с полем `id=1`, используйте команду:

```
$ {solr_root}/bin/post -c {test_collection} -d "<delete><id>1</id></delete>"
```

Для удаления одного или нескольких документов с помощью языка запросов Solr используйте следующий синтаксис:

```
$ /usr/lib/solr/bin/post -c demo_index_collection -d "<delete><query>txn_id:1</query></delete>"
```

Команда для удаления всех документов из коллекции выглядит следующим образом:

```
$ /usr/lib/solr/bin/post -c demo_index_collection -d "<delete><query>*:*</query></delete>"
```

Поиск данных

Поисковые запросы в Solr являются обычными HTTP-запросами, в которых параметры поиска передаются в виде query-параметров URL. Для отправки поискового запроса можно использовать HTTP-клиенты, такие как curl, wget, Postman, или воспользоваться инструментом Post. Пример URL для отправки поискового запроса показан ниже:

```
http://ka-adh-1.ru-central1.internal:8983/solr/demo_index_collection/select?indent=true&q.op=AND&q=*%3A*
```

Где `ka-adh-1.ru-central1.internal` — имя хоста с активным компонентом Solr Server.

Ответ Solr-сервера на такой запрос выглядит следующим образом:

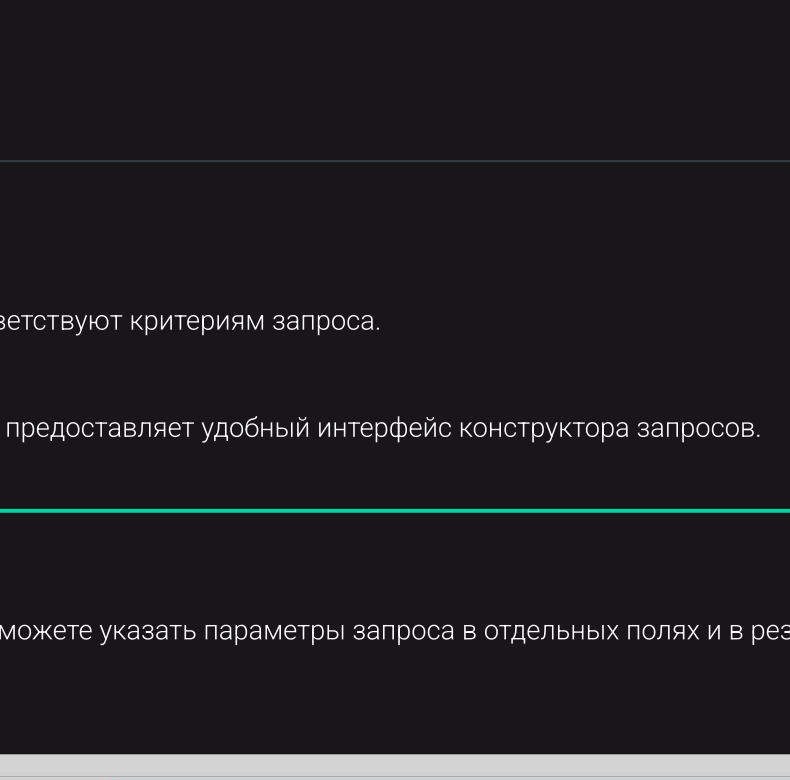
```
{
    "responseHeader": {
        "zkConnected": true,
        "status": 0,
        "QTime": 6,
        "params": {
            "q": "*:*",
            "indent": "true"
        }
    },
    "response": { ①
        "numFound": 1,
        "start": 0,
        "maxScore": 1.0,
        "numFoundExact": true,
        "docs": [ ②
            {
                "id": "1",
                "txn_id": [
                    1
                ],
                "acc_id": [
                    1001
                ],
                "txn_value": [
                    75.0
                ],
                "txn_date": [
                    "2024-01-02T00:00:00Z"
                ],
                "comment": [
                    "Updated transaction description."
                ],
                "_version_": 1807199238892814336
            }
        ]
    }
}
```

① Раздел с результатами запроса.

② Массив найденных документов, которые соответствуют критериям запроса.

Также вы можете использовать Solr Admin UI, который предоставляет удобный интерфейс конструктора запросов.

Используя интерфейс конструктора запросов, вы можете указать параметры запроса в отдельных полях и в результате получить готовый URL-адрес запроса, как показано на рисунке ниже.



URL запроса

Парсеры запросов Solr

Когда поисковый запрос поступает в Solr, запрос обрабатывается специальным плагином (request handler), который определяет дальнейшую логику обработки запроса. Затем обработчик запроса активирует парсер запросов (query parser), который интерпретирует параметры полученного запроса. Solr поддерживает несколько парсеров, перечисленных ниже. Перейдя по ссылке, можно получить более подробную информацию об использовании каждого парсера.

- Standard Query Parser. Парсер запросов по умолчанию. Основным недостатком является неспособность парсера корректно реагировать на синтаксические ошибки.
- DisMax. Позволяет обрабатывать простые фразы (без сложного синтаксиса) и искать термины по нескольким полям, используя механизм весов (weighting), основанный на значимости каждого поля.
- Extended DisMax (eDismax). Расширенная версия парсера DisMax, предоставляющая такие возможности, как частичное экранирование, расширенная обработка стоп-слов и так далее.
- Другие парсеры, например Block Join Query Parser, Boolean Query Parser, Boost Query Parser и так далее.

Примеры поиска

Следующие примеры демонстрируют базовые возможности поиска с использованием стандартного парсера запросов (Standard Query Parser). Подробная информация о параметрах поиска, которые поддерживаются различными парсерами, доступна в документации Solr.

Поиск всех документов с сортировкой

Следующий пример поискового запроса возвращает все поля всех документов из коллекции, сортируя результаты по полю `txn_id` в порядке возрастания.

Параметры запроса

Параметр	Значение
q	*:* Данная строка запроса по умолчанию эквивалентна SQL-запросу `SELECT * FROM <collection_name>`
sort	txn_id asc

Пример URL запроса:

```
http://ka-adh-1.ru-central1.internal:8983/solr/demo_index_collection/select?indent=true&q.op=OR&q=*%3A*&sort=txn_id%20asc
```

Ответ Solr-сервера:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":4,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR",
      "sort":"txn_id asc",
      "_":"1723495198651"}},
  "response":{"numFound":3,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"1",
        "txn_id":[1],
        "acc_id":[1001],
        "txn_value":[75.0],
        "txn_date":["2024-01-02T00:00:00Z"],
        "comment":["The first transaction."],
        "_version_":1807214831002976256},
      {
        "id":"3",
        "txn_id":[5],
        "acc_id":[1003],
        "txn_value":[105.0],
        "txn_date":["2024-02-02T00:00:00Z"],
        "comment":["Yet another transaction."],
        "_version_":1807215090454233088},
      {
        "id":"2",
        "txn_id":[256],
        "acc_id":[1002],
        "txn_value":[55.0],
        "txn_date":["2024-02-03T00:00:00Z"],
        "comment":["Another transaction."],
        "_version_":1807214995052691456}]
  }}
```

Получение значений определенных полей с фильтрацией

Следующий пример поискового запроса возвращает только определенные поля (`fl=txn_id,acc_id,txn_value,comment`) документов, у которых значение `txn_date` соответствует указанной дате. Кроме того, выполняется фильтрация по полю `comment`, оставляя в выборке лишь те результаты, в которых содержится строка `first`.

Параметры запроса

Параметр	Значение
q	txn_date:"2024-01-02T00:00:00Z"
fl	txn_id,acc_id,txn_date,comment
fq	comment:\"*first*\"

Пример URL запроса:

```
http://ka-adh-1.ru-central1.internal:8983/solr/demo_index_collection/select?
fl=txn_id%2Cacc_id%2Ctxn_value%2Ccomment&fq=comment%3A%22*first1*%22&indent=true&q.op=OR&q=txn_date%3A%222024-01-
02T00%3A00%3A00Z%22
```

Ответ Solr-сервера:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":5,
    "params":{
      "q":"txn_date:\"2024-01-02T00:00:00Z\"",
      "indent":"true",
      "fl":"txn_id,acc_id,txn_value,comment",
      "q.op":"OR",
      "fq":"comment:\"*first*\"",
      "_":"1723492286745"}},
  "response":{"numFound":1,"start":0,"maxScore":1.0,"numFoundExact":true,"docs":[
      {
        "txn_id":[1],
        "acc_id":[1001],
        "txn_value":[75.0],
        "comment":["The first transaction."]}]
  }}
```

Комбинация условий поиска

Используя булевы операторы, можно создавать сложную логику запросов и комбинировать несколько условий поиска. Например, следующий запрос выполняет поиск документов, у которых в поле `comment` содержится строка `transaction`, а поле `txn_id` равно `1`.

Параметры запроса

Параметр	Значение
q	comment:transaction AND txn_id:1

Пример URL запроса:

```
http://ka-adh-1.ru-central1.internal:8983/solr/demo_index_collection/select?indent=true&q.
op=OR&q=comment%3Atransaction%20AND%20txn_id%3A1
```

Ответ Solr-сервера:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":4,
    "params":{
      "q":"comment:transaction AND txn_id:1",
      "indent":"true",
      "q.op":"OR",
      "_":"1723492286745"}},
  "response":{"numFound":1,"start":0,"maxScore":1.1307646,"numFoundExact":true,"docs":[
      {
        "id":"1",
        "txn_id":[1],
        "acc_id":[1001],
        "txn_value":[75.0],
        "txn_date":["2024-01-02T00:00:00Z"],
        "comment":["The first transaction."],
        "_version_":1807214831002976256}]
  }}
```

Поиск по диапазону

Следующий запрос возвращает документы, у которых значение поля `txn_value` попадает в диапазон 100-200.

Параметры запроса

Параметр	Значение
q	txn_value:[100 TO 200]

Пример URL запроса:

```
http://ka-adh-1.ru-central1.internal:8983/solr/demo_index_collection/select?fl=txn_id%2Ctxn_value%2Cacc_id&indent=true&q.
op=AND&q=txn_value%3A%5B100%20TO%20200%5D
```

Ответ Solr-сервера:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":8,
    "params":{
      "q":"txn_value:[100 TO 200]",
      "indent":"true",
      "fl":"txn_id,txn_value,acc_id",
      "q.op":"AND",
      "_":"1723539053709"}},
  "response":{"numFound":1,"start":0,"maxScore":1.0,"numFoundExact":true,"docs":[
      {
        "txn_id":[5],
        "acc_id":[1003],
        "txn_value":[105.0]}]
  }}
```

Размер и формат ответа Solr-сервера

Следующий запрос возвращает первые три документа, которые соответствуют критериям поиска. Также в запросе указано, что сервер Solr должен вернуть ответ в формате XML.

Параметры запроса

Параметр	Значение
q	*:*
rows	3
wt	xml

Пример URL запроса:

```
http://ka-adh-1.ru-central1.internal:8983/solr/demo_index_collection/select?indent=true&q.op=OR&q=*%3A*&rows=3&start=0&wt=xml
```

Ответ Solr-сервера:

```
<?xml version="1.0" encoding="UTF-8"?>
<response>

<lst name="responseHeader">
  <bool name="zkConnected">true</bool>
  <int name="status">0</int>
  <int name="QTime">4</int>
  <lst name="params">
    <str name="q">*:*</str>
    <str name="indent">true</str>
    <str name="start">0</str>
    <str name="q.op">OR</str>
    <str name="rows">3</str>
    <str name="wt">xml</str>
    <str name="_">1723492286745</str>
  </lst>
</lst>
<result name="response" numFound="3" start="0" maxScore="1.0" numFoundExact="true">
  <doc>
    <str name="id">2</str>
    <arr name="txn_id">
      <long>256</long>
    </arr>
    <arr name="acc_id">
      <long>1002</long>
    </arr>
    <arr name="txn_value">
      <double>55.0</double>
    </arr>
    <arr name="txn_date">
      <date>2024-02-03T00:00:00Z</date>
    </arr>
    <arr name="comment">
      <str>Another transaction.</str>
    </arr>
    <long name="_version_">1807214995052691456</long></doc>
  <doc>
    <str name="id">3</str>
    <arr name="txn_id">
      <long>5</long>
    </arr>
    <arr name="acc_id">
      <long>1003</long>
    </arr>
    <arr name="txn_value">
      <double>105.0</double>
    </arr>
    <arr name="txn_date">
      <date>2024-02-02T00:00:00Z</date>
    </arr>
    <arr name="comment">
      <str>Yet another transaction.</str>
    </arr>
    <long name="_version_">1807215090454233088</long></doc>
  <doc>
    <str name="id">1</str>
    <arr name="txn_id">
      <long>1</long>
    </arr>
    <arr name="acc_id">
      <long>1001</long>
    </arr>
    <arr name="txn_value">
      <double>75.0</double>
    </arr>
    <arr name="txn_date">
      <date>2024-01-02T00:00:00Z</date>
    </arr>
    <arr name="comment">
      <str>The first transaction.</str>
    </arr>
    <long name="_version_">1807214831002976256</long></doc>
</result>
</response>
```

Индексирование вложенных документов в Solr

Константин Алпашкин

Содержание

Назад к содержанию

- Обзор
  - Конфигурация схемы
  - Анонимные вложенные документы
- Индексирование вложенных документов
- Поиск по вложенным данным
  - Child document transformer
  - Block join query parsers

Обзор

В Solr вложенные документы — это обычные документы, которые были проиндексированы в составе другого (родительского) документа. Например, несколько комментариев, которые относятся к одной и той же публикации в блоге, могут храниться в виде объектов, вложенных в документ публикации. Ниже приведен пример документа, моделирующего такие отношения между родительскими и вложенными объектами:

```
[
  {
    "post_id":"blog_post0001",
    "nodeType":"post",
    "comments":[
      {
        "comment_id":"comment_1",
        "nodeType":"comment",
        "content":"foo bar"
      },
      {
        "comment_id":"comment_2",
        "nodeType":"comment",
        "content":"buzz"
      }
    ]
  }, {
    "post_id":"blog_post0002",
    "nodeType":"post",
    "comments":[
      {
        "comment_id":"comment_3",
        "nodeType":"comment",
        "content":"Some text"
      }
    ]
  }
]
```

Использование вложенных документов позволяет увеличить скорость выполнения запросов, поскольку отношения "родитель-потомок" между документами уже присутствуют в индексе и их не нужно вычислять при каждом обращении. Однако структуры с вложенными объектами менее гибкие, чем операции соединения при запросе (query-time joins), поскольку они накладывают строгие правила, которые могут быть неприемлемы для некоторых приложений.

Вложенные документы могут быть проиндексированы в форматах XML, JSON или SolrJ. Независимо от формата, каждый документ должен содержать поле, идентифицирующее документ как родительский или дочерний. Это поле может иметь любое имя, за исключением зарезервированных полей, определенных в схеме. В приведенном выше примере этой цели служит поле `nodeType` — оно содержит значения `post` или `comment`. При поиске по вложенным документам это поле используется в качестве входного параметра для семейства парсеров block join query.

Конфигурация схемы

Для работы с вложенными документами необходимо наличие в файле схемы поля с именем `_root_`, которое выглядит следующим образом:

```
<field name="_root_" type="string" indexed="true" stored="false" docValues="false" />
```

Также для корректной работы трансформера child document transformer в файле схемы должно быть объявлено поле `_nest_path_`, как показано ниже:

```
<fieldType name="_nest_path_" class="solr.NestPathField" />
<field name="_nest_path_" type="_nest_path_" />
```

По умолчанию данные поля присутствуют в стандартном файле схемы и будут унаследованы схемами, созданными на основе стандартной.

Анонимные вложенные документы

Иногда можно встретить следующий синтаксис для объявления вложенных анонимных документов:

```
[
  {
    "post_id":"blog_post0001",
    "nodeType":"post",
    "_childDocuments_":[
      {
        "comment_id":"comment_1",
        "nodeType":"comment",
        "content":"foo bar"
      },
      {
        "comment_id":"comment_2",
        "nodeType":"comment",
        "content":"buzz"
      }
    ]
  }
]
```

Обратите внимание на поле `_childDocuments_` с массивом документов. Это явное указание для Solr, что элементы в данном массиве — дочерние по отношению к текущему документу. Такой подход был распространен в старых версиях Solr и теперь устарел, однако он по-прежнему работает при одноуровневой вложенности.

Индексирование вложенных документов

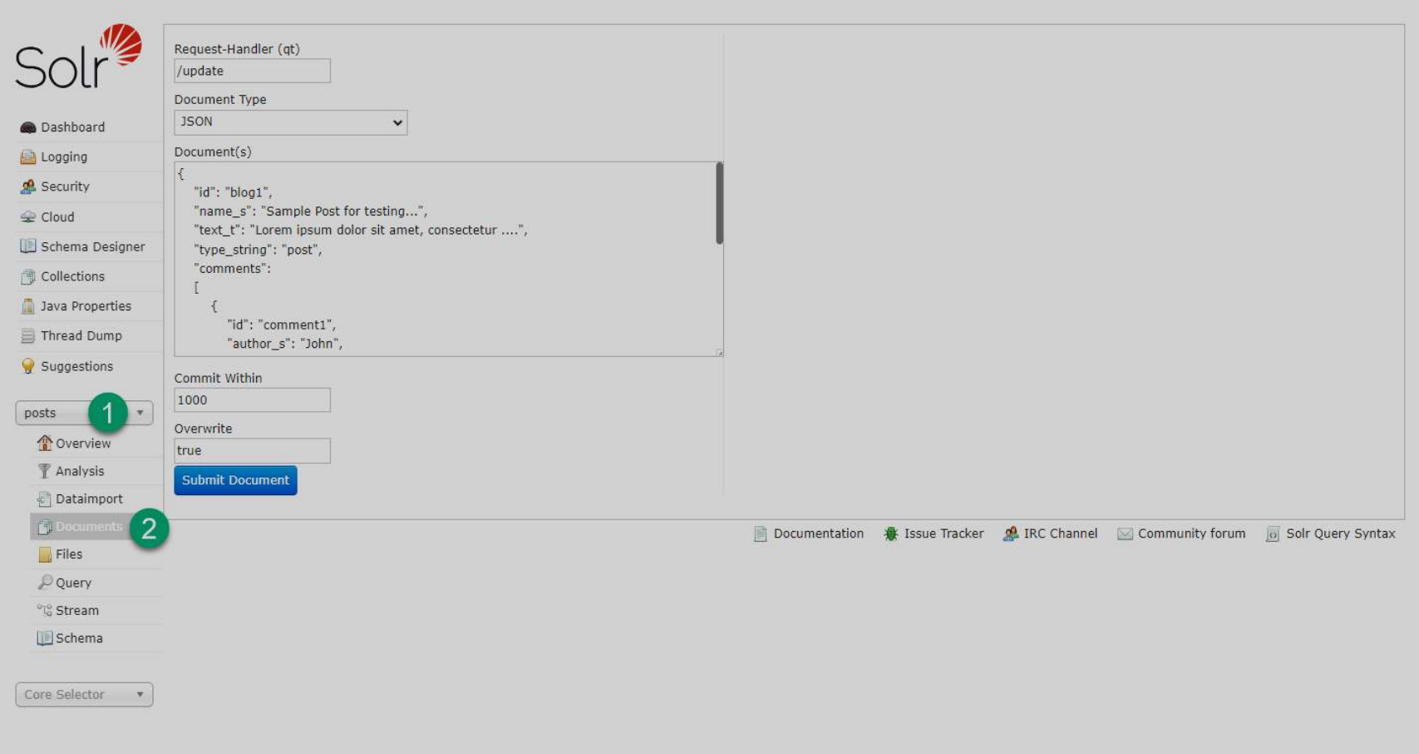
В следующем примере описан процесс индексирования Solr-документа с вложенными элементами.

1. Откройте Solr Admin UI. Актуальную ссылку на веб-интерфейс можно найти в ADCM (***Clusters* → *<clusterName>* → *Services* → *Solr* → *Info***).

2. Перейдите на страницу ***Collections*** и создайте тестовую коллекцию, используя параметры в таблице.

Параметр	Описание
name	posts
config set	_default * Используя этот параметр, новая коллекция наследует конфигурацию дефолтной схемы, включая базовые функции для работы с вложенными документами
numShards	1
replicationFactor	1

3. Выберите новую коллекцию из списка и перейдите на страницу ***Documents***, как показано на рисунке.



Request-Handler (qt): /update

Document Type: JSON

Document(s):

```
{
    "id": "blog1",
    "name_s": "Sample Post for testing...",
    "text_t": "Lorem ipsum dolor sit amet, consectetur ....",
    "type_string": "post",
    "comments":
    [
        {
            "id": "comment1",
            "author_s": "John",
```

Commit Within: 1000

Overwrite: true

Страница Documents

4. Загрузите в Solr следующие тестовые документы. Для этого поочередно вставьте содержимое документа в поле ***Document(s)*** и кликните ***Submit Document***.

post1.json

```
{
    "id": "post1",
    "name_s": "Sample Post for testing...",
    "text_t": "Lorem ipsum dolor sit amet, consectetur ....",
    "content_type": "post",
    "comments":
    [
        {
            "id": "comment1",
            "author_s": "John",
            "text_t": "Nice post!",
            "rated_i": 5,
            "content_type": "comment"
        },
        {
            "id": "comment2",
            "author_s": "Ann",
            "text_t": "Not interesting post...",
            "rated_i": 3,
            "content_type": "comment"
        }
    ]
}
```

post2.json

```
{
    "id": "post2",
    "name_s": "Another demo post...",
    "text_t": "Foo Bar Buzz ....",
    "content_type": "post",
    "comments":
    [
        {
            "id": "comment3",
            "author_s": "Luke",
            "text_t": "I like this post!",
            "rated_i": 4,
            "content_type": "comment"
        }
    ]
}
```

ПРИМЕЧАНИЕ

Не используйте Post tool для загрузки вложенных документов, так как данный инструмент автоматически делает их JSON-структуру "плоской" при загрузке.

В качестве альтернативы пользовательскому интерфейсу вы можете использовать curl для загрузки тестовых документов в Solr. Пример команды приведен ниже.

Команда загрузки с помощью curl

```
curl -X POST -H 'Content-Type: application/json' 'http://ka-adh-1.ru-central1.internal:8983/solr/posts/update' --data-binary
'[
    {
        "id": "blog1",
        "name_s": "Sample Post for testing...",
        "text_t": "Lorem ipsum dolor sit amet, consectetur ....",
        "comments":
        [
            {
                "id": "comment1",
                "author_s": "John",
                "text_t": "Nice post!",
                "rated_i": 5
            },
            {
                "id": "comment2",
                "author_s": "Ann",
                "text_t": "Not interesting post...",
                "rated_i": 3
            }
        ]
    },
    {
        "id": "blog2",
        "name_s": "Another demo post...",
        "text_t": "Foo Bar Buzz ....",
        "comments":
        [
            {
                "id": "comment3",
                "author_s": "Luke",
                "text_t": "I like this post!",
                "rated_i": 4
            }
        ]
    }
]'
```

5. На странице ***Query*** кликните ***Execute Query***, не изменяя параметры поиска по умолчанию, чтобы получить все документы из индекса. Ответ Solr-сервера выглядит следующим образом:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR",
      "_":"1724336874767"}},
  "response":{"numFound":5,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"comment1",
        "author_s":"John",
        "text_t":"Nice post!",
        "rated_i":5,
        "content_type":["comment"],
        "_version_":1808098244634345472},
      {
        "id":"comment2",
        "author_s":"Ann",
        "text_t":"Not interesting post...",
        "rated_i":3,
        "content_type":["comment"],
        "_version_":1808098244634345472},
      {
        "id":"post1",
        "name_s":"Sample Post for testing...",
        "text_t":"Lorem ipsum dolor sit amet, consectetur ....",
        "content_type":["post"],
        "_version_":1808098244634345472},
      {
        "id":"comment3",
        "author_s":"Luke",
        "text_t":"I like this post!",
        "rated_i":4,
        "content_type":["comment"],
        "_version_":1808098259769491456},
      {
        "id":"post2",
        "name_s":"Another demo post...",
        "text_t":"Foo Bar Buzz ....",
        "content_type":["post"],
        "_version_":1808098259769491456}]
  }}
```

Обратите внимание, что в ответе сервера документы имеют "плоскую" структуру, т.е. дочерние документы сгруппированы над родительскими. Именно в таком виде Solr хранит документы с вложенными элементами под капотом.

Поиск по вложенным данным

По умолчанию документы, удовлетворяющие поисковому запросу, не включают вложенных элементов, даже если таковые присутствуют в индексе. Например, выполните поиск документа, используя строку запроса `q=id:post1`. URL-адрес для такого запроса выглядит следующим образом:

```
http://ka-adh-1.ru-central1.internal:8983/solr/posts/select?indent=true&q.op=OR&q=id%3Apost1
```

В ответе содержится один документ без вложенных объектов:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"id:post1",
      "indent":"true",
      "q.op":"OR",
      "_":"1724336874767"}},
  "response":{"numFound":1,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"post1",
        "name_s":"Sample Post for testing...",
        "text_t":"Lorem ipsum dolor sit amet, consectetur ....",
        "content_type":["post"],
        "_version_":1808098244634345472}]
  }}
```

Child document transformer

Для добавления дочерних документов в ответ Solr, используйте child document transformer. Следующий запрос содержит параметр `fl=*,[child]`, с помощью которого в ответ сервера попадают все поля найденных документов и дополнительно все их дочерние элементы.

Пример URL запроса:

```
http://ka-adh-1.ru-central1.internal:8983/solr/posts/select?fl=*%2C%5Bchild%5D&indent=true&q.op=OR&q=id%3Apost1
```

Ответ Solr:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"id:post1",
      "indent":"true",
      "fl":"*,[child]",
      "q.op":"OR",
      "_":"1724336874767"}},
  "response":{"numFound":1,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"post1",
        "name_s":"Sample Post for testing...",
        "text_t":"Lorem ipsum dolor sit amet, consectetur ....",
        "content_type":["post"],
        "_version_":1808098244634345472,
        "comments":[
          {
            "id":"comment1",
            "author_s":"John",
            "text_t":"Nice post!",
            "rated_i":5,
            "content_type":["comment"],
            "_version_":1808098244634345472},

          {
            "id":"comment2",
            "author_s":"Ann",
            "text_t":"Not interesting post...",
            "rated_i":3,
            "content_type":["comment"],
            "_version_":1808098244634345472}]}]
  }}
```

Block join query parsers

Существует два парсера типа block join query, которые позволяют индексировать и выполнять поиск по вложенным документам. К ним относятся:

- Block join children query parser. Используется для получения дочерних объектов документа, который удовлетворяет запросу.
- Block join parent parser. Находит один или несколько дочерних документов и возвращает их родителя.

Пример: block join children query parser

Следующий поисковый запрос задействует парсер block join children query:

```
q={!child of="content_type:post"}text_t:"Lorem"
```

Solr интерпретирует эту строку запроса следующим образом:

1. Найти все родительские документы, у которых в поле `text_t` содержится строка `Lorem`.
2. Получить дочерние объекты найденных документов и добавить их в ответ сервера.

Обратите внимание на выражение `of="content_type:post"`, которое называется Block Mask. Данное выражение сообщает серверу Solr, что родительскими документами следует считать те объекты, у которых `"content_type"="post"`.

URL для такого запроса выглядит следующим образом:

```
http://ka-adh-1.ru-central1.internal:8983/solr/posts/select?indent=true&q.
op=OR&q=q%3D%7B!child%20of%3D%22content_type%3Apost%22%7Dtext_t%3A%22Lorem%22
```

Ответ:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"q={!child of=\"content_type:post\"}text_t:\"Lorem\"",
      "indent":"true",
      "q.op":"OR",
      "_":"1724274693355"}},
  "response":{"numFound":2,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"comment1",
        "author_s":"John",
        "text_t":"Nice post!",
        "rated_i":5,
        "content_type":["comment"],
        "_version_":1808018337579401216},
      {
        "id":"comment2",
        "author_s":"Ann",
        "text_t":"Not interesting post...",
        "rated_i":3,
        "content_type":["comment"],
        "_version_":1808018337579401216}]
  }}
```

Пример: block join parent query parser

Следующий поисковый запрос задействует парсер block join parent query:

```
q={!parent which="content_type:post"}author_s:John
```

Solr интерпретирует эту строку запроса следующим образом:

1. Найти вложенные документы, у которых поле `author_s` равно `John`.
2. Получить родителя найденного документа и вернуть его в ответе сервера.

Обратите внимание на выражение `which="content_type:post"`, которое называется Block Mask. Данное выражение сообщает серверу Solr, что родительскими документами следует считать те объекты, у которых `"content_type"="post"`.

URL запроса выглядит следующим образом:

```
http://ka-adh-1.ru-central1.internal:8983/solr/posts/select?indent=true&q.
op=OR&q=q%3D%7B!parent%20which%3D%22content_type%3Apost%22%7Dauthor_s%3AJohn
```

Ответ:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"q={!parent which=\"content_type:post\"}author_s:John",
      "indent":"true",
      "q.op":"OR",
      "_":"1724260126428"}},
  "response":{"numFound":1,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"blog1",
        "name_s":"Sample Post for testing...",
        "text_t":"Lorem ipsum dolor sit amet, consectetur ....",
        "content_type":["post"],
        "_version_":1808018337579401216}]
  }}
```

Частичное обновление документов в Solr

Константин Алпашкин

Содержание

Назад к содержанию

- Обзор
- Атомарные обновления
  - Обновление вложенных документов
- Обновления типа in-place
- Метод оптимистичного параллелизма

Обзор

Solr предоставляет три основных метода для частичного обновления данных в индексе. Данная статья описывает использование каждого метода с примерами, а именно:

- Атомарные обновления (atomic updates). Позволяют изменять одно или несколько полей документа путем переиндексирования всего документа.
- Обновления типа in-place (in-place updates). Подвид атомарных обновлений, позволяющие модифицировать числовые поля без переиндексирования всего документа.
- Метод оптимистичного параллелизма (optimistic concurrency). Позволяет выполнять обновления по условию, используя версию документа.

В этой статье примеры операций обновления предназначены для изменения следующего тестового документа:

```
{
  "id": 1,
  "post_name": "Sample blog post...",
  "categories": ["leisure", "hobby"],
  "post_rank": 75.0,
  "post_date": "2024-01-02",
  "post_text": "Lorem ipsum dolor sit amet ...",
  "description": "A sample post for testing purposes"
}
```

Данный документ хранится в индексе Solr в коллекции `test_collection`. Информация о добавлении документа в индекс доступна в разделе Обзор работы с индексами в Solr.

Атомарные обновления

Данный метод позволяет изменять отдельные поля документа. Этот подход следует применять, когда скорость и частота изменений в индексе критична для клиентских приложений.

Solr поддерживает несколько модификаторов, которые используются для обновления отдельных полей документа. Эти модификаторы представлены ниже:

- `set` — устанавливает новое или перезаписывает существующее значение поля. Удаляет значение, если указан `null` или пустой список. Принимает одно значение или список.
- `add` — добавляет указанное значение(я) в поле типа `multiValued`. Принимает одно значение или список.
- `add-distinct` — добавляет указанное значение(я) в поле типа `multiValued`, если указанное значение не присутствует в индексе. Принимает одно значение или список.
- `remove` — удаляет все совпадения указанного значения из поля типа `multiValued`. Принимает одно значение или список.
- `removeregex` — удаляет все совпадения, соответствующие регулярному выражению, из поля типа `multiValued`. Принимает одно значение или список.
- `inc` — увеличивает числовое поле на заданное значение. Принимает одно числовое значение.

Чтобы обновить документ, укажите соответствующий модификатор в качестве значения обновляемого поля и загрузите документ в Solr. Следующий пример обновляет сразу несколько полей тестового документа:

```
{
  "id": 1,
  "post_name": {"set": "Updated post name ..."},
  "categories": {"add": ["science"]},
  "post_rank": {"inc": 5},
  "post_date": "2024-01-02",
  "post_text": "Lorem ipsum dolor sit amet ...",
  "description": {"removeregex": ".*testing.*"}
}
```

Команда загрузки с помощью curl

```
$ curl -X POST 'http://ka-adh-1.ru-central1.internal:8983/solr/test_collection/update?commit=true' -H 'Content-Type:
application/json' --data-binary '[{
  "id": 1,
  "post_name": {"set": "Updated post name ..."},
  "categories": {"add": ["science"]},
  "post_rank": {"inc": 5},
  "post_date": "2024-01-02",
  "post_text": "Lorem ipsum dolor sit amet ...",
  "description": {"removeregex": ".*testing.*"}
}]'
```

Когда Solr получает такой документ, он распознает выражения-модификаторы и обновляет соответствующие поля документа. После обновления документ имеет следующий вид:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR",
      "_":"1726172218281"}},
  "response":{"numFound":1,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"1",
        "post_name":["Updated post name ..."],
        "categories":["leisure",
          "hobby",
          "science"],
        "post_rank":[80.0],
        "post_date":["2024-01-02T00:00:00Z"],
        "post_text":["Lorem ipsum dolor sit amet ..."],
        "_version_":1810022770405277696}]
  }}
```

Обновление вложенных документов

Solr позволяет изменять, добавлять и удалять вложенные документы с помощью операций атомарного обновления. Процесс очень похож на обновление обычных (невложенных) документов, а основные особенности, которые следует учитывать, заключаются в следующем:

- Каждая операция по обновлению вложенного объекта должна быть направлена в нужный шард Solr — тот, в котором хранится родительский документ. Поскольку Solr определяет целевой шард на основе ID документа, важно использовать ID корневого (root) документа, а не ID обновляемого вложенного документа. Для выполнения этого требования можно либо явно указать Solr-роутер (через параметр `_route_`), либо использовать compositeId router (используется по умолчанию) для направления операций обновления в нужный шард. Больше информации о правилах маршрутизации при обновлении вложенных документов доступно в документации Solr.
- При обновлении вложенных объектов необходимо проинформировать Solr, какой документ является родительским для обновляемого объекта. Это можно сделать, указав поле `_root_` в запросе на обновление.
- По умолчанию Solr пытается применить обновления ко всему дереву вложенных документов вместо того, чтобы изменять отдельные объекты. Это может приводить к дополнительной нагрузке.

Предположим, что имеется следующий документ с вложенными объектами-комментариями, который необходимо обновить:

```
{
    "id": "post2",
    "name_s": "Another demo post...",
    "text_t": "Foo Bar Buzz ....",
    "content_type": "post",
    "comments":
    [
        {
            "id": "post2!comment3",
            "author_s": "Luke",
            "text_t": "I like this post!",
            "rated_i": 4,
            "content_type": "comment"
        }
    ]
}
```

Например, чтобы увеличить числовое поле `rated_i`, используя атомарные обновления, необходимо загрузить следующий документ в Solr:

```
{
  "id": "post2!comment3", ①
  "_root_": "post2", ②
  "rated_i": { "inc": 1 }
}
```

① ID вложенного документа, который необходимо обновить. Обратите внимание, что ID является составным (состоит из частей, разделенных символом `!`). Использование составного ID позволяет стандартному compositeId router направить операцию обновления в соответствующий шард Solr.

② Поле `_root_` является обязательным. Оно информирует Solr о том, какой документ является родительским по отношению к обновляемому объекту. Если поле не указано, Solr отклонит операцию обновления.

Результат обновления:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR",
      "_":"1726172218281"}},
  "response":{"numFound":2,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"post2!comment3",
        "author_s":"Luke",
        "text_t":"I like this post!",
        "rated_i":5,
        "content_type":["comment"],
        "_version_":1810022942789074944},
      {
        "id":"post2",
        "name_s":"Another demo post...",
        "text_t":"Foo Bar Buzz ....",
        "content_type":["post"],
        "_version_":1810022942789074944}]
  }}
```

Пример: добавление вложенных документов

Загрузка следующего документа в Solr добавляет еще один вложенный объект к родительскому документу (`"id": "post2"`):

```
{
  "id": "post2",
  "comments": { "add": { "id": "post2!comment4",
                         "author_s": "Rick Sanchez",
                         "text_t": "Another added comment here..",
                         "rated_i": 5,
                         "content_type": "comment"
                       } }
}
```

Результат:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR",
      "_":"1726172218281"}},
  "response":{"numFound":3,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"post2!comment3",
        "author_s":"Luke",
        "text_t":"I like this post!",
        "rated_i":5,
        "content_type":["comment"],
        "_version_":1810023009958756352},
      {
        "id":"post2!comment4",
        "author_s":"Rick Sanchez",
        "text_t":"Another added comment here..",
        "rated_i":5,
        "content_type":["comment"],
        "_version_":1810023009958756352},
      {
        "id":"post2",
        "name_s":"Another demo post...",
        "text_t":"Foo Bar Buzz ....",
        "content_type":["post"],
        "_version_":1810023009958756352}]
  }}
```

Пример: удаление вложенных документов

Отправка следующего документа в Solr удаляет вложенный документ по ID:

```
{
    "id": "post2",
    "comments": {
        "remove": {
            "id": "post2!comment4"
        }
    }
}
```

Результат:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR",
      "_":"1726172218281"}},
  "response":{"numFound":2,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"post2!comment3",
        "author_s":"Luke",
        "text_t":"I like this post!",
        "rated_i":5,
        "content_type":["comment"],
        "_version_":1810023111907606528},
      {
        "id":"post2",
        "name_s":"Another demo post...",
        "text_t":"Foo Bar Buzz ....",
        "content_type":["post"],
        "_version_":1810023111907606528}]
  }}
```

Обновления типа in-place

Обновления типа in-place (in-place updates) являются подмножеством атомарных обновлений. Однако при обычном атомарном обновлении под капотом происходит переиндексация всего документа. При обновлениях типа in-place изменяются только необходимые поля, а остальная часть документа остается нетронутой. Таким образом, эффективность in-place обновлений не зависит от количества полей в обновляемом документе. Кроме внутренних особенностей имплементации, влияющих на производительность, между атомарными и in-place обновлениями нет особых функциональных различий.

Операция обновления может быть выполнена как обновление типа in-place, если обновляемые поля отвечают следующим требованиям:

- Обновляемые поля являются числовыми и имеют следующие свойства: `indexed="false"`, `stored="false"`, `multiValued="false"`, `docValues="true"`.
- В обновляемом документе присутствует поле `_version_`, которое имеет свойства: `indexed="false"`, `stored="false"`, `multiValued="false"`, `docValues="true"`.
- CopyField destination обновляемого поля (если таковые используются) является числовым и имеет следующие свойства: `indexed="false"`, `stored="false"`, `multiValued="false"`, `docValues="true"`.

Для обновлений типа in-place Solr поддерживает следующие модификаторы:

- `set` — устанавливает новое или перезаписывает существующее значение поля.
- `inc` — увеличивает числовое поле на определенное значение.

Чтобы обновить числовое поле документа с помощью метода in-place, соответствующее поле должно быть определено в схеме Solr. Например:

```
<field name="post_rank" type="float" indexed="false" stored="false" docValues="true"/>
```

Чтобы обновить тестовый документ, используя подход in-place, загрузите в Solr следующий документ:

```
{
    "id": 1,
    "post_rank": {
        "set": 100
    }
}
```

В этом случае документ будет изменен с помощью стратегии обновления in-place. Результаты операции обновления:

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR",
      "_":"1726172218281"}},
  "response":{"numFound":1,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"1",
        "post_name":["Updated post name ..."],
        "categories":["leisure",
          "hobby",
          "science"],
        "post_rank":[100.0],
        "post_date":["2024-01-02T00:00:00Z"],
        "post_text":["Lorem ipsum dolor sit amet ..."],
        "_version_":1810023165979525120}]
  }}
```

Метод оптимистичного параллелизма

Обновление документов с помощью метода оптимистичного параллелизма (optimistic concurrency) — это функция Solr, которая гарантирует, что один и тот же документ не будет изменен несколькими клиентскими приложениями параллельно. Эта функция активно использует поле `_version_` и ожидает, что данное поле присутствует в каждом документе в индексе Solr. По умолчанию схема Solr включает поле `_version_`, так что поле автоматически добавляется в каждый новый документ. При обновлении документа Solr сравнивает значение `_version_` из индекса со значением, предоставленным в запросе, тем самым гарантируя, что документ не был изменен другими клиентами.

Чтобы обновить документ с помощью метода оптимистичного параллелизма, клиент должен передать актуальное значение `_version_` вместе с документом. Поле `_version_` — это обычное поле Solr, которое можно запрашивать так же, как и любое другое поле, например:

```
$ curl -X GET 'http://ka-adh-1.ru-central1.internal:8983/solr/test_collection/query?
q=*:*&fl=id,post_name,_version_&omitHeader=true' -H 'Content-Type: application/json'
```

Ответ Solr-сервера:

```
{
  "response":{"numFound":1,"start":0,"numFoundExact":true,"docs":[
      {
        "id":"1",
        "post_name":["Sample blog post..."],
        "_version_":1809999244663193600}]
  }}
```

Вы можете передать версию документа в Solr двумя способами:

- Используя параметр запроса `_version_=<version>` в URL. Например:

```
$ curl -X POST 'http://ka-adh-1.ru-central1.internal:8983/solr/test_collection/update?_version_=123' -H 'Content-Type:
application/json' --data-binary '[{"id": 1, "updated_field": "updated value"}]'
```

- Указав значение `_version_` в обновленном документе. Например:

```
{
  "id": 1,
  "_version_": 123,
  "post_rank": {"set": 80}
}
```

Этот способ удобен, когда документы отправляются в Solr пакетом и для каждого документа нужно указать разные значения `_version_`.

Если Solr получает некорректную версию документа, он отклоняет запрос на обновление, возвращает HTTP-ответ 409 и следующую ошибку:

```
 "error":{
    "metadata":[
      "error-class","org.apache.solr.common.SolrException",
      "root-error-class","org.apache.solr.common.SolrException"],
    "msg":"version conflict for 1 expected=12345 actual=1809999244663193600",
    "code":409}
```

Если версия документа указана корректно, Solr обновляет необходимые поля и присваивает новую версию для обновленного документа. Следующий запрос обновляет тестовый документ, передавая в качестве параметра запроса актуальную версию документа.

```
$ curl -X POST 'http://ka-adh-1.ru-central1.internal:8983/solr/test_collection/update?
_version_=1810003184734699520&versions=true&omitHeader=true' -H 'Content-Type: application/json' --data-binary
'[{"id":1,"post_rank": {"set": 80}}]'
```

Ответ сервера:

```
{
  "adds":[
    "1",1810019385852559360]
}
```

РЕКОМЕНДАЦИЯ

Параметр запроса `versions=true` добавляет версию документа в каждый ответ Solr-сервера. Это может быть полезно, чтобы избежать лишних GET-запросов для получения информации о версии документа.

Переиндексирование данных в Solr

Константин Алпашкин

Содержание

Назад к содержанию

- Случаи, когда требуется переиндексирование
  - Обновление схемы
  - Обновление данных
  - Поврежденный индекс
  - Улучшение производительности
- Стратегии переиндексирования
  - Удаление всех документов
  - Индексирование в другую коллекцию
- Команда REINDEXCOLLECTION
  - Параметры команды
  - Примеры использования команды

Переиндексирование в Solr — это процесс удаления всего содержимого индекса Solr с последующей записью данных из источника данных (базы данных, файла или другого хранилища) для создания нового индекса. При выполнении переиндексирования важно, чтобы все документы и сегменты Lucene были удалены для избежания конфликтов при повторном воссоздании индекса.

В этой статье перечислены распространенные ситуации, когда переиндексирование может быть необходимо, а также описаны популярные подходы к выполнению переиндексирования. Основными вызовами при выполнении переиндексирования являются время простоя, а также влияние на производительность. Переиндексирование больших наборов данных — это длительный и ресурсоемкий процесс, который может привести к временному бездействию коллекции или некорректным результатам поиска. Такие техники, как использование алиасов, описанные в этой статье, могут помочь обойти эти проблемы.

Случаи, когда требуется переиндексирование

Ниже описаны наиболее распространенные кейсы, требующие полного переиндексирования коллекции. Для остальных же случаев может быть достаточно использования частичного обновления документов.

Обновление схемы

> Принимая решение о необходимости переиндексирования коллекции при обновлении Solr-схемы, следует руководствоваться следующим правилом. Если в схеме были изменены параметры, отвечающие за запись данных в индекс, переиндексирование необходимо. Если изменилась только логика выполнения поисковых запросов (например, вы обновили поведение аналайзера при выполнении поиска), переиндексирование не требуется (хоть и рекомендуется).

Обновление схемы Solr-коллекции (например, изменение типов полей, добавление новых полей или изменение логики анализа) почти всегда требует полного переиндексирования.

Это требование обусловлено архитектурой поискового движка Lucene. Сегменты Lucene, в которых хранятся "сырые" данные, являются неизменяемыми, то есть после записи сегмента на диск он не может быть изменен. Сегменты можно добавлять, объединять, удалять, но не изменять. Сегменты Lucene слабо связаны с Solr-схемой, поскольку схема Solr — это концепция только Solr. Файл схемы лишь указывает Solr, как данные должны быть записаны в индекс Lucene и как их запросить в дальнейшем. После записи данных в индекс Lucene схема Solr не управляет хранимыми структурами данных.

Обновление данных

Если исходные данные были значительно изменены (например, в результате массовых обновлений или исправлений), то более эффективным будет переиндексирование всего набора данных вместо применения инкрементных обновлений.

Поврежденный индекс

Если индекс Solr поврежден или некоторые документы проиндексированы неправильно, переиндексирование может помочь вернуть индекс в рабочее состояние.

Улучшение производительности

Изменения настроек индексирования (токенизация, фильтры, анализ) или оптимизаций могут потребовать переиндексирования для вступления в силу. Здесь также следует руководствоваться правилом: если обновляются настройки, связанные с записью данных в индекс, переиндексирование необходимо; если обновляется лишь логика выполнения запросов — переиндексирование не обязательно.

Стратегии переиндексирования

Существует две основные стратегии переиндексирования, описанные ниже. Обе стратегии позволяют гарантировать, что индекс Lucene полностью пуст, прежде чем в него будут повторно помещены данные.

Удаление всех документов

Первый подход предполагает ручное удаление всех документов из индекса Solr. Например, вы можете удалить все документы из коллекции, выполнив следующий запрос:

```
$ curl http://ka-adh-1.ru-central1.internal:8983/solr/test_collection/update?commit=true -H "Content-Type: application/json" --
data-binary '{"delete": { "query": "*:*" }}'
```

Где:

- `ka-adh-1.ru-central1.internal` — имя хоста, на котором запущен сервер Solr;
- `test_collection` — имя коллекции, которую нужно очистить.

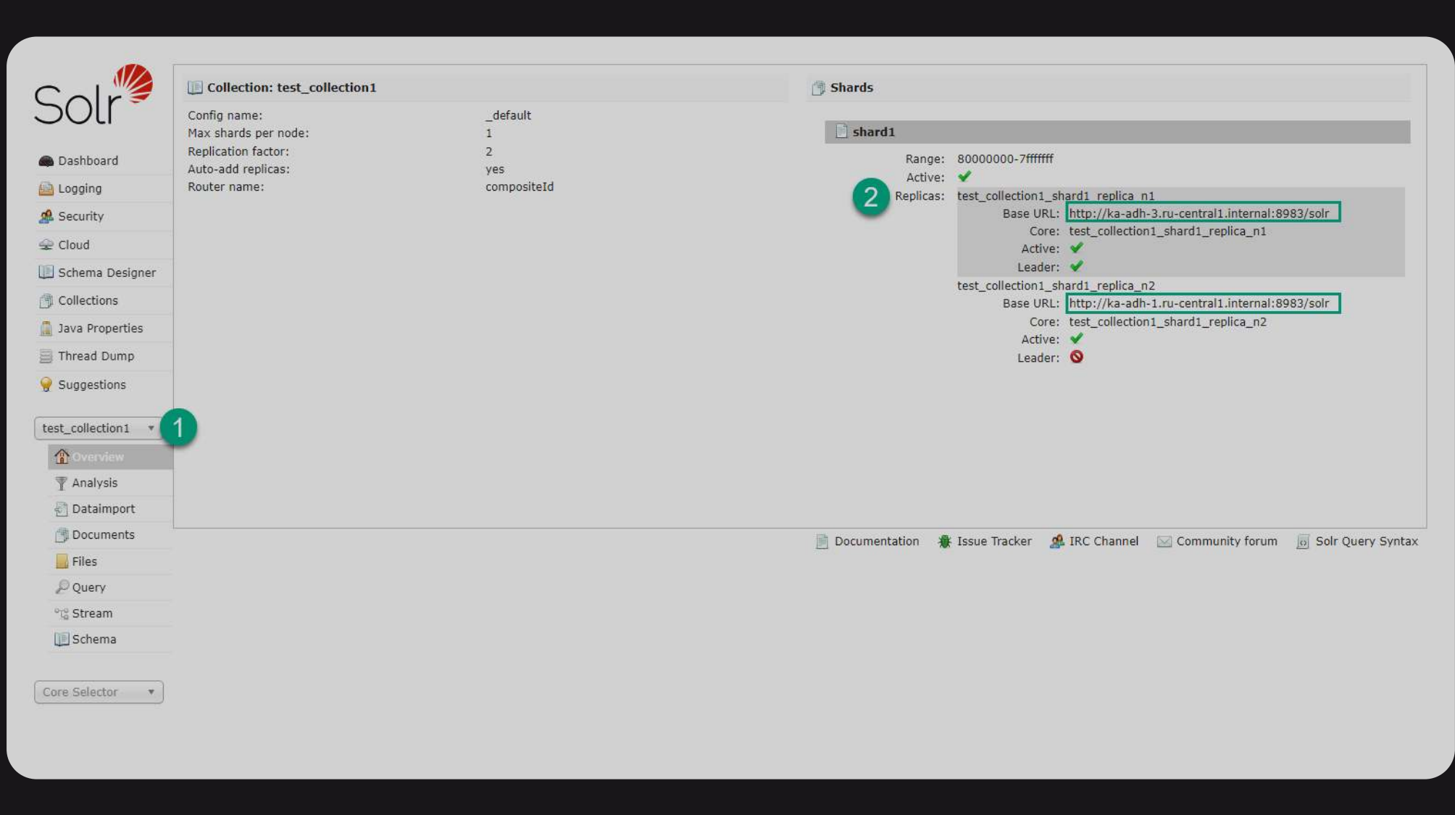
Пример ответа Solr-сервера:

```
{
  "responseHeader":{
    "rf":1,
    "status":0,
    "QTime":40}}
```

РЕКОМЕНДАЦИЯ

После операции удаления еще раз выполните поисковый запрос, чтобы убедиться, что все документы были удалены из коллекции. Это гарантирует, что сегменты индекса Lucene также были удалены.

Чтобы убедиться, что в индексе не осталось сегментов Lucene, проверьте директорию *{SOLR_HOME}/<collection_name>_*/data/index/* коллекции и убедитесь, что в ней нет файлов сегментов. Это необходимо выполнить для каждого шарда и каждой реплики вашего Solr-кластера. Чтобы найти шарды и реплики Lucene, принадлежащие данной коллекции, воспользуйтесь Solr web UI, как показано на рисунке.



Collection: test_collection1
Config name: _default
Max shards per node: 1
Replication factor: 2
Auto-add replicas: yes
Router name: compositeId

shard1
Range: 80000000-7fffffff
Active: ✓
Replicas: test_collection1_shard1_replica_n1
Base URL: http://ka-adh-3.ru-central1.internal:8983/solr
Core: test_collection1_shard1_replica_n1
Active: ✓
Leader: ✓
test_collection1_shard1_replica_n2
Base URL: http://ka-adh-1.ru-central1.internal:8983/solr
Core: test_collection1_shard1_replica_n2
Active: ✓
Leader: ⦸

Расположение шардов Solr

Затем подключитесь к каждому хосту, на котором хранятся шарды и реплики, и выполните команду:

```
$ ls -l /srv/solr/server/test_collection_*/data/index
```

Где:

- `/srv/solr/server/` — директория по умолчанию (`SOLR_HOME`), где Solr хранит данные коллекций;
- `test_collection` — имя коллекции.

Следующий вывод показывает, что данные коллекции были удалены и индекс пуст:

```
[admin@ka-adh-3 ~]$ ls -l /srv/solr/server/test_collection_*/data/index
total 4
-rw-r--r--. 1 solr solr 135 Oct 11 12:37 segments_7
-rw-r--r--. 1 solr solr   0 Oct 11 09:00 write.lock
```

Файлы *segments_N* и *write.lock* являются вспомогательными и не должны удаляться во время переиндексирования. Если директория *data/index/* вашей коллекции находится в таком состоянии, то ваша коллекция готова к переиндексированию.

ВАЖНО

Не удаляйте вручную файлы в директории *data/index/*, так как это неизбежно приведет к повреждению индекса.

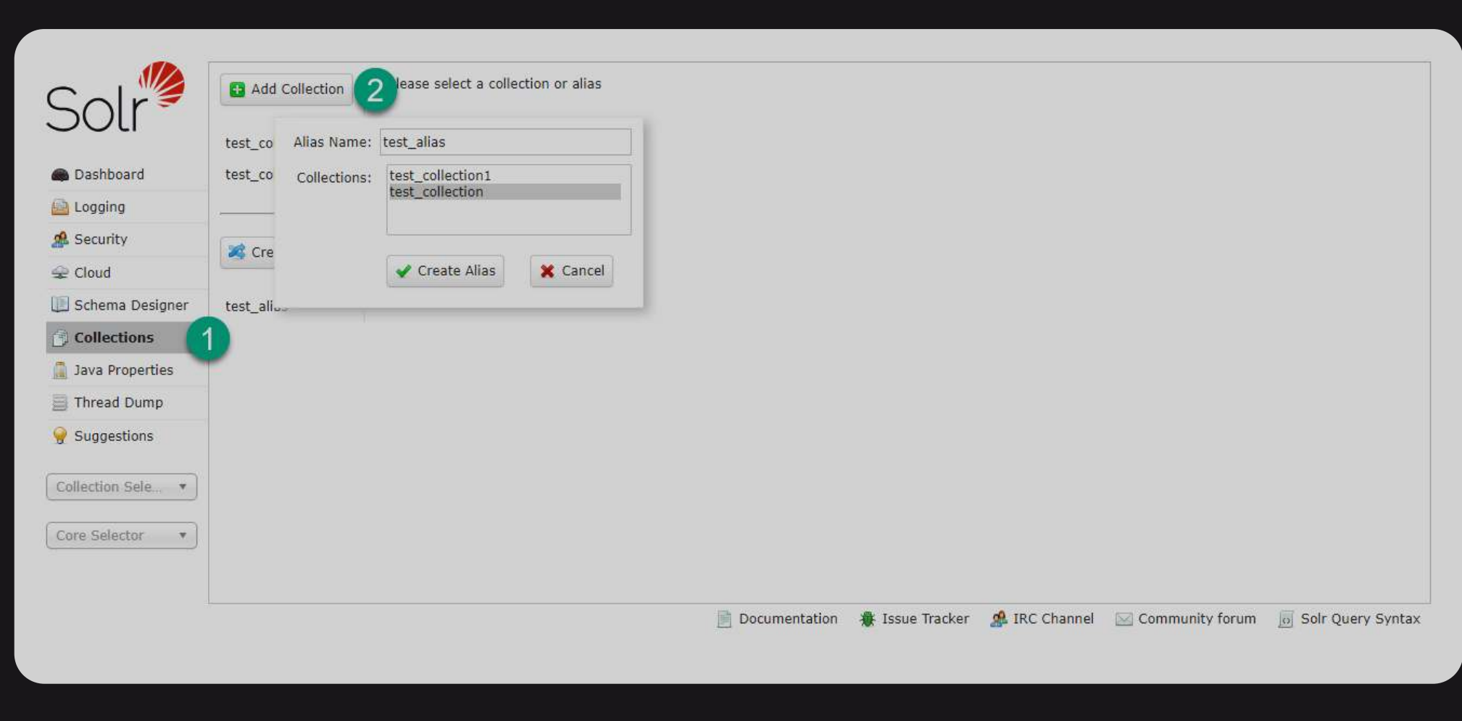
Основным недостатком этого подхода является возможное время простоя или некорректные результаты поиска при повторной записи данных в индекс. Следующая описанная стратегия помогает обойти эту проблему.

Индексирование в другую коллекцию

Другой подход к переиндексированию предполагает использование алиасов коллекций (collection alias), что позволяет перенаправлять запросы от клиентских приложений в другую коллекцию через алиас, избегая простоев.

При таком подходе процесс переиндексирования состоит из следующих шагов:

1. Создайте новую коллекцию. Старая коллекция, требующая переиндексирования, продолжает работать в обычном режиме.
2. Индексируйте свежие данные в новую коллекцию.
3. После индексирования создайте алиас, который будет перенаправлять клиентские запросы к новой коллекции. Вы можете создать алиас для коллекции с помощью Solr web UI, как показано на рисунке ниже.



Alias Name: test_alias
Collections: test_collection1
test_collection

Создание алиаса

4. Удалите старую коллекцию как ненужную.

Команда REINDEXCOLLECTION

Solr предоставляет команду REINDEXCOLLECTION, которая позволяет выполнить индексирование коллекции, используя данные из исходной коллекции.

ПРИМЕЧАНИЕ

При использовании команды `REINDEXCOLLECTION` могут теряться данные, так как команда копирует только данные со значением полей stored=true. Рекомендуется не удалять исходную коллекцию сразу после переиндексирования, а сохранить ее в качестве бэкапа на случай потери некоторых данных.

Чтобы запустить команду, отправьте GET-запрос на URL */admin/collections?action=REINDEXCOLLECTION&name=<collection_name>*. После запуска команды исходная коллекция переходит в read-only режим, чтобы обеспечить корректную обработку всех исходных документов. Для каждой исходной коллекции может быть запущена только одна операция `REINDEXCOLLECTION`. Вы можете принудительно завершить длительные, зависшие или неудачные операции с помощью опции abort, которая также удаляет частично скопированные документы. Если при копировании документов в целевую коллекцию произошел сбой, Solr удаляет временные и целевые коллекции, а также отключает режим read-only исходной коллекции.

Параметры команды

Команда `REINDEXCOLLECTION` принимает следующие параметры в виде параметров запроса URL.

Параметр	Описание
name	Обязательный параметр. Имя исходной коллекции или алиас, указывающий на нее
cmd	Указывает операцию, которую необходимо выполнить. Поддерживаются следующие операции: ■ `start` — запускает процесс переиндексирования (значение по умолчанию). ■ `abort` — прерывает запущенный или неудачный сеанс переиндексирования. Удаляет частичные результаты, если таковые имеются. ■ `status` — возвращает информацию о состоянии запущенного сеанса переиндексирования.
target	Имя целевой коллекции. Если не указано, Solr создает коллекцию с уникальным именем и после переиндексирования создает алиас, указывающий на вновь созданную коллекцию, тем самым "изолируя" исходную коллекцию от запросов на чтение и обновление
q	Опциональная строка запроса для выборки документов для переиндексирования. По умолчанию используется значение `*:*`
fl	Опциональный список полей для переиндексирования. По умолчанию используется значение `*`
rows	Указывает размер пакета для копирования документов. В зависимости от среднего размера документа большие размеры пакета могут вызывать проблемы с памятью. Значение по умолчанию — 100
configName	Опциональное имя конфигсета для целевой коллекции. По умолчанию Solr использует тот же конфигсет, что и исходная коллекции
removeSource	Если указано значение `true`, удаляет исходную коллекцию после переиндексирования

Примеры использования команды

Чтобы выполнить приведенные ниже примеры, создайте тестовую коллекцию и проиндексируйте тестовый документ, как описано в статье Обзор работы с индексами в Solr.

Пример: запуск REINDEXCOLLECTION с параметрами по умолчанию

Следующий запрос запускает процесс переиндексирования исходной коллекции с именем `test_collection`.

```
$ curl 'http://localhost:8983/solr/admin/collections?action=REINDEXCOLLECTION&name=test_collection'
```

Ответ Solr-сервера:

```
{
  "responseHeader":{
    "status":0,
    "QTime":6453},
  "reindexStatus":{
    "actualSourceCollection":"test_collection",
    "actualTargetCollection":".rx_test_collection_3",
    "alias":"test_collection -> .rx_test_collection_3",
    "checkpointCollection":".rx_ck_test_collection",
    "inputDocs":1,
    "phase":"done",
    "processedDocs":1,
    "state":"finished"}}
```

Ответ показывает, что Solr создал новую коллекцию `.rx_test_collection_3` и алиас `test_collection`, указывающий на новую коллекцию. Проверьте содержимое новой коллекции, явно указав ее имя в обход алиаса:

```
$ curl 'http://ka-adh-1.ru-central1.internal:8983/solr/.rx_test_collection_3/select?indent=true&q.op=OR&q=*%3A*'
```

Ответ содержит тестовый документ, скопированный из исходной коллекции.

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR"}},
  "response":{"numFound":1,"start":0,"numFoundExact":true,"docs":[
      {
        "txn_id":[1],
        "txn_date":["2024-01-02T00:00:00Z"],
        "txn_value":[75.0],
        "comment":["The first transaction."],
        "id":"1",
        "acc_id":[1001],
        "_version_":1812905933055983616}]
  }}
```

Исходная коллекция (`test_collection`) не удалена, однако теперь Solr направляет все запросы на чтение/обновление к новой коллекции через алиас.

Пример: реиндексирование определенных данных

Следующий запрос запускает процесс переиндексирования с указанием параметров выборки данных, подлежащих копированию в целевую коллекцию.

```
$ curl 'http://localhost:8983/solr/admin/collections?
action=REINDEXCOLLECTION&name=test_collection&q=txn_id:1&fl=id,txn_id,txn_value,acc_id&target=test_collection2'
```

Ответ Solr-сервера:

```
{
  "responseHeader":{
    "status":0,
    "QTime":6215},
  "reindexStatus":{
    "actualSourceCollection":"test_collection",
    "actualTargetCollection":"test_collection2",
    "checkpointCollection":".rx_ck_test_collection",
    "inputDocs":1,
    "phase":"done",
    "processedDocs":1,
    "state":"finished"}}
```

Проверьте содержимое целевой коллекции:

```
$ curl 'http://ka-adh-1.ru-central1.internal:8983/solr/test_collection2/select?indent=true&q.op=OR&q=*%3A*'
```

Ответ содержит тестовый документ с ограниченным набором полей, которые были скопированы из исходной коллекции, согласно параметрам выборки.

```
{
  "responseHeader":{
    "zkConnected":true,
    "status":0,
    "QTime":0,
    "params":{
      "q":"*:*",
      "indent":"true",
      "q.op":"OR"}},
  "response":{"numFound":1,"start":0,"numFoundExact":true,"docs":[
      {
        "txn_id":[1],
        "txn_value":[75.0],
        "id":"1",
        "acc_id":[1001],
        "_version_":1812908971102568448}]
  }}
```

Дедупликация в Solr

Константин Алпашкин

Содержание

Назад к содержанию

- Обзор
- Дедупликация по уникальному ключу
  - Автоматическая генерация ID
- Дедупликация с использованием сигнатуры
- Пример дедупликации
  - Шаг 1. Создание тестовой коллекции
  - Шаг 2. Изменение схемы Solr
  - Шаг 3. Изменение файла solrconfig.xml
  - Шаг 4. Отправка данных для индексации
- Ограничения дедупликации в режиме SolrCloud

Обзор

Дедупликация в Solr — это процесс выявления и удаления дублирующихся документов в индексе, чтобы гарантировать, что результаты поиска не содержат повторяющихся сущностей. Дублирующиеся записи в индексе Solr, которые представляют одну и ту же бизнес-сущность, приводят к нагромождению результатов поиска и ухудшают эффективность поисковых запросов.

В данной статье описаны два основных способа идентификации дубликатов в индексе Solr:

1. Дедупликация по уникальному ключу. Этот подход основан на предотвращении записи дублирующихся документов с помощью специального поля, которое выступает уникальным идентификатором для каждого документа в коллекции. Такой подход похож на использование первичных ключей (primary key) в традиционных базах данных.

2. Дедупликация с использованием сигнатуры. Данный метод предполагает вычисление сигнатуры (хеша) для каждого документа на этапе индексирования, а также сравнение этих хешей перед добавлением новых документов в коллекцию.

Более подробная информация о каждом подходе приведена ниже.

Дедупликация по уникальному ключу

Конфигурационный параметр `uniqueKey` определяет поле, по которому Solr может идентифицировать каждый документ в коллекции. Данный параметр позволяет присвоить уникальный идентификатор каждому документу, тем самым предотвращая попадание дубликатов в индекс. Если Solr индексирует документ со значением `uniqueKey`, которое уже присутствует в индексе, Solr перезаписывает старый документ на новый, а не индексирует его как отдельную запись.

Параметр `uniqueKey` указывается в схеме Solr и по умолчанию определяет поле `id` в качестве уникального ключа. По умолчанию определение уникального ключа в файле Solr-схемы имеет следующий вид:

```
<uniqueKey>id</uniqueKey>
```

Автоматическая генерация ID

При записи документов в индекс Solr без указания поля `id` (или любого другого поля, используемого в качестве `uniqueKey`) Solr автоматически генерирует UUID (универсальный уникальный идентификатор) для этого поля, чтобы идентифицировать документ среди других. Такой UUID уникален в рамках коллекции, однако при его генерации не учитывается содержимое других полей документа. При индексировании документов без указания `uniqueKey` вам может понадобиться сохранять автоматически генерируемые UUID, возвращаемые сервером Solr. В большинстве случаев ответственность за генерацию уникального ключа лежит на клиенте или источнике данных, поставляющем документы в Solr.

Дедупликация с использованием сигнатуры

Другой метод предотвращения дубликатов заключается в вычислении сигнатуры (хеша) для каждого документа на этапе индексирования с последующей проверкой индекса на наличие документа с таким хешем. Если сигнатура уже присутствует в индексе, Solr может либо перезаписать, либо пропустить такой документ. Вычисление сигнатуры можно настроить по одному или нескольким полям, а также указать произвольное поле документа, в котором будет храниться сигнатура документа.

Для реализации данного подхода используются обработчики запросов обновления (update request processors) и алгоритмы хеширования с низким числом коллизий (low collision algorithms) или алгоритмы с нечеткой логикой (fuzzy algorithms). Solr предоставляет "из коробки" следующие реализации класса `Signature`, который используется для генерации сигнатуры:

- MD5Signature — генерирует 128-битный хеш MD5.
- Lookup3Signature — генерирует 64-битный хеш. Процесс гораздо быстрее, чем у MD5, а также на выходе получается более короткая сигнатура, что оптимально с точки зрения индексирования.
- TextProfileSignature — реализация хеширования с нечеткой логикой для обнаружения дубликатов, которая лучше всего подходит для длинных текстов.

Пример дедупликации

В следующих шагах рассмотрен базовый сценарий дедупликации на основе сигнатур. Чтобы настроить процесс дедупликации с использованием этого подхода, необходимо редактирование файлов *solrconfig.xml* и *schema.xml*, как описано далее.

Шаг 1. Создание тестовой коллекции

Сначала создайте тестовую коллекцию, выполнив следующую команду на хосте с компонентом Solr Server:

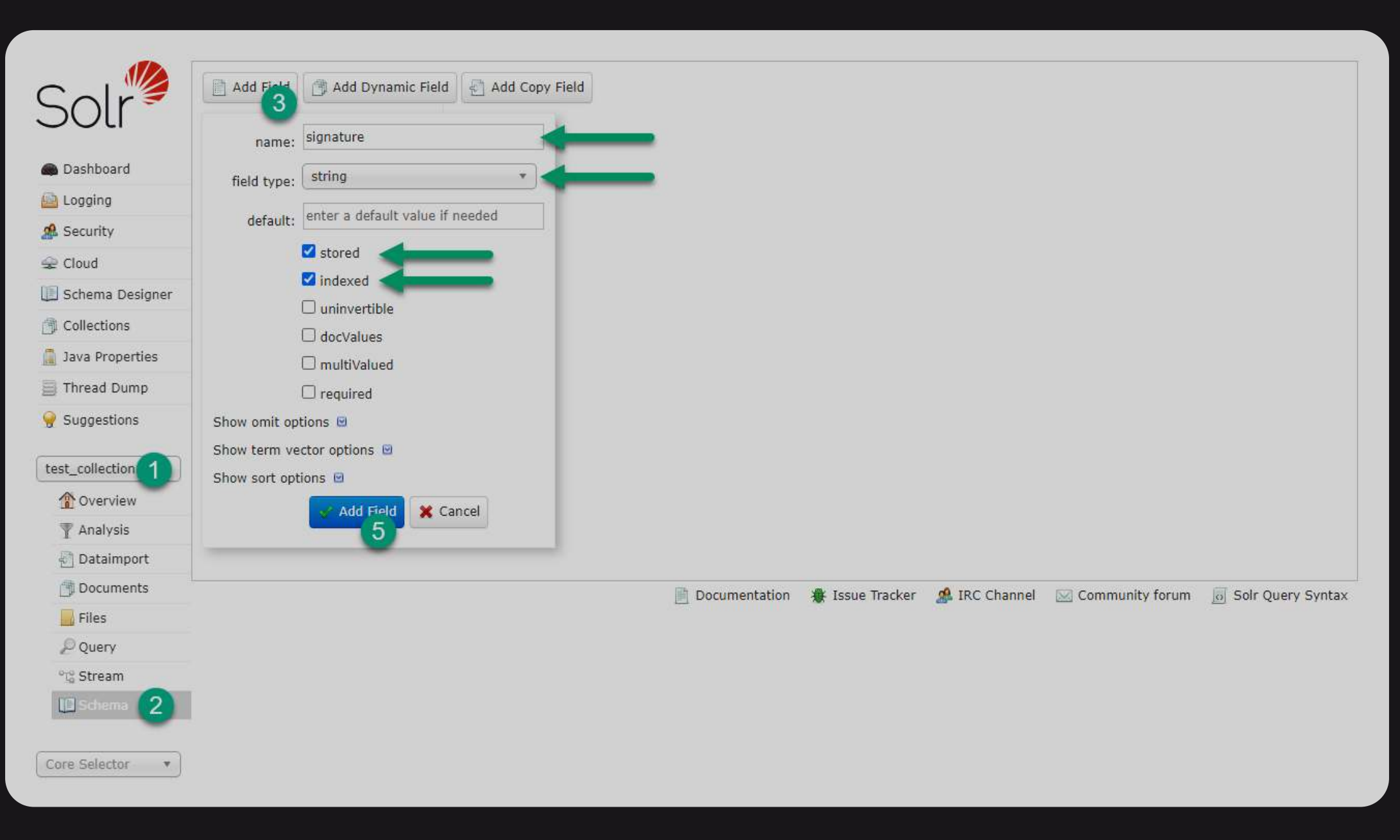
```
$ /usr/lib/solr/bin/solr create -c test_collection -s 1 -rf 1
```

Данная команда создает новую коллекцию (`test_collection`) с одним шардом (shard) и фактором репликации (replication factor), равным `1`. Коллекция создается с использованием стандартного конфигсета (configset) `_default`. Вывод команды:

```
WARNING: Using _default configset with data driven schema functionality. NOT RECOMMENDED for production use.
         To turn off: bin/solr config -c demo_collection_dedup -p 8983 -action set-user-property -property update.
autoCreateFields -value false
Created collection 'test_collection' with 1 shard(s), 1 replica(s) with config-set 'test_collection'
```

Шаг 2. Изменение схемы Solr

В Solr-схеме добавьте новое строковое поле (например, `signature`), в котором Solr будет хранить сигнатуры каждого документа. Поле должно иметь свойства `stored=true` и `indexed=true`. Чтобы добавить поле, используйте Solr Admin UI, как показано на рисунке ниже.



name: signature
field type: string
default: enter a default value if needed
☑ stored
☑ indexed
☐ uninvertible
☐ docValues
☐ multiValued
☐ required

Создание поля в схеме Solr

Шаг 3. Изменение файла solrconfig.xml

Поскольку в ADH Solr всегда работает в режиме SolrCloud, конфигурационные файлы, такие как *solrconfig.xml*, хранятся в ZooKeeper для синхронизации конфигураций между несколькими узлами кластера. Выполните следующие шаги, чтобы обновить *solrconfig.xml* через ZooKeeper:

1. На хосте с компонентами Solr Server и ZooKeeper Server подключитесь к ZooKeeper через CLI:

   ```
   $ sudo /usr/lib/zookeeper/bin/zkCli.sh
   ```

2. В открывшейся оболочке ZooKeeper получите список всех znodes:

   ```
   $ ls /
   ```

   Вывод:

   ```
   [zk: localhost:2181(CONNECTED) 4] ls /
   [Arenadata.Hadoop-11.solr.server, arenadata, hadoop-ha, zookeeper]
   ```

   В этом примере *Arenadata.Hadoop-11.solr.server* — это Znode, в котором хранятся все файлы конфигурации Solr. Данный znode понадобится на дальнейших шагах.

3. Загрузите конфигсет из ZooKeeper в локальную файловую систему с помощью команды zk downconfig:

   ```
   $ /usr/lib/solr/bin/solr zk downconfig -n test_collection -d ~/test_collection
   ```

   Где:

   - `-n test_collection` — это имя Solr-конфигурации для тестовой коллекции, хранящейся в ZooKeeper. Полный путь к конфигурации в ZooKeeper выглядит следующим образом: */Arenadata.Hadoop-11.solr.server/configs/test_collection*.
   - `-d ~/test_collection` — директория в локальной файловой системе для сохранения конфигурации.

   Вывод команды:

   ```
   Downloading configset test_collection from ZooKeeper at ka-adh-1.ru-central1.internal:2181,ka-adh-2.ru-central1.
   internal:2181,ka-adh-3.ru-central1.internal:2181/Arenadata.Hadoop-11.solr.server to directory /home/admin/test_collection/conf
   ```

4. В загруженной директории отредактируйте файл *solrconfig.xml*, например, с помощью `vi`:

   ```
   $ vi test_collection/conf/solrconfig.xml
   ```

5. Добавьте следующий блок цепочки обработчиков запросов обновления (update request processor chain) в *solrconfig.xml*:

   ```
   <updateRequestProcessorChain name="mydedupechain"> ①
      <processor class="solr.processor.SignatureUpdateProcessorFactory">
          <bool name="enabled">true</bool>
          <bool name="overwriteDupes">false</bool> ②
          <str name="signatureField">signature</str> ③
          <str name="fields">txn_id,acc_id,txn_value</str> ④
          <str name="signatureClass">solr.processor.Lookup3Signature</str> ⑤
     </processor>
     <processor class="solr.LogUpdateProcessorFactory" />
     <processor class="solr.RunUpdateProcessorFactory" />
   </updateRequestProcessorChain>
   ```

   ① Имя цепочки обработчиков запросов обновления. В будущем, при отправке запросов на индексирование в Solr, эту цепочку обработчиков можно будет явно указать по имени.

   ② Определяет, следует ли перезаписывать документы с совпадающими сигнатурами. Перезапись работает только если поле `uniqueKey` используется в качестве `signatureField`. Больше информации доступно в разделе Ограничения дедупликации в режиме SolrCloud.

   ③ Имя поля для хранения вычисляемых сигнатур. В текущем примере соответствующее поле (`signature`) было добавлено в схему ранее.

   ④ Одно или несколько полей документов для вычисления сигнатуры документа.

   ⑤ Реализация класса `Signature` для генерации хеша.

6. Сохраните изменения и загрузите обновленную конфигурацию в ZooKeeper с помощью команды zk upconfig:

   ```
   $ /usr/lib/solr/bin/solr zk upconfig -n test_collection -d ~/test_collection
   ```

   Вывод:

   ```
   Uploading /home/admin/test_collection/conf for config test_collection to ZooKeeper at ka-adh-1.ru-central1.internal:2181,ka-
   adh-2.ru-central1.internal:2181,ka-adh-3.ru-central1.internal:2181/Arenadata.Hadoop-11.solr.server
   ```

7. Выполните рестарт Solr-сервиса.

8. Убедитесь, что Solr использует обновленную конфигурацию. Для этого в Solr Admin UI выберите тестовую коллекцию и перейдите в раздел ***Files***. Файлы *solrconfig.xml* и *managed-schema* должны содержать обновленную конфигурацию.

Шаг 4. Отправка данных для индексации

На хосте с компонентом Solr Server создайте файл с тестовыми документами, как показано ниже.

```
[
  {
    "id": 1,
    "txn_id": "1",
    "acc_id": 1001,
    "txn_value": 75.0,
    "comment": "The first transaction."
  },
  {
    "id": 2,
    "txn_id": "1",
    "acc_id": 1001,
    "txn_value": 75.0,
    "comment": "Intentionally duplicate transaction record."
  }
]
```

Эти два документа описывают одну и ту же бизнес-сущность — транзакцию с идентичными значениями `txn_id`, `acc_id`, `txn_value` и, по сути, являются дубликатами. Однако у них разные поля `id`, что заставляет Solr считать их не дубликатами. Чтобы активировать механизм дедупликации, отправьте тестовые документы в Solr с помощью следующего запроса:

```
$ curl -X POST 'http://ka-adh-3.ru-central1.internal:8983/solr/demo_collection_dedup/update?commit=true&update.chain=mydedupechain'
-H 'Content-Type: application/json' --data-binary @transactions.json
```

Где:

- `ka-adh-3.ru-central1.internal` — имя хоста, на котором запущен компонент Solr Server;
- `update.chain=mydedupechain` — имя цепочки обработчиков обновления, через которую следует направить запрос на индексирование.

Вывод:

```
{
  "responseHeader":{
    "rf":1,
    "status":0,
    "QTime":168}}
```

Теперь выполните запрос, чтобы увидеть проиндексированные документы:

```
http://ka-adh-3.ru-central1.internal:8983/solr/test_collection/select?indent=true&q.op=OR&q=*%3A*&omitHeader=true
```

Пример вывода:

```
{
  "response": {
    "numFound": 2,
    "start": 0,
    "numFoundExact": true,
    "docs": [
      {
        "id": "1",
        "txn_id": [1],
        "acc_id": [1001],
        "txn_value": [75],
        "comment": [
          "The first transaction."
        ],
        "signature": "3db4527ba951dcd3",
        "_version_": 1.8136481935224996e+18
      },
      {
        "id": "2",
        "txn_id": [1],
        "acc_id": [1001],
        "txn_value": [75],
        "comment": [
          "Intentionally duplicate transaction record."
        ],
        "signature": "3db4527ba951dcd3",
        "_version_": 1.8136481935256453e+18
      }
    ]
  }
}
```

Обратите внимание на поле `signature`, которое идентично для обоих документов, поскольку они содержат почти идентичные данные. Добавляя сигнатуру в каждый документ, можно легко обнаружить накопившиеся в коллекции дубликаты и при необходимости удалить их.

Ограничения дедупликации в режиме SolrCloud

Хотя Solr предоставляет возможность перезаписывать документы с совпадающими сигнатурами (`<bool name="overwriteDupes">true</bool>`), эта функция работает только в том случае, если поле `uniqueKey` используется в качестве `SignatureField`. Это ограничение касается режима SolrCloud, в котором Solr функционирует в ADH. Таким образом, попытка дедупликации с указанием `signatureField`, отличного от значения параметра `uniqueKey`, не будет работать корректно из-за коллизий при роутинге обновлений к репликам на других узлах кластера.

Обзор Solr UI

Константин Алпашкин

Содержание

Назад к содержанию

Dashboard

Logging

Security

Cloud

- Nodes
- Tree
- ZK Status
- Graph

Schema Designer

Collections

Java Properties

Thread Dump

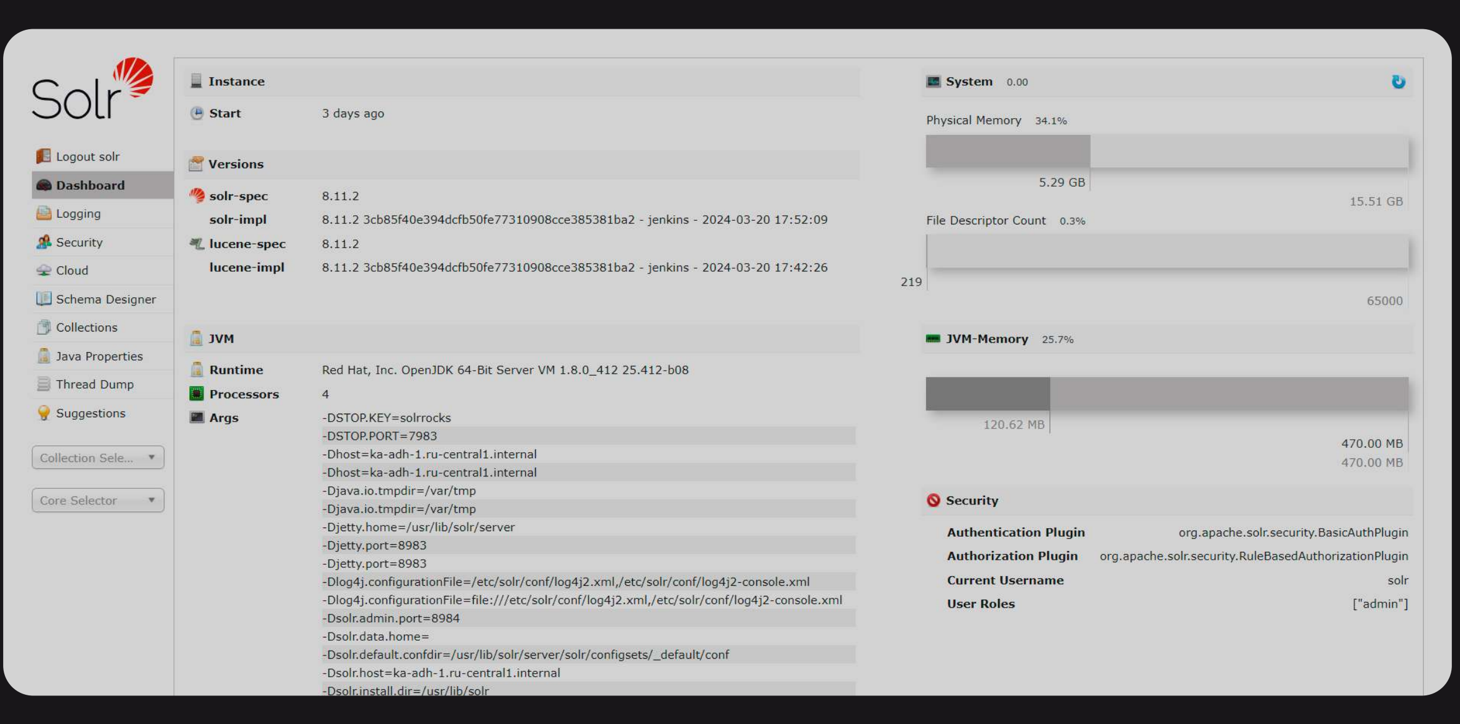
Сервис Solr предоставляет веб-интерфейс, который отражает диагностическую и статус-информацию, позволяет управлять коллекциями, настройками безопасности и так далее. По умолчанию веб-интерфейс доступен по адресу *http://<host>:8983/solr/#/*, где `<host>` — это хост ADH с установленным компонентом Solr Server. Вы можете изменить номер порта по умолчанию (`8983`) в разделе конфигурации ***solr.xml*** в ADCM. Актуальный URL веб-интерфейса Solr доступен в ADCM (***Clusters → <YOUR_CLUSTER> → Services → Solr → Info***).

Веб-интерфейс Solr включает следующие разделы, описанные ниже:

- Dashboard
- Logging
- Security
- Cloud
- Schema Designer
- Collections
- Java Properties
- Thread Dump

Dashboard

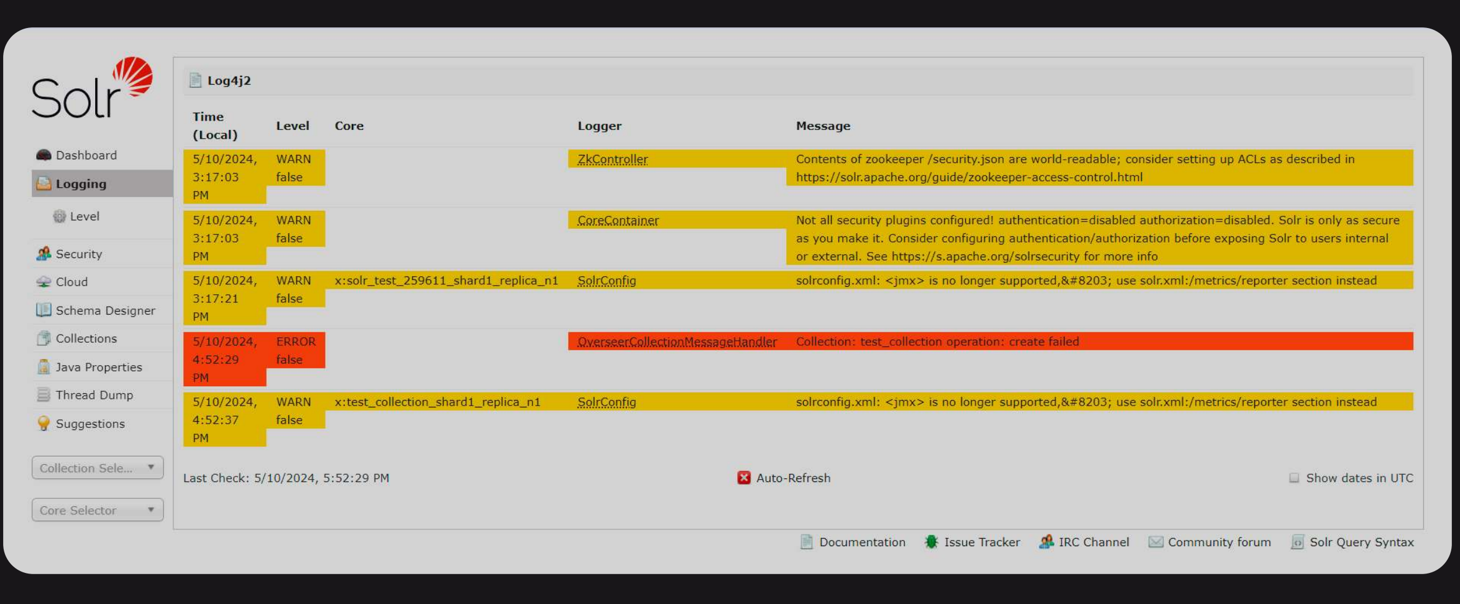
На этой странице отображается информация о Solr-сервере, работающем на определенном хосте. На странице отражены потребление системных ресурсов, параметры JVM и параметры безопасности.



Страница Dashboard

Logging

На этой странице доступны логи Log4j2, сгенерированные сервером Solr.

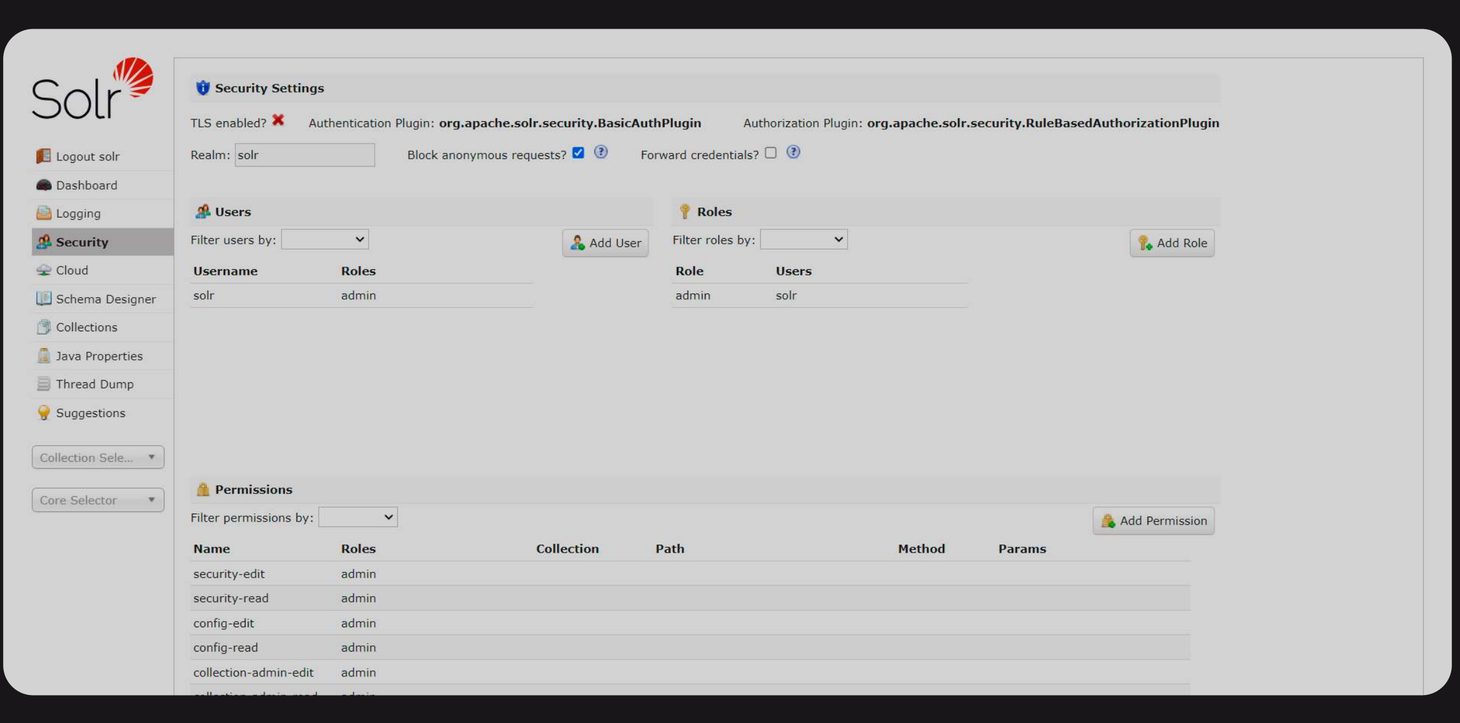


Страница Logging

Используя вкладку ***Level***, можно установить уровни логирования для отдельных компонентов и классов Solr.

Security

Данная страница содержит настройки безопасности Solr, такие как пользователи, роли, разрешения, параметры аутентификации/авторизации. Эти настройки являются общекластерными, то есть влияют на конфигурацию безопасности для всех серверов Solr в кластере ADH.



Страница Security

ПРИМЕЧАНИЕ

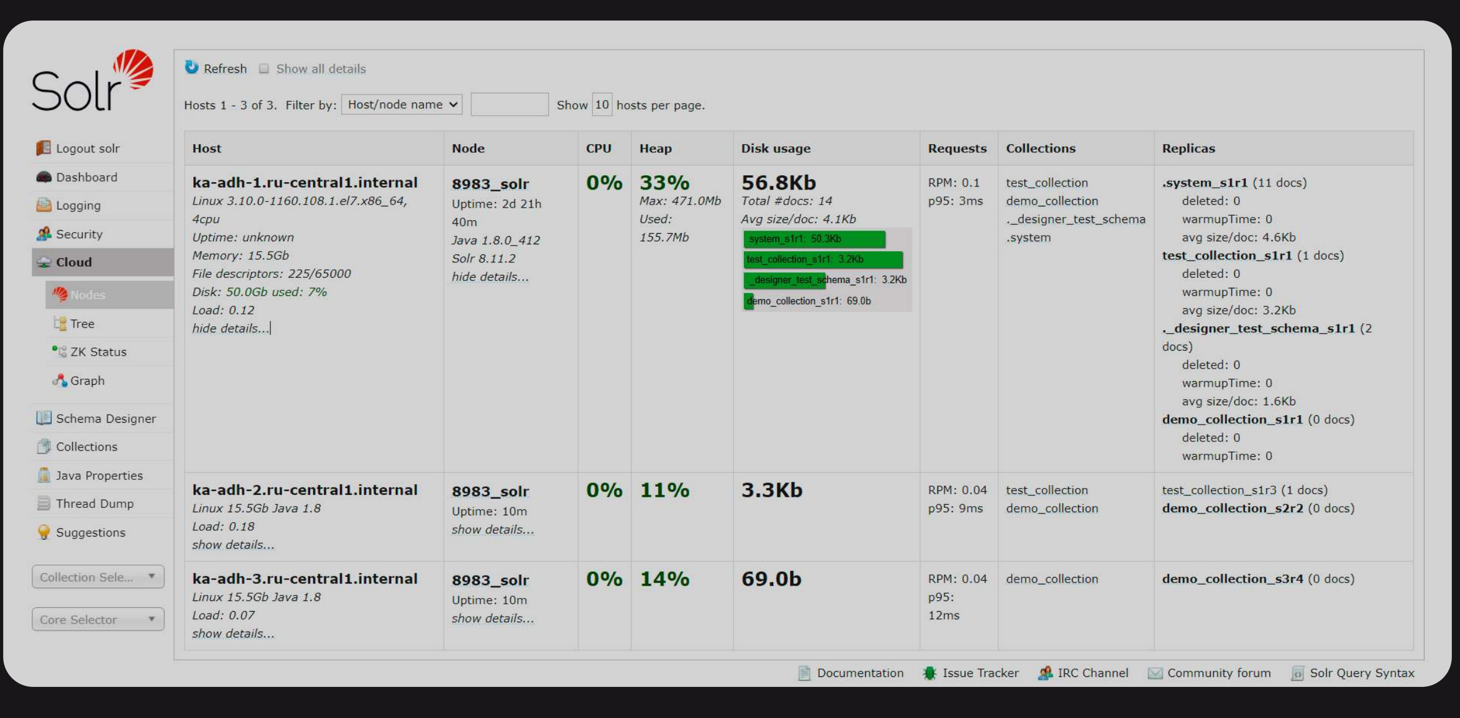
Для отображения содержимого данной страницы необходимо активировать аутентификацию и авторизацию для сервиса Solr.

Cloud

Данный раздел содержит страницы, отражающие активность Solr в контексте всего ADH-кластера.

Nodes

На этой странице отображается информация о состоянии серверов Solr, работающих на различных узлах ADH-кластера.



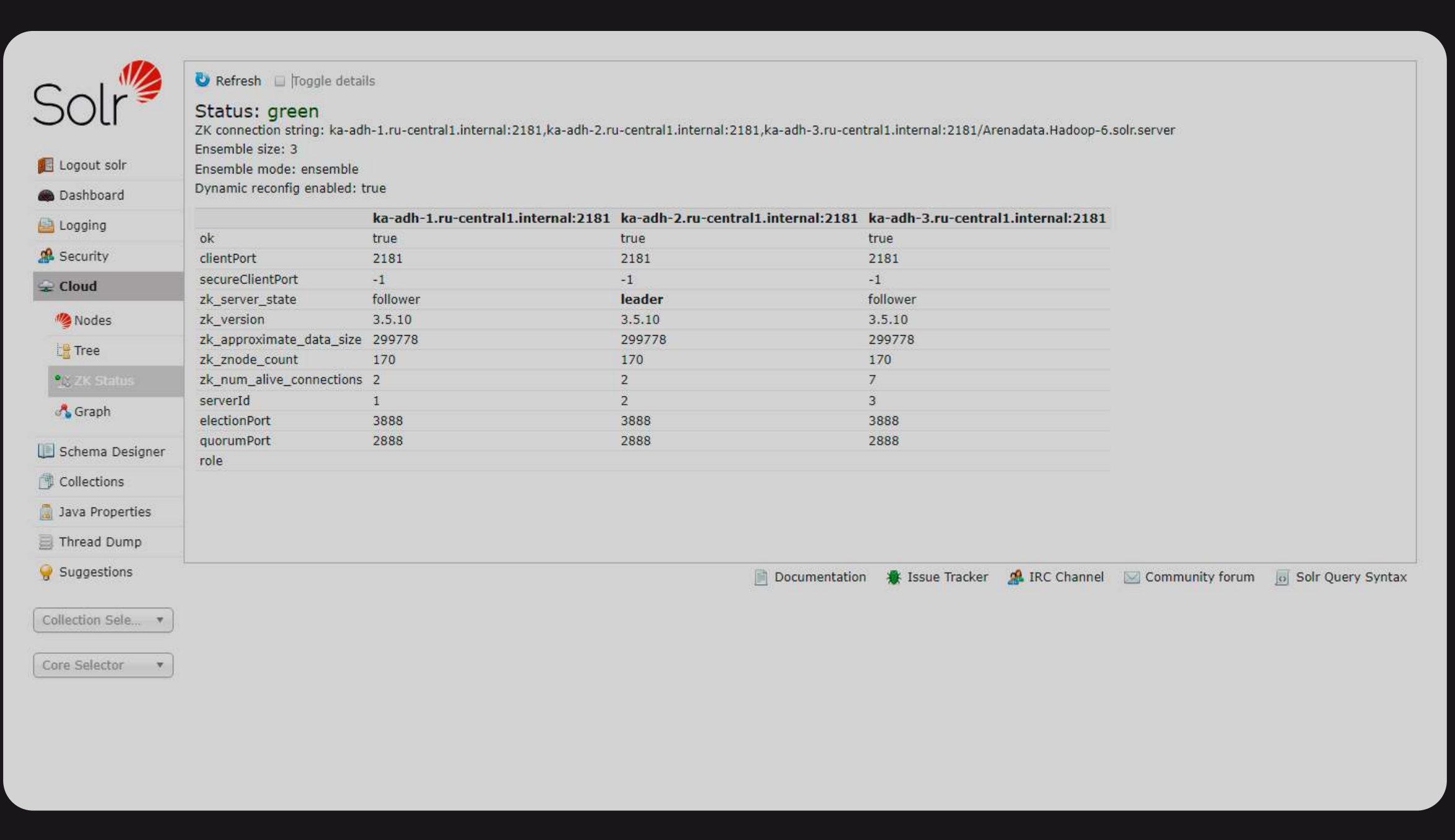
Страница Nodes

Tree

На этой странице отображается структура директорий данных ZooKeeper в виде дерева.

ZK Status

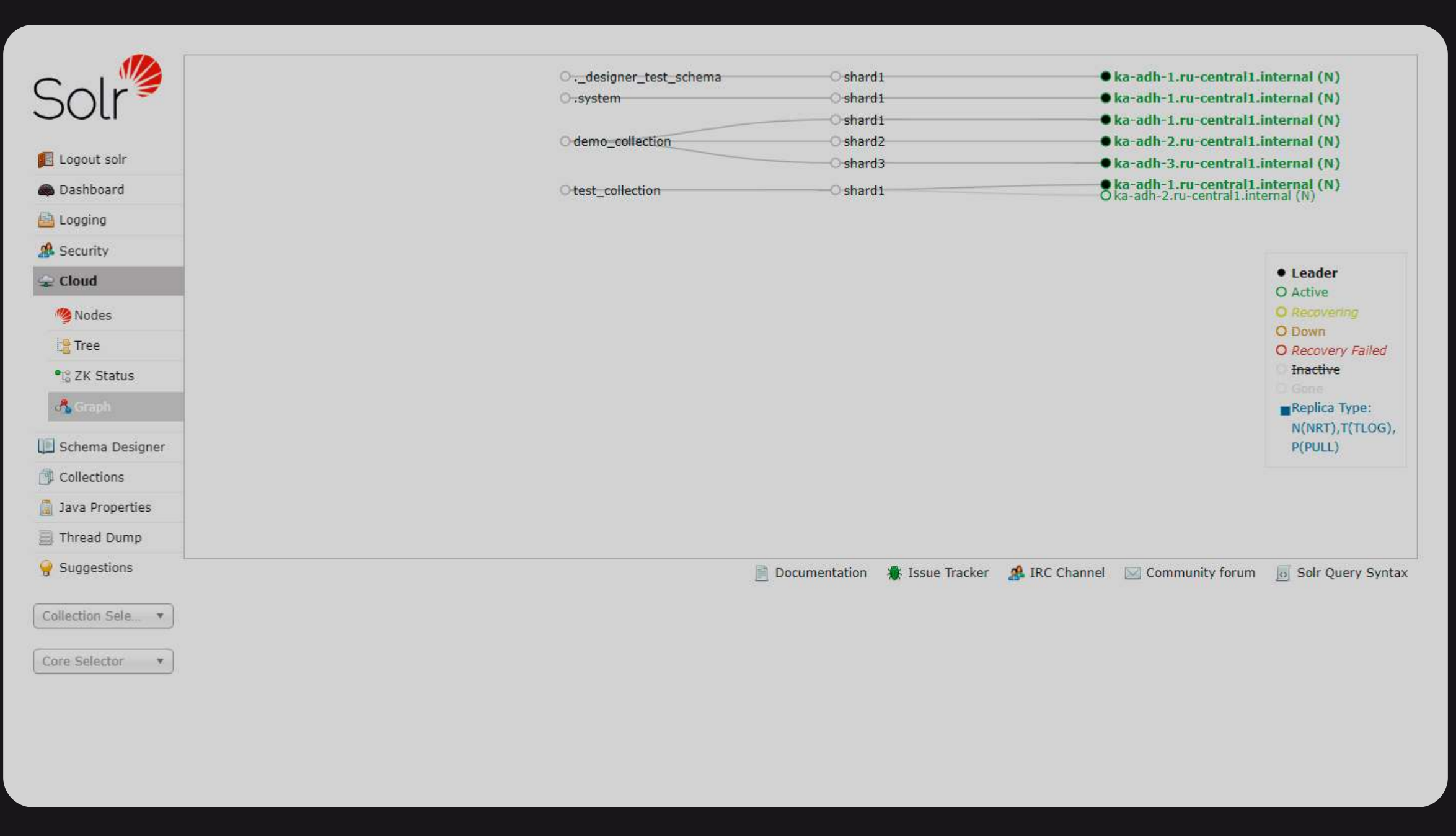
Эта страница отражает состояние ZooKeeper ensemble, используемого Solr. Используйте опцию ***Toggle details***, чтобы просмотреть более подробную информацию о каждом узле.



Страница ZK Status

Graph

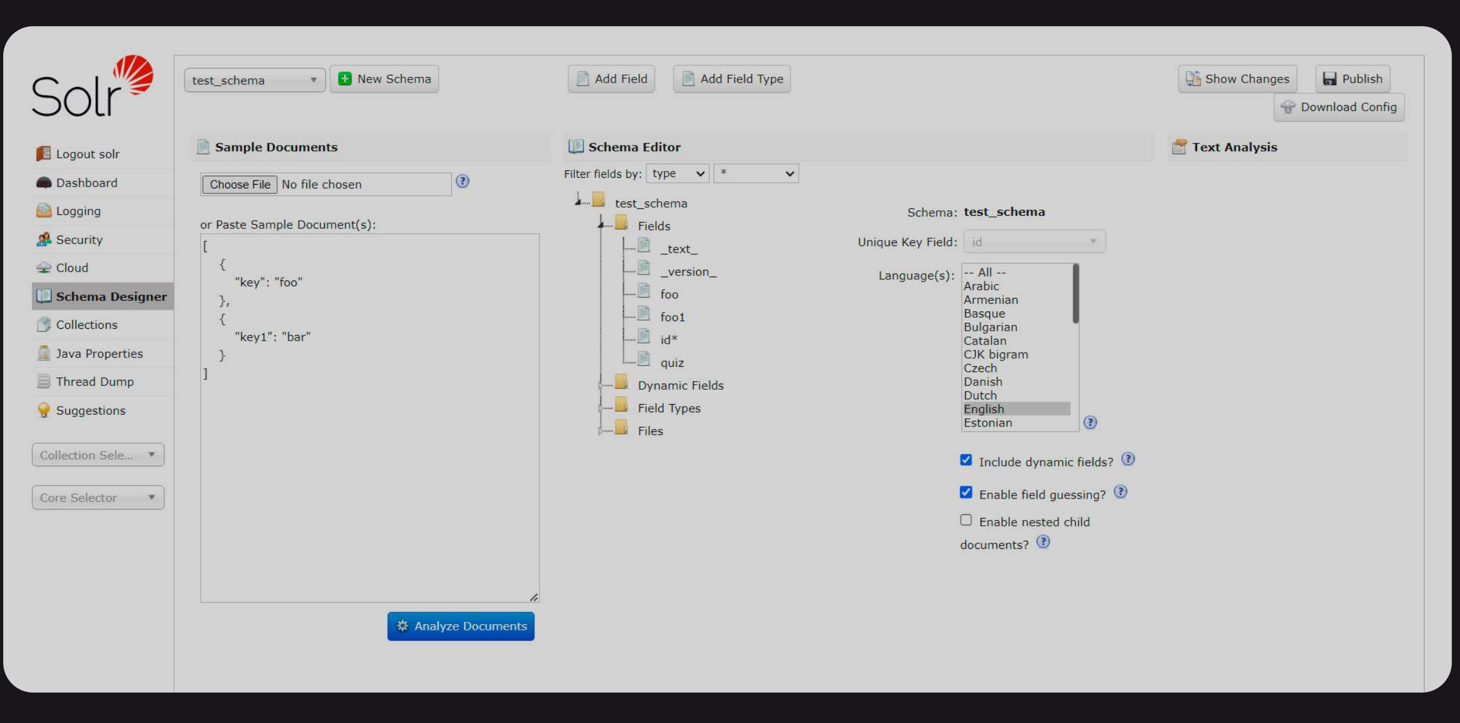
На этой странице отображаются коллекции, представленные в виде графиков, включая шарды и узлы, на которых размещены реплики.



Страница Graph

Schema Designer

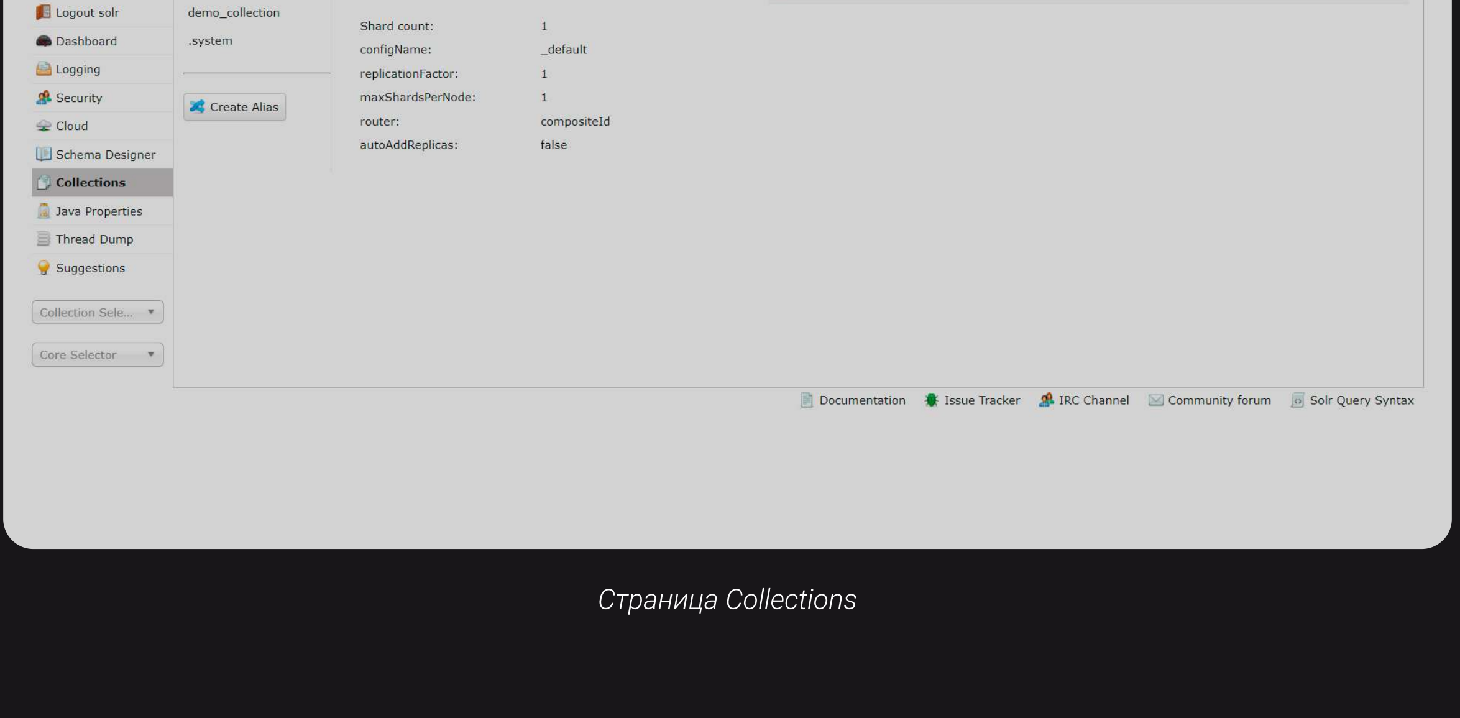
На этой странице вы можете создать новую схему Solr для хранения документов. Можно либо указать все поля схемы вручную, либо загрузить данные в форму, чтобы Solr сгенерировал схему на их основе. Вы также можете протестировать новую схему, используя раздел ***Query Tester***.



Страница Schema Designer

Collections

На этой странице вы можете управлять коллекциями Solr. Здесь вы можете создавать, редактировать и удалять коллекции, изменять количество реплик, создавать новые алиасы и так далее.



Страница Collections

Java Properties

На этой странице выводятся все JVM-параметры, которые использовались при старте Solr, включая classpath, кодировки, параметры памяти JVM и так далее.

Thread Dump

На этой странице доступны трассировки стека, которые можно развернуть для анализа активных потоков серверов Solr.

Коллекции Solr

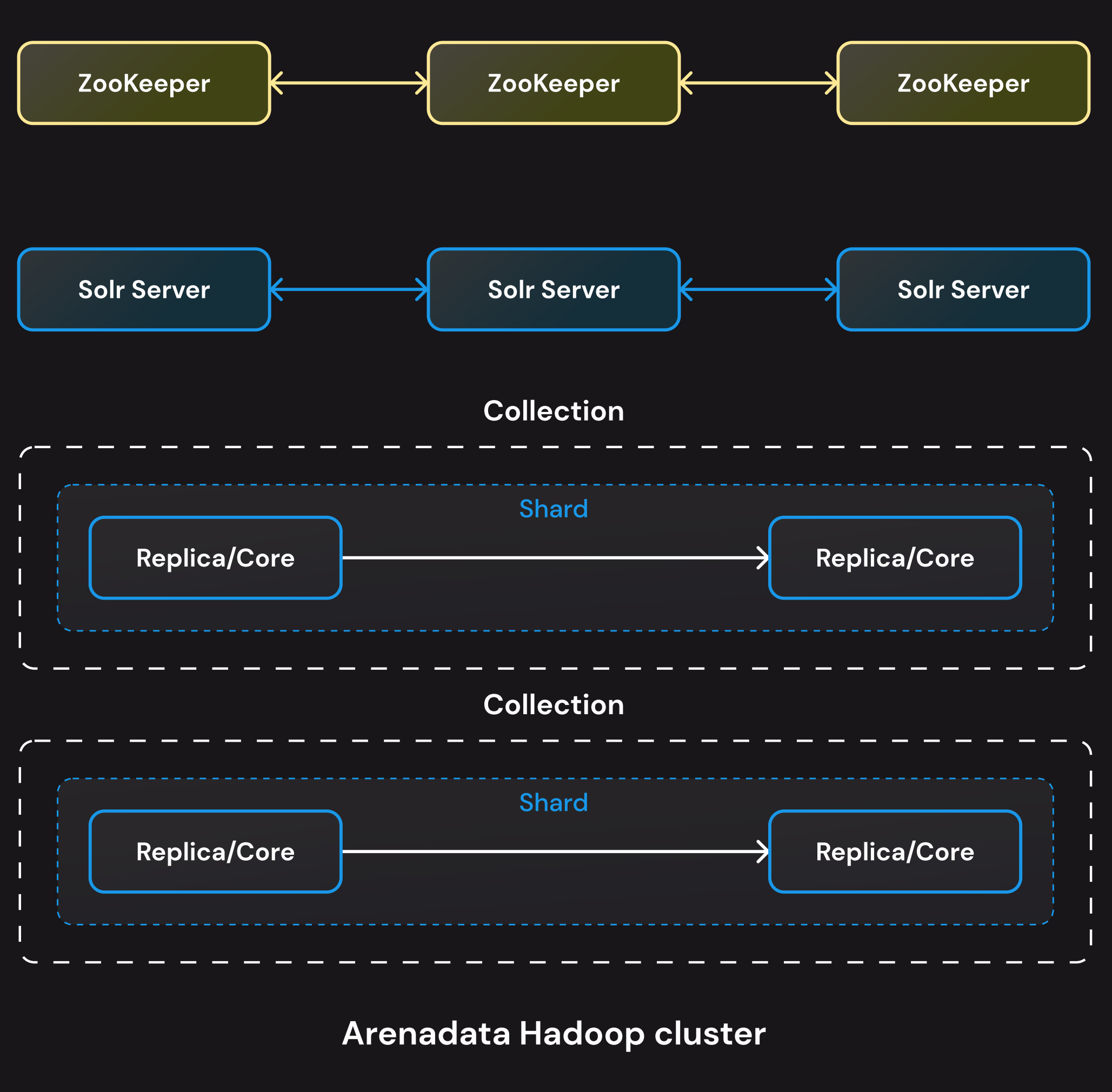
Константин Алпашкин

Содержание

Назад к содержанию

- Создание коллекции через web-интерфейс
- Создание коллекции через CLI
  - Создание коллекции
  - Получение списка коллекций
  - Удаление коллекции
  - Проверка состояния коллекции

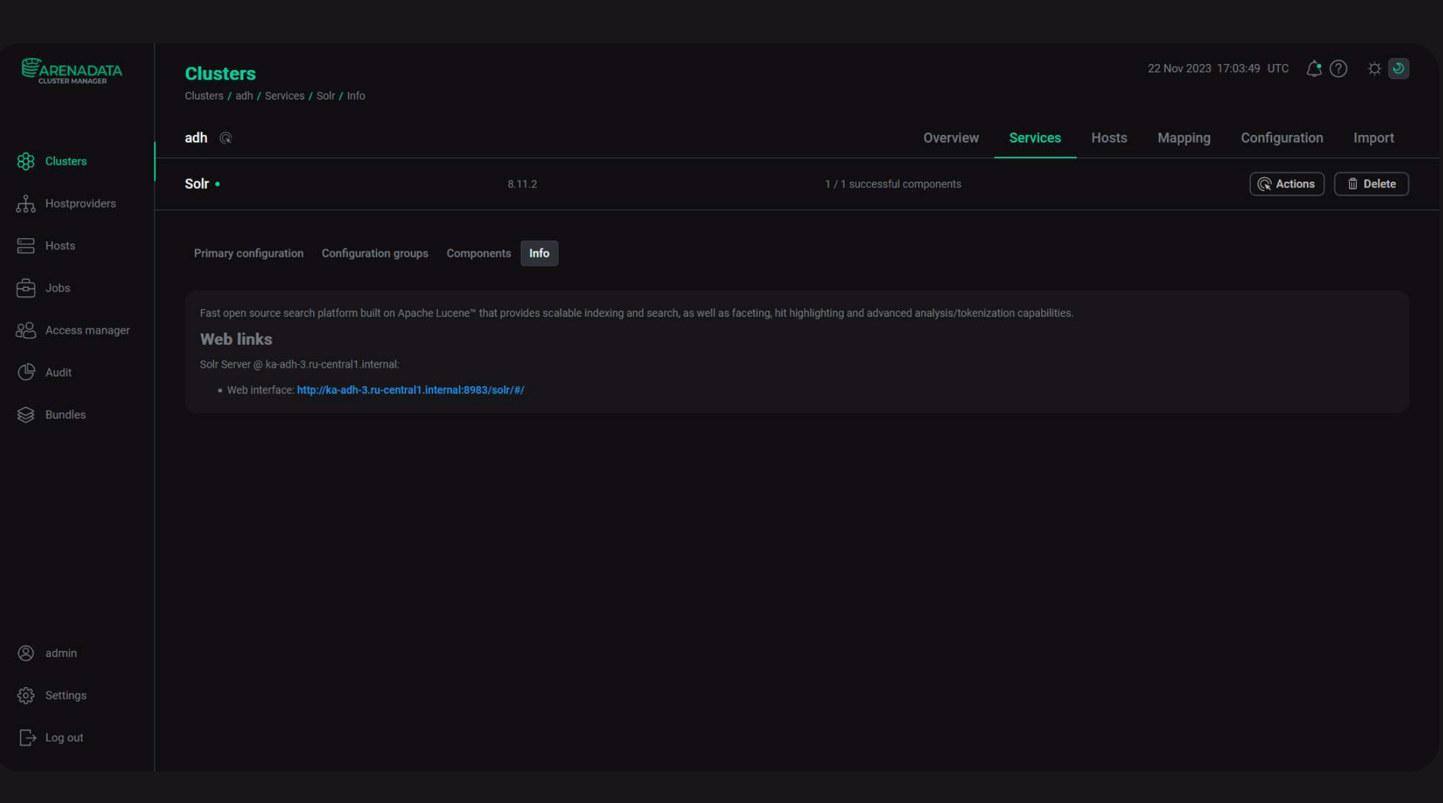
В терминологии Solr **Коллекция (Collection)** — это совокупность проиндексированных документов. Каждая коллекция имеет свой набор настроек и схем, которые могут разниться от коллекции к коллекции. Также Solr-коллекцией называют группу шардов/ядер, которые в совокупности формируют единый логический индекс. Solr разделяет данные на шарды, распределяет шарды между серверами (в зависимости от количества шардов на один узел) и обеспечивает репликацию шардов для надежности, используя фактор репликации (replication factor), установленный на уровне ядра. Рисунок ниже отображает данную концепцию.


ZooKeeper
ZooKeeper
ZooKeeper
Solr Server
Solr Server
Solr Server
Collection
Shard
Replica/Core
Replica/Core
Collection
Shard
Replica/Core
Replica/Core
Arenadata Hadoop cluster

Коллекции Solr в кластере

Создание коллекции через web-интерфейс

1. Откройте ваш браузер и перейдите на URL веб-интерфейса Solr. Вы можете увидеть данный URL на странице ADCM.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / adh / Services / Solr / Info
22 Nov 2023 17:03:49 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
adh
Overview
Services
Hosts
Mapping
Configuration
Import
Solr
8.11.2
1 / 1 successful components
Actions
Delete
Primary configuration
Configuration groups
Components
Info

Fast open source search platform built on Apache Lucene™ that provides scalable indexing and search, as well as faceting, hit highlighting and advanced analysis/tokenization capabilities.

Web links

Solr Server @ ka-adh-3.ru-central1.internal:

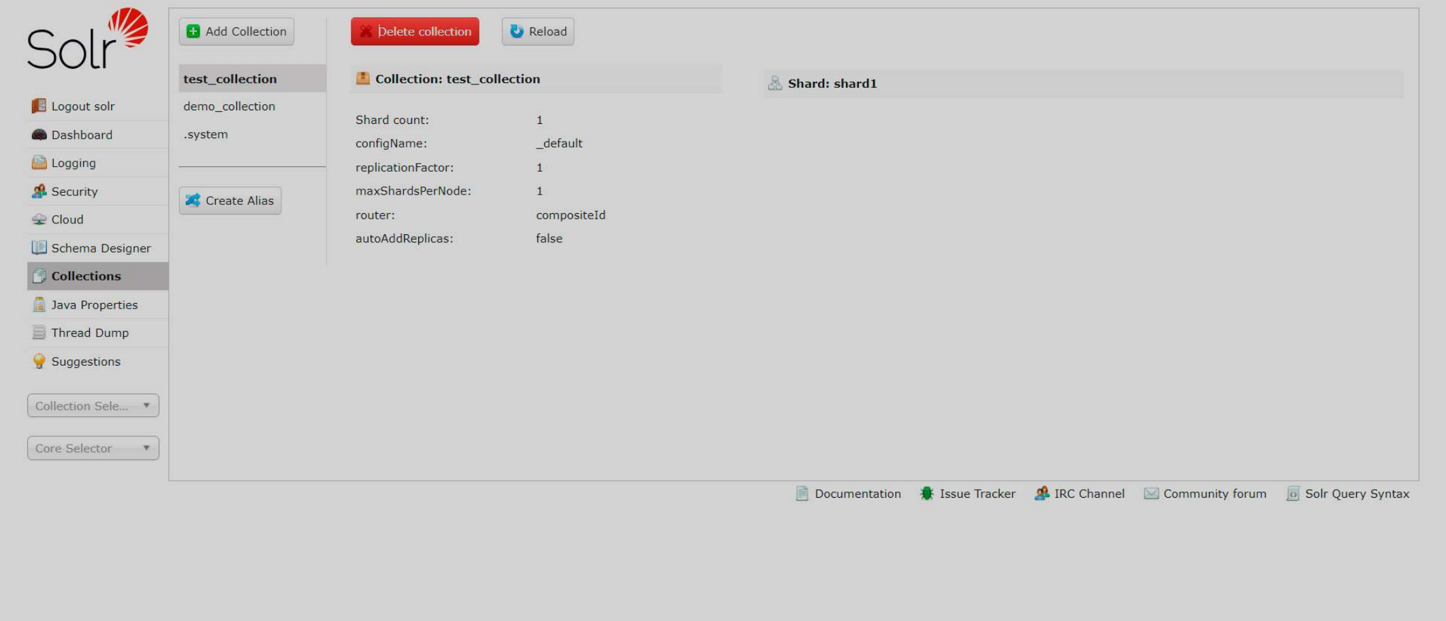
- Web interface: http://ka-adh-3.ru-central1.internal:8983/solr/#/

Страница Solr в ADCM

Вкладка ***Collections*** содержит список существующих коллекций в кластере. Кликнув на имя коллекции, можно увидеть основные метаданные коллекции — ее параметры, текущие шарды и реплики; также вы можете добавить или удалить отдельные реплики.

С помощью кнопок вверху страницы, вы можете управлять коллекциями в кластере, например, добавлять новые коллекции и алиасы, перезагружать или удалять отдельные коллекции и так далее.

Чтобы создать новую коллекцию, кликните ***Add Collection***.


Solr
Add Collection
Delete collection
Reload
test_collection
demo_collection
.system
Create Alias
Logout solr
Dashboard
Logging
Security
Cloud
Schema Designer
Collections
Java Properties
Thread Dump
Suggestions
Collection Sele...
Core Selector
Shard: shard1
Documentation
Issue Tracker
IRC Channel
Community forum
Solr Query Syntax

Collection: test_collection

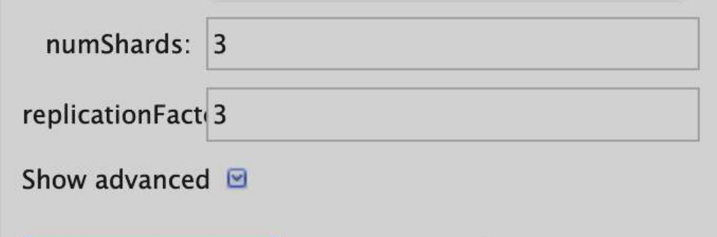
Shard count: 1
configName: _default
replicationFactor: 1
maxShardsPerNode: 1
router: compositeId
autoAddReplicas: false

Web-интерфейс Solr

2. В появившейся панели укажите параметры коллекции:

   - ***name***. Имя коллекции. Этот параметр обязателен. При выборе имени придерживайтесь следующих правил:
     - Используйте только ASCII буквенно-цифровые символы (`A-Za-z0-9`), дефис (`-`), или нижнее подчеркивание (`_`).
     - Не используйте строки "shard" и "replica".
   - ***config set***. Конфигурация для создаваемой коллекции. Вы можете выбрать `_default` или найти нужную конфигурацию, используя поиск.
   - ***config set***. Количество шардов, которые могут быть созданы в коллекции. Этот параметр является обязательным, если для поля `router` выбрано значение `Composite ID`.
   - ***replicationFactor***. Количество реплик, создаваемых на каждый шард. Параметр опциональный.

   > РЕКОМЕНДАЦИЯ
   >
   > Значение по умолчанию — `1`. Максимальное значение должно соответствовать количеству запущенных узлов Solr сервера.


Select an Option
Add Collection
Cancel

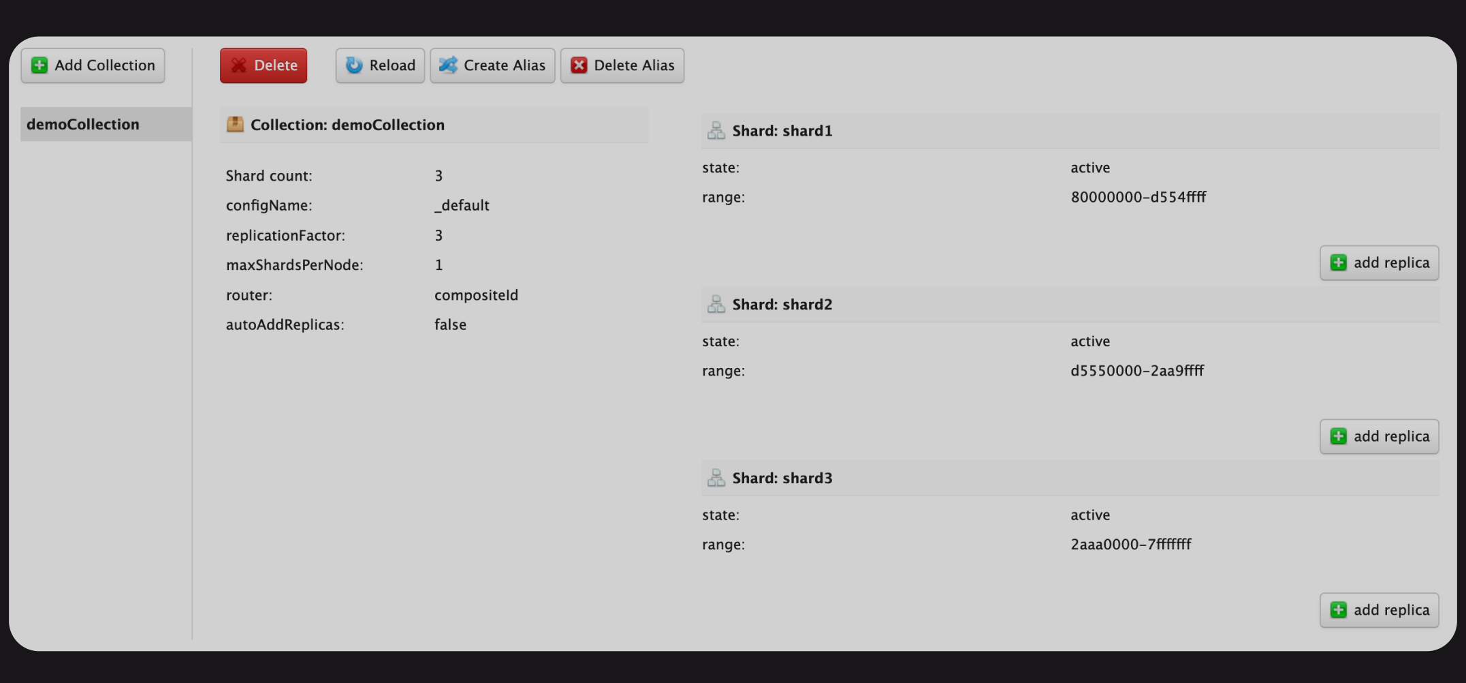
name: demoCollection
config set: Select an Option
numShards: 3
replicationFact 3
Show advanced ☑

Создание новой коллекции

3. (Опционально) Кликните ***Show advanced***, чтобы увидеть дополнительные настройки:

   - ***router***. Используемый роутер (router). Роутер определяет распределение документов по шардам. Доступные значения: `Composite ID` или `Implicit`. Параметр `shards` является обязательным, если используется роутер `Implicit`. При использовании роутера типа `Composite ID` обязателен параметр `numShards`.
     - ***Implicit***. Документы не отправляются автоматически в другие шарды. Все документы отправляются в тот шард, который был указан в запросе на индексацию (или внутри каждого документа).
     - ***Composite ID***. Данный роутер хэширует значение поля `uniqueKey`. Затем находит данный хэш в кластере и определяет, в какой шард должен отправиться документ; такой роутер дает дополнительную возможность управлять роутингом вручную.
   - ***maxShardsPerNode***. При создании коллекции, шарды и/или реплики распределяются по всем живым (live) узлам, так что две реплики одного шарда не могут находиться на одном узле. Если узел недоступен при вызове команды `CREATE`, такой узел не получит ни одной части новой коллекции, и это приводит к слишком большому количеству реплик на одном живом узле. Параметр ***maxShardsPerNode*** устанавливает предел на количество реплик, распределяемых командой `CREATE` на каждый узел. Если коллекция целиком не помещается на живых узлах, коллекция не будет создана. Значение по умолчанию: `1`.
   - ***shards***. Список имен шардов, разделенных запятой (например, `shard-x,shard-y,shard-z`). Этот параметр является обязательным при использовании роутера типа `Implicit`.
   - ***router.field***. Если этот параметр задан, то для того, чтобы определить, в какой шард направлять документ, роутер будет использовать кастомное поле для вычисления хеша вместо поля `uniqueKey`. Если значение указанного поля `null`, такой документ не будет обработан.
   - ***autoAddReplicas***. Позволяет автоматически добавлять реплики на общих файловых системах, если параметр равен `true`. Значение по умолчанию: `false`.

4. После установки всех параметров, кликните ***Add Collection***, чтобы создать новую коллекцию.


Add Collection
Delete
Reload
Create Alias
Delete Alias
demoCollection
add replica
add replica
add replica

Collection: demoCollection

Shard count: 3
configName: _default
replicationFactor: 3
maxShardsPerNode: 1
router: compositeId
autoAddReplicas: false

Shard: shard1
state: active
range: 80000000-d554ffff

Shard: shard2
state: active
range: d5550000-2aa9ffff

Shard: shard3
state: active
range: 2aaa0000-7fffffff

Новая коллекция

Создание коллекции через CLI

Данный раздел описывает базовые операции над коллекциями с использованием интерфейса командной строки (CLI). Все операции над Solr-коллекциями вызываются скриптом */usr/lib/solr/bin/solr* или утилитой `solrctl`.

> ПРИМЕЧАНИЕ
>
> Все примеры в данном разделе подразумевают отсутствие Kerberos в кластере.

Создание коллекции

Для создания новой коллекции доступны следующие команды:

```
$ /usr/lib/solr/bin/solr create -c <collection_name> -s <shards_num> -rf <replicas_num>
# или
$ solrctl collection --create <collection_name> -s <shards_num> -c <config_set>
```

Где:

- <collection_name> — буквенно-цифровое имя коллекции;
- <shards_num> — количество шардов (shards), на которое будет разбиты коллекция;
- <replicas_num> — количество реплик каждого документа в коллекции.

Пример:

```
$ /usr/lib/solr/bin/solr create -c Collection0 -s 2 -rf 2
```

Получение списка коллекций

Для вывода списка доступных коллекций используйте следующую команду:

```
$ solrctl collection --list
```

Вывод команды имеет следующий вид:

```
Collection0 (5)
demoCollection_1 (5)
...
```

Удаление коллекции

Для удаления коллекции доступны следующие команды:

```
$ /usr/lib/solr/bin/solr delete -c <collection_name>
# или
$ solrctl collection --delete <collection_name>
```

Вывод команды имеет следующий вид:

```
{
  "responseHeader":{
    "status":0,
    "QTime":236},
  "success":{
    "ka-adh-3.ru-central1.internal:8983_solr":{"responseHeader":{
        "status":0,
        "QTime":26}},
    "ka-adh-1.ru-central1.internal:8983_solr":{"responseHeader":{
        "status":0,
        "QTime":26}},
    "ka-adh-2.ru-central1.internal:8983_solr":{"responseHeader":{
        "status":0,
        "QTime":42}}}}


Deleted collection 'collection1' using command:
http://ka-adh-1.ru-central1.internal:8983/solr/admin/collections?action=DELETE&name=collection1
```

Проверка состояния коллекции

Для проверки состояния (health check) определенной коллекции и получения диагностической информации используйте команду:

```
$ /usr/lib/solr/bin/solr healthcheck -c <collection_name>
```

Пример:

```
$ /usr/lib/solr/bin/solr healthcheck -c Collection0
```

Вывод команды имеет следующий вид:

```
{
  "collection":"Collection0",
  "status":"healthy",
  "numDocs":0,
  "numShards":2,
  "shards":[
    {
      "shard":"shard1",
      "status":"healthy",
      "replicas":[
        {
          "name":"core_node3",
          "url":"http://ka-adh-2.ru-central1.internal:8983/solr/Collection0_shard1_replica_n1/",
          "numDocs":0,
          "status":"active",
          "uptime":"2 days, 22 hours, 21 minutes, 49 seconds",
          "memory":"158.9 MB (%32.8) of 485 MB"},
        {
          "name":"core_node5",
          "url":"http://ka-adh-1.ru-central1.internal:8983/solr/Collection0_shard1_replica_n2/",
          "numDocs":0,
          "status":"active",
          "uptime":"2 days, 22 hours, 21 minutes, 49 seconds",
          "memory":"240.4 MB (%49.9) of 481.5 MB",
          "leader":true}]},
    {
      "shard":"shard2",
      "status":"healthy",
      "replicas":[
        {
          "name":"core_node7",
          "url":"http://ka-adh-3.ru-central1.internal:8983/solr/Collection0_shard2_replica_n4/",
          "numDocs":0,
          "status":"active",
          "uptime":"2 days, 22 hours, 21 minutes, 49 seconds",
          "memory":"196 MB (%40) of 489.5 MB",
          "leader":true},
        {
          "name":"core_node8",
          "url":"http://ka-adh-2.ru-central1.internal:8983/solr/Collection0_shard2_replica_n6/",
          "numDocs":0,
          "status":"active",
          "uptime":"2 days, 22 hours, 21 minutes, 49 seconds",
          "memory":"160.8 MB (%33.2) of 485 MB"}]}]}
```

Логирование в Solr

Константин Алпашкин

Содержание

Назад к содержанию

Расположение и формат лог-файлов
Настройка логирования
  Стандартный файл log4j2.xml
Loglevel API
Логирование медленных запросов
Logging UI

Под капотом Solr использует фреймворк Log4j 2 для ведения логов. Сервис генерирует несколько типов лог-файлов, а также предоставляет Logging UI для просмотра и анализа логов Solr. Все настройки логирования указываются через конфигурационные файлы Log4j2, которые можно редактировать через ADCM.

Расположение и формат лог-файлов

Solr генерирует несколько типов логов для сохранения всех событий в системе. В файловой системе эти логи хранятся в виде обычных текстовых файлов с расширением *.log* в директории */var/log/solr/* (локация по умолчанию). Большинство лог-событий Solr имеют следующую структуру:

```
2024-10-21 15:04:36.923 INFO  (coreLoadExecutor-27-thread-1-processing-n:ka-adh-3.ru-central1.internal:8983_solr)
[c:test_collection s:shard1 r:core_node2 x:test_collection_shard1_replica_n1] o.a.s.c.CoreContainer Creating SolrCore
'test_collection_shard1_replica_n1' using configuration from configset test_collection, trusted=true
```

Путь к лог-файлам настраивается индивидуально и может быть изменен путем редактирования конфигурации log4j2.xml через ADCM. Пример содержимого директории */var/log/solr/* показан ниже:

```
-rw-r--r-- 1 solr solr    3141 Nov  6 12:50 solr-8983-console.log ①
-rw-r--r-- 1 solr solr  703758 Nov 10 20:11 solr_gc.log.0.current ②
-rw-r--r-- 1 solr solr 4686227 Nov  9 13:00 solr.log ③
-rw-r--r-- 1 solr solr  113261 Oct 21 15:23 solr.log.1
-rw-r--r-- 1 solr solr 9054280 Oct 28 23:32 solr.log.10
-rw-r--r-- 1 solr solr  178374 Oct 21 15:52 solr.log.2
-rw-r--r-- 1 solr solr 1829020 Oct 22 19:02 solr.log.3
-rw-r--r-- 1 solr solr   40809 Oct 22 19:10 solr.log.4
-rw-r--r-- 1 solr solr   53809 Oct 22 19:14 solr.log.5
-rw-r--r-- 1 solr solr  463368 Oct 22 19:43 solr.log.6
-rw-r--r-- 1 solr solr   68754 Oct 22 19:50 solr.log.7
-rw-r--r-- 1 solr solr   57230 Oct 22 19:54 solr.log.8
-rw-r--r-- 1 solr solr  234734 Oct 22 20:31 solr.log.9
-rw-r--r-- 1 solr solr       0 Oct 21 14:21 solr_slow_requests.log ④
```

① Хранит вывод консоли Solr, перенаправленный в STDOUT.

② Хранит информацию о вызовах GC в Solr.

③ Основные файлы логов, в которых хранятся записи событий. По умолчанию файлы *solr.log.{n}* ротируются по достижении определенного размера.

④ Хранит информацию о медленных запросах.

Настройка логирования

При использовании Log4j2 настройка процесса логирования в Solr сводится к редактированию соответствующих конфигурационных файлов Log4j2:

- *log4j2.xml* — основной конфигурационный файл, отвечающий за логирование всех событий Solr Server.
- *log4j2-console.xml* — конфигурационный файл, отвечающий за логирование активности консольного клиента.

Данные файлы хранятся в директории */usr/lib/solr/server/resources* и Solr считывает их при запуске. ADCM предоставляет редактор, в котором вы можете изменить содержимое этих файлов. Чтобы отредактировать конфигурацию Log4j2 с помощью ADCM:

1. В ADCM перейдите на страницу ***Clusters*** → ***<clusterName>*** → ***Services*** → ***Solr*** → ***Components*** → ***Solr Server*** → ***Primary configuration***.
2. Выберите нужный файл конфигурации Log4j2 (например, ***log4j2.xml***).
3. Измените конфигурацию. Ниже в статье представлен дефолтный файл *log4j2.xml*, в котором описаны основные конфигурационные параметры. Подробная информация о конфигурационных свойствах Log4j доступна в документации Log4j 2.
4. Сохраните конфигурацию сервиса Solr.
5. Выполните рестарт сервиса Solr.

Стандартный файл log4j2.xml

Ниже приведена стандартная конфигурация *log4j2.xml*, используемая Solr. Основные параметры конфигурации выделены комментариями.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Configuration for asynchronous logging -->
<Configuration>
<Appenders>
    <Console name="STDOUT" target="SYSTEM_OUT"> ①
      <PatternLayout>
        <Pattern> ②
          %maxLen{%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p (%t) [%X{collection} %X{shard} %X{replica} %X{core}] %c{1.} %m%notEmpty{
=>%ex{short}}}{10240}%n
        </Pattern>
      </PatternLayout>
    </Console>

    <RollingRandomAccessFile ③
        name="MainLogFile"
        fileName="/var/log/solr/solr.log" ④
        filePattern="/var/log/solr/solr.log.%i" > ⑤
      <PatternLayout>
        <Pattern>
          %maxLen{%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p (%t) [%X{collection} %X{shard} %X{replica} %X{core}] %c{1.} %m%notEmpty{
=>%ex{short}}}{10240}%n
        </Pattern>
      </PatternLayout>
      <Policies>
        <OnStartupTriggeringPolicy /> ⑥
        <SizeBasedTriggeringPolicy size="32 MB"/> ⑦
      </Policies>
      <DefaultRolloverStrategy max="10"/> ⑧
    </RollingRandomAccessFile>

    <RollingRandomAccessFile ⑨
        name="SlowLogFile"
        fileName="/var/log/solr/solr_slow_requests.log"
        filePattern="/var/log/solr/solr_slow_requests.log.%i" >
      <PatternLayout>
        <Pattern>
          %maxLen{%d{yyyy-MM-dd HH:mm:ss.SSS} %-5p (%t) [%X{collection} %X{shard} %X{replica} %X{core}] %c{1.} %m%notEmpty{
=>%ex{short}}}{10240}%n
        </Pattern>
      </PatternLayout>
      <Policies>
        <OnStartupTriggeringPolicy />
        <SizeBasedTriggeringPolicy size="32 MB"/>
      </Policies>
      <DefaultRolloverStrategy max="10"/>
    </RollingRandomAccessFile>

  </Appenders>
  <Loggers> ⑩
    <AsyncLogger name="org.apache.hadoop" level="warn"/>
    <AsyncLogger name="org.apache.solr.update.LoggingInfoStream" level="off"/>
    <AsyncLogger name="org.apache.zookeeper" level="warn"/>
    <AsyncLogger name="org.apache.solr.core.SolrCore.SlowRequest" level="info" additivity="false">
      <AppenderRef ref="SlowLogFile"/>
    </AsyncLogger>

    <AsyncRoot level="info">
      <AppenderRef ref="MainLogFile"/>
      <AppenderRef ref="STDOUT"/>
    </AsyncRoot>
  </Loggers>
</Configuration>

<!-- Configuration for synchronous logging
there _may_ be a very small window where log messages will not be flushed
to the log file on abnormal shutdown. If even this risk is unacceptable, use
the configuration below
-->
<!--Configuration>
    ...
</Configuration-->
```

① Консольный аппендер (appender) с именем `STDOUT`, который направляет логи в стандартный вывод.

② Шаблон, используемый для формирования отдельных лог-записей, сохраняемых в файл аппендером. Подробная информация о синтаксисе `PatternLayout` доступна на странице Log4j2 layouts.

③ Использование аппендера RollingRandomAccessFile, который записывает Solr-логи в файлы, ротируемые по размеру.

④ Указание имени лог-файла.

⑤ Указание шаблона имен лог-файлов при выполнении переноса (rollover).

⑥ Политика, выполняющая перенос при каждом запуске Solr.

⑦ Политика, выполняющая перенос при достижении файлом размера 32 МБ.

⑧ Определяет максимальное количество архивных логов. Когда этот предел достигнут, самые старые файлы удаляются.

⑨ Использование аппендера `RollingRandomAccessFile`, который записывает информацию о медленных запросах.

⑩ Список всех логеров (loggers), которые логируют события Solr.

> **ПРИМЕЧАНИЕ**
>
> По умолчанию файл *log4j2.xml* имеет две конфигурации — для синхронного и асинхронного логирования. Синхронная конфигурация отключена по умолчанию, так как асинхронное логирование Log4j обеспечивает лучшую производительность, выполняя все операции ввода-вывода в отдельном потоке. Однако в случае нештатного завершения работы Solr существует вероятность, что некоторые события могут не записаться на диск и, таким образом, могут быть потеряны. Если это неприемлемо, используйте синхронную конфигурацию Logj2. Для этого в файле _log4j2.xml закомментируйте блок с асинхронной конфигурацией, затем раскомментируйте блок с синхронной конфигурацией и перезапустите Solr.

Loglevel API

Solr предоставляет REST-эндпойнт для обновления конфигурации логирования компонентов Solr. Например, чтобы установить новый уровень логирования для отдельного компонента Solr (`org.apache.solr.security`), отправьте следующий запрос:

```
$ curl http://ka-adh-3.ru-central1.internal:8983/solr/admin/info/logging --data-binary "set=org.apache.solr.security:WARN"
```

Запрос на установку уровня логирования сразу для нескольких компонентов Solr (например, для всех дочерних компонентов `org.apache.zookeeper`) выглядит следующим образом:

```
$ curl http://ka-adh-3.ru-central1.internal:8983/solr/admin/info/logging --data-binary "set=org.apache.zookeeper:WARN"
```

В качестве ответа Solr возвращает JSON-документ, отражающий текущие уровни Log4j2 для каждого компонента Solr. Пример ответа приведен ниже.

Ответ Solr-сервера

```
{
  "responseHeader":{
    "status":0,
    "QTime":1},
  "watcher":"Log4j2",
  "levels":["ALL",
    "TRACE",
    "DEBUG",
    "INFO",
    "WARN",
    "ERROR",
    "FATAL",
    "OFF"],
  "loggers":[{
      "name":"root",
      "level":"INFO",
      "set":true},
    {
      "name":"com",
      "level":null,
      "set":false},
    {
      "name":"com.codahale",
      "level":null,
      "set":false},
    {
      "name":"com.codahale.metrics",
      "level":null,
      "set":false},
    {
      "name":"com.codahale.metrics.jmx",
      "level":null,
      "set":false},
    {
      "name":"com.codahale.metrics.jmx.JmxReporter",
      "level":"INFO",
      "set":true},
    {
      "name":"org",
      "level":"INFO",
      "set":true},
    {
      "name":"org.apache",
      "level":null,
      "set":false},
    {
      "name":"org.apache.calcite",
      "level":null,
      "set":false},
    {
      "name":"org.apache.calcite.avatica",
      "level":null,
      "set":false},
    {
      "name":"org.apache.calcite.avatica.remote",
      "level":null,
      "set":false},
    {
      "name":"org.apache.calcite.avatica.remote.Driver",
      "level":"INFO",
      "set":true},
    {
      "name":"org.apache.hadoop",
      "level":"WARN",
      "set":true},
    ....
```

Логирование медленных запросов

В приложениях, которые часто отправляют поисковые запросы Solr, логирование каждого запроса может привести к избыточному количеству лог-данных и, в зависимости от объема, может негативно отразиться на производительности. Когда подробные логи необходимы (например, для автоматического анализа логов или раннего обнаружения возможных проблем), логирование каждого запроса может быть полезным. С другой стороны, если вас интересуют только предупреждения и ошибки, связанные с запросами, вы можете установить уровень `WARN` для ведения логов. Однако в этом случае возникает потенциальная проблема — Solr не будет выдавать предупреждения о запросах, которые выполняются длительное время, поскольку такие запросы будут логироваться с уровнем `INFO`.

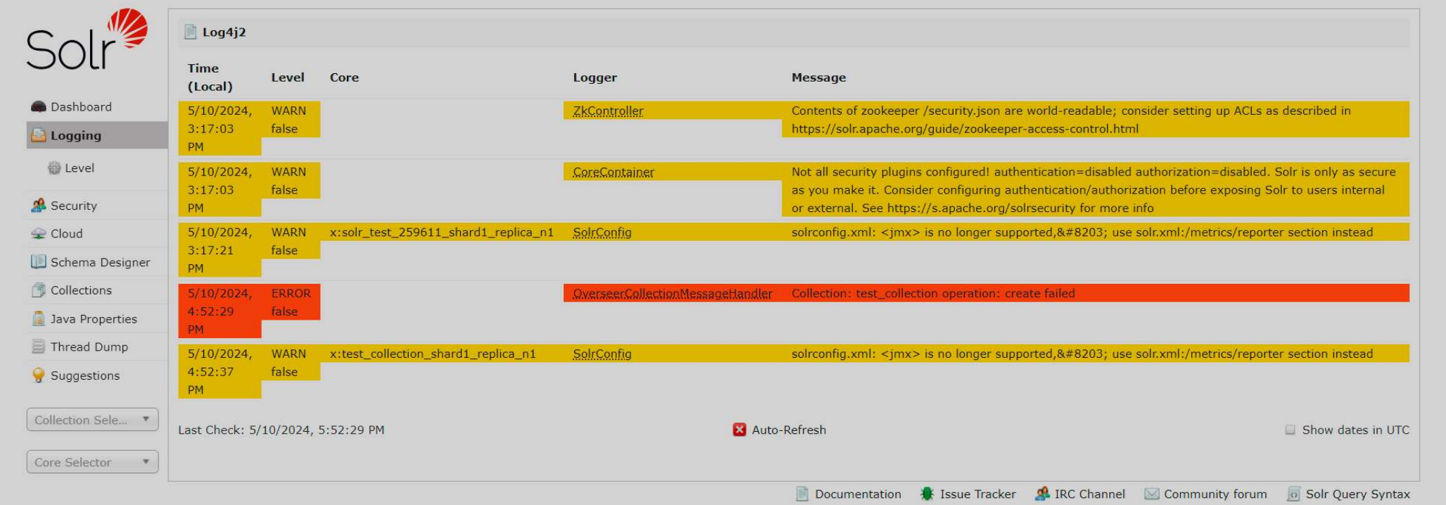
Если вы хотите использовать уровень логирования `WARN`, но при этом также иметь возможность обнаруживать медленные запросы, Solr предоставляет возможность установить тайм-аут, при превышении которого запрос будет считаться медленным. Такие запросы будут логироваться с уровнем `WARN`, что позволит легко выявить медленные запросы в системе. Чтобы включить эту функцию, добавьте параметр slowQueryThresholdMillis в *solrconfig.xml*. Например:

```
<requestHandler name="/select" class="solr.SearchHandler">
  <int name="slowQueryThresholdMillis">3000</int>
</requestHandler>
```

Все запросы, которые выполняются дольше указанного периода, будут логироваться как медленные на уровне `WARN`. Такие события записываются в отдельный лог-файл, который по умолчанию называется *solr_slow_requests.log*.

Logging UI

Вы можете увидеть логи Solr на странице Logging в Solr Admin UI, как показано ниже.

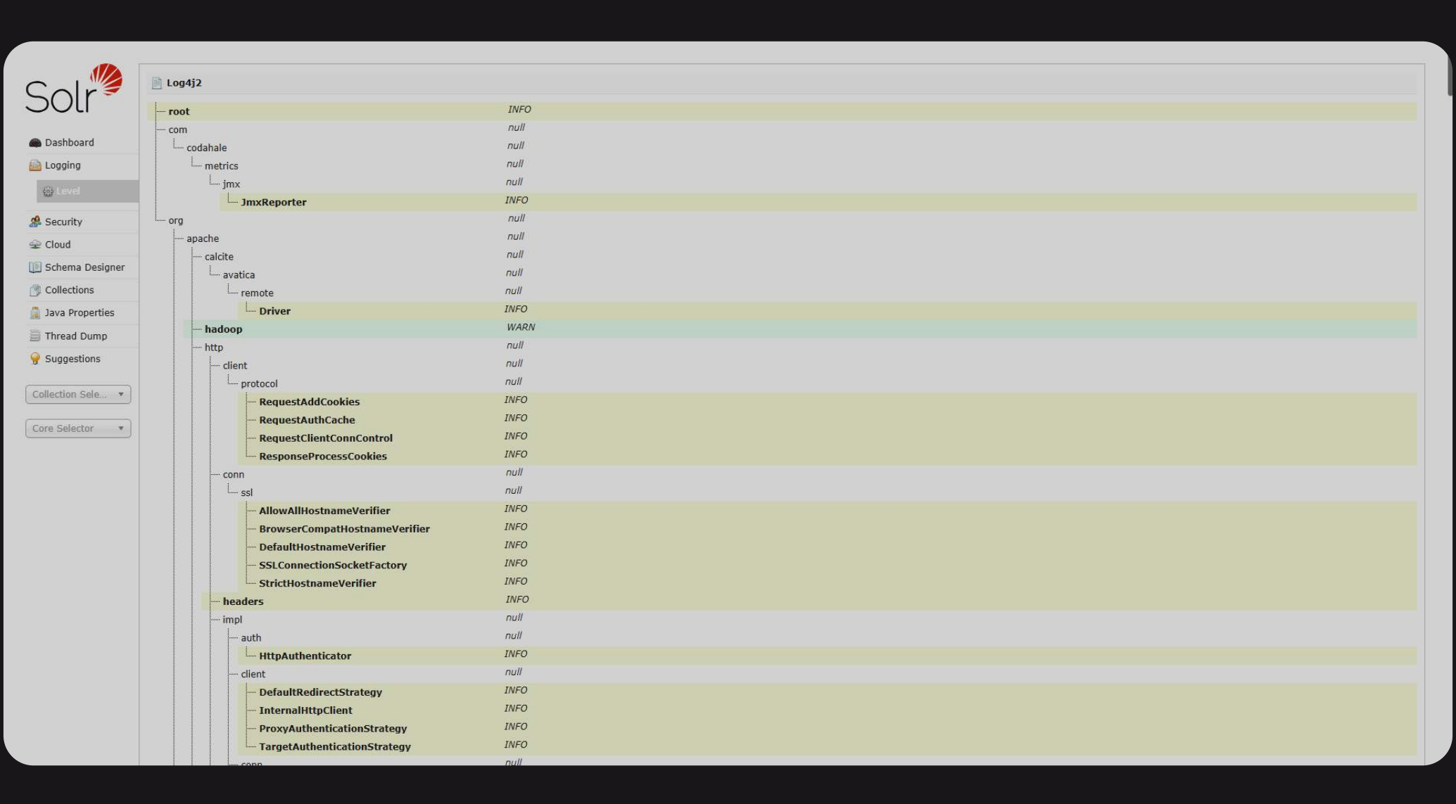


Time (Local)	Level	Core	Logger	Message
5/10/2024, 3:17:03 PM	WARN false		ZkController	Contents of zookeeper /security.json are world-readable; consider setting up ACLs as described in https://solr.apache.org/guide/zookeeper-access-control.html
5/10/2024, 3:17:03 PM	WARN false		CoreContainer	Not all security plugins configured! authentication=disabled authorization=disabled. Solr is only as secure as you make it. Consider configuring authentication/authorization before exposing Solr to users internal or external. See https://s.apache.org/solrsecurity for more info
5/10/2024, 3:17:21 PM	WARN false	x:solr_test_259611_shard1_replica_n1	SolrConfig	solrconfig.xml: <jmx> is no longer supported,&#8203; use solr.xml:/metrics/reporter section instead
5/10/2024, 4:52:29 PM	ERROR false		OverseerCollectionMessageHandler	Collection: test_collection operation: create failed
5/10/2024, 4:52:37 PM	WARN false	x:test_collection_shard1_replica_n1	SolrConfig	solrconfig.xml: <jmx> is no longer supported,&#8203; use solr.xml:/metrics/reporter section instead

Last Check: 5/10/2024, 5:52:29 PM

Страница Logging

Кроме того, используя вкладку ***Level***, можно контролировать объем информации, генерируемой отдельными компонентами и классами Solr. Для этого установите желаемый уровень логирования для компонентов Solr, доступных на странице. Обратите внимание, что узлы компонентов имеют иерархичную структуру, то есть вы можете установить уровень логирования для одного узла, и он будет унаследован всеми его дочерними компонентами. Это позволяет изменить сразу несколько категорий, изменяя уровень логирования одного родительского элемента.



```
root                                   INFO
com                                    null
  codahale                             null
    metrics                            null
      jmx                              null
        JmxReporter                    INFO
org                                    null
  apache                               null
    calcite                            null
      avatica                          null
        remote                         null
          Driver                       INFO
    hadoop                             WARN
    http                               null
      client                           null
        protocol                       null
          RequestAddCookies            INFO
          RequestAuthCache             INFO
          RequestClientConnControl     INFO
          ResponseProcessCookies       INFO
      conn                             null
        ssl                            null
          AllowAllHostnameVerifier     INFO
          BrowserCompatHostnameVerifier INFO
          DefaultHostnameVerifier      INFO
          SSLConnectionSocketFactory   INFO
          StrictHostnameVerifier       INFO
      headers                          INFO
      impl                             null
        auth                           null
          HttpAuthenticator            INFO
        client                         null
          DefaultRedirectStrategy      INFO
          InternalHttpClient           INFO
          ProxyAuthenticationStrategy  INFO
          TargetAuthenticationStrategy INFO
```

Страница Level

> **ПРИМЕЧАНИЕ**
>
> На этой странице можно изменить настройки логирования только для запущенного экземпляра Solr Server. Эти настройки не сохраняются для следующего запуска Solr.

Оптимизация производительности в Solr

Константин Алпашкин

Содержание

Назад к содержанию

- Обзор
- Настройка индекса
  - Параметры схемы
  - Размер индекса
  - Репликация и шардирование
- Память и параметры JVM
- Параметры коммитов
- Оптимизация запросов

Обзор

В данной статье описаны методы оптимизации и лучшие практики для повышения производительности вашего Solr-кластера.

> Не существует универсального руководства по улучшению производительности, которое бы подходило для всех Solr-кластеров. Solr состоит из множества настраиваемых компонентов, которые влияют друг на друга, поэтому изменение одного параметра конфигурации может повлиять на работу других. При настройке производительности Solr важно итеративно тестировать новые настройки и вносить лишь несколько изменений между тестами. Так можно быть уверенным, что вы движетесь в правильном направлении, а также сможете раньше выявить проблемные места.

Все рекомендации и оптимизации, описанные далее в статье, можно разделить на условные группы:

- Настройка индекса
- Память и параметры JVM
- Параметры коммитов
- Оптимизация запросов

Настройка индекса

Правильные настройки индексирования позволяют Solr эффективнее хранить данные документов в файлах индекса Lucene и обеспечивают более быстрое выполнение запросов при поиске.

Параметры схемы

- Выбирайте типы полей, которые наиболее точно соответствуют типу данных, хранимых в этих полях. Например, хранение дат в полях типа `DatePointField`, а не `StrField`, может улучшить производительность как на этапе индексирования, так и при поиске.
- Сведите к минимуму использование полей типа `text_general` (`TextField`) для хранения простых строк. Индексирование и поиск по полям такого типа запускает процессы токенизации/анализа для каждого запроса, что расходует вычислительную мощность. Вместо этого используйте поля типа `string` (`StrField`) для сохранения строк без дополнительной обработки.
- Минимизируйте использование динамических полей (dynamic fields). Используя динамические поля, Solr может индексировать произвольные типы данных, которые не были явно определены в схеме. Однако такая гибкость влечет за собой дополнительные накладные расходы. Например, использование динамических полей — оптимальный выбор, если поле не всегда содержит значение либо когда тип поля может меняться от документа к документу.
- Используйте docValues для полей, которые часто используются в сортировке или фасетировании (faceting).
- При использовании кастомных полей указывайте параметры полей с учетом того, как поле будет использоваться в будущем при поиске. Основные рекомендации:
  - Избегайте большого количества полей с параметром `multiValued`. Данный параметр указывает, что документ может содержать несколько значений для типа поля.
  - Не используйте свойство `stored=true`, если данное поле не используется для поиска.
  - Используйте свойство `indexed=false` для полей, которые не задействованы при поиске или фильтрации.

Размер индекса

- Избегайте записи избыточной и дублирующейся информации в индекс. Более подробная информация о предотвращении дубликатов доступна в статье Дедупликация в Solr.
- Измените политику слияния (merge policy), которая определяет стратегию перезаписи нескольких сегмент-файлов Lucene в один. По умолчанию `TieredMergePolicy` объединяет сегмент-файлы примерно одинакового размера, с учетом допустимого количества сегментов в директории.

Репликация и шардирование

- Используйте несколько Solr-шардов (shards), расположенных на разных хостах ADH, чтобы обрабатывать больше запросов одновременно. В индексе Solr, разбитом на шарды, документы распределяются по шардам таким образом, что каждый документ содержится только в одном шарде.
- Используйте репликацию для обеспечения высокой доступности и отказоустойчивости.

Память и параметры JVM

Одним из важнейших факторов, влияющих на производительность Solr, является оперативная память. Solr требует достаточно памяти для области Java heap, а также дополнительно нуждается в свободной памяти для выполнения операций кеширования на уровне операционной системы.

- Выделяйте достаточное количество памяти Java heap для каждого компонента Solr Server. Для этого используйте параметр ***Solr Server Heap Memory*** в ADCM (***Clusters → <clusterName> → Services → Solr → Primary configuration***). Однако просто выделение большего размера под heap не гарантирует ускорения производительности Solr. Heap Solr должен с запасом вмещать все внутренние объекты и кеши, оставляя при этом достаточно свободной памяти во избежание частых вызовов GC. Также в памяти хоста должно остаться место для операций кеширования на уровне ОС, чтобы кешировать данные индекса.
- Выделяемый под heap размер памяти должен быть равен 40-70% от всей оперативной памяти хоста, часть памяти всегда должна оставаться свободной. Помимо heap, Solr использует off-heap память для более быстрого кеширования/записи данных документов в файлы индекса Lucene. Слишком большой heap может не оставить достаточно памяти для выполнения этих операций, делая их узким местом.
- Анализируйте логи сборщика мусора (Garbage Collection, GC) для выявления аномалий и мониторинга использования памяти. По умолчанию Solr хранит логи GC в файлах */var/log/solr/solr_gc.log.**. Больше информации о логах Solr доступно в статье Логирование в Solr.

Параметры коммитов

Коммиты (commits) в Solr используются для того, чтобы сделать обновленные данные документа видимыми для поисковых запросов. Пока не выполнен коммит, обновления недоступны для поиска другими клиентами Solr. Solr выполняет два типа коммитов:

- **Жесткий коммит (hard commit)**. Делает обновления в документах видимыми для поиска, вызывая `fsync` для файлов индекса Lucene и записывая данные на диск. После выполнения жесткого коммита Solr прерывает текущий лог транзакций (transaction log) и открывает новый.
- **Мягкий коммит (soft commit)**. Делает изменения индекса видимыми для поиска без вызова `fsync` для сегмент-файлов Lucene. После выполнения мягкого коммита обновленные данные документа продолжают записываться в лог транзакций, однако еще не сохраняются в сегмент-файл.

Мягкие коммиты являются хорошей опцией для получения результатов поиска в режиме "почти реального времени" (Near Real Time, NRT), поскольку они делают обновления видимыми другим клиентам с небольшой задержкой, не дожидаясь дорогостоящего жесткого коммита. Жесткие коммиты необходимы для обеспечения надежности и гарантии того, что данные не будут потеряны, если хост внезапно выйдет из строя. Настройки коммитов можно изменить в файле *solrconfig.xml* (блоки `<autoCommit>` и `<autoSoftCommit>`).

Используйте следующие рекомендации по настройке механизма коммитов:

- Избегайте слишком частых жестких коммитов, так как их нагрузка на CPU и диск может сильно повлиять на общую производительность. Процесс записи на диск может стать основным узким местом при выполнении жестких коммитов, особенно если не хватает памяти для кеширования данных в page-кеше ОС. При выборе значения для автоматических жестких коммитов необходим баланс между как можно более редкими коммитами и соблюдением SLA доступности обновлений для поиска.
- Хотя мягкие коммиты Solr "дешевле" жестких, они все равно потребляют определенное количество ресурсов процессора и памяти. Мягкие коммиты рекомендуется выполнять чаще, чем жесткие, но при этом необходимо соблюдать баланс между задержкой в обновлениях данных и скоростью индексации.

Оптимизация запросов

- Сложные запросы могут замедлить работу Solr. Если ваш запрос содержит много фильтров и выполняется слишком долго, попробуйте разбить его на два более коротких.
- При необходимости изменяйте поведение кеширования запросов. Solr автоматически кеширует частые и повторяющиеся запросы. Время, затрачиваемое на создание такого кеша, называется временем разогрева кеша (cache warm-up). Однако, когда данные записываются в Solr, все кеши сбрасываются после каждого коммита и должны быть заново прогреты, что потребляет ресурсы. Используйте параметр `autowarmCount=0` для настроек `<filterCache>`, `<queryResultCache>` и `<documentCache>` в вашем *solrconfig.xml*, чтобы уменьшить количество циклов прогрева кеша.
- Не используйте слишком много фильтров в одном поисковом запросе.
- Избегайте использования дорогостоящих операций, таких как поиск по подстановочным знакам в начале термина.
- Используйте "ленивую" загрузку полей (параметр `<enableLazyFieldLoading>`), что позволяет Solr загружать поля, которые не запрашиваются напрямую, но могут понадобиться позже по мере необходимости. Это может повысить производительность, если большинство запросов запрашивают только небольшой список полей.

Управление сервисом Solr через ADCM

Константин Алпашкин

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Solr. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Solr, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Solr
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Manage Ranger plugin	Включает или отключает плагин Ranger для Solr
Update collection expiration	Обновляет время действия (поле `_expire_at_`) определенной коллекции Solr. При выполнении этого действия доступны следующие опции: ▪ ***Collection name*** — имя Solr-коллекции для обновления. ▪ ***Expire at*** — строка с меткой времени, определяющая время жизни проиндексированных документов в Solr. Например, `2023-07-28T18:05:03.371Z`. ▪ ***Batch size*** — количество документов, обрабатываемых в одном пакете (batch).
Validate configuration	Проверяет конфигурацию Solr (совместимость с ОС, конфликты портов) на всех хостах с компонентами Solr

spark3-submit и spark3-shell

Константин Алпашкин

Содержание

Назад к содержанию

Запуск через spark3-submit

Запуск через spark3-shell

Существует два способа запуска задач Spark (jobs) в кластере. Вы можете запустить задачу на любом хосте, где установлен Spark3 client:

- `spark3-submit`. Позволяет выполнять команды в интерактивном режиме, используя оболочку Scala, Python или R.
- `spark3-shell`. Создает приложение Spark, которое выполняется в интерактивном/batch режиме, используя Scala, Python или R.

Также вы можете использовать Livy для создания и управления Spark-приложениями в кластере. Livy — это сервис Spark, позволяющий локальным и удаленным приложениям взаимодействовать с Spark через открытый REST-интерфейс. Также Livy предоставляет дополнительные возможности в области безопасности и разделяемого доступа.

Запуск через spark3-submit

Чтобы запустить приложение Spark в кластере, используйте скрипт *bin/spark3-submit*. Такой подход позволяет использовать все типы планировщиков, которые поддерживает Spark, через единый интерфейс. Также это дает возможность не настраивать каждое приложение под отдельный планировщик.

Если код вашего приложения содержит зависимости из других проектов, необходимо включить все зависимости в архив приложения, чтобы оно могло отработать в кластере Spark. Для этого, создайте jar-файл (или uber-jar), который содержит код и зависимости. Вы можете использовать плагины *sbt* или *Maven*. При создании jar укажите тип зависимостей Spark и Hadoop как `provided`; эти зависимости необязательно включать в бандл, так как планировщик кластера предоставляет их во время выполнения.

После того, как ваш uber-jar собран, вы можете запустить jar, используя скрипт *bin/spark3-submit*. Данный скрипт автоматически настраивает путь к классам Spark и зависимостям. Скрипт поддерживает различные планировщики кластера и режимы деплоя.

Более детальную информации можно найти на странице Submitting Applications документации Spark.

Запуск через spark3-shell

Оболочка Spark позволяет быстро ознакомиться с API, а также служит мощным инструментом для интерактивного анализа данных. Вы можете начать работу с API в интерактивном режиме, запустив Scala-оболочку (скрипт *spark-shell*). Обратите внимание, что каждая запущенная оболочка создает новый объект `SparkContext` в переменной `sc`. Scala-оболочка работает в Java VM и имеет широкую поддержку Java-библиотек. Чтобы запустить оболочку, выполните следующую команду в директории Spark:

```
$ ./bin/spark3-shell
```

Пример запуска задач с Dataset

Константин Алпашкин

Содержание

Назад к содержанию

- Dataset
- Подготовка
- Запуск приложения
- Просмотр результатов

Пример программы, описанной в данной статье, позволяет проверить работоспособность Spark и показывает как запускать Spark задачи с помощью `spark3-submit`. Программа собирает статистику по криминальной ситуации в разных районах Бостона. Dataset, используемый в примере, можно получить по ссылке.

Dataset

Dataset содержит записи с 14 июня 2015 по 3 сентября 2018 года. Каждая строка представляет запись о преступлении, включая вид преступления, дату, время и место.

Программа принимает следующие файлы на входе:

- *crime.csv* — основной файл с записями о преступлениях;
- *offense_codes.csv* — файл, содержащий коды преступлений.

В результате выполнения программы мы получим следующую информацию:

- наиболее частые виды преступлений;
- распределение преступлений по районам;
- частота совершения преступления по видам.

Данные файлы мы помещаем в директорию с проектом.

Подготовка

Для получения информации по созданию проекта Scala с использованием инструмента сборки Scala обратитесь к документации Scala.

> **ПРИМЕЧАНИЕ**
>
> Если проект создается локально, не забудьте скопировать jar файл в мастер-хост Spark.

Используя Spark, мы агрегируем все районы (поле `district`) по следующими метрикам:

- `crimes_total` — общее количество преступлений, совершенных в указанном районе.
- `crimes_monthly` -- усредненное количество преступлений в месяц, совершенных в указанном районе.
- `frequent_crime_types` — три наиболее частых вида преступлений в указанном районе за всю историю наблюдений. Значения разделены запятой и пробелом (`, `) и упорядочены по нисходящей от наиболее частых к редким.
- `crime_type` — первая часть поля `NAME` из таблицы *offense_codes.csv*, отсеченная разделителем `-`. Например, если поле `NAME` содержит `BURGLARY - COMMERICAL - ATTEMPT`, то `crime_type` определяется как `BURGLARY`.
- `lat` — широта координат района, средняя по всем преступлениям.
- `lng` — долгота координат района, средняя по всем преступлениям.

Вы можете использовать следующие примеры кода для сборки Scala проекта.

Основной пример кода Scala

```scala
package com.example

import com.example.entity.{Crime, OffenseCode}
import org.apache.spark.sql._
import org.apache.spark.sql.functions._

object BostonCrimesMap extends App {

    val crimeFilepath = args(0)
    val offenseCodesFilePath = args(1)
    val resultFilePath = args(2)

  val spark = SparkSession
    .builder()
    .master("local[*]")
    .appName("boston_crimes")
    .getOrCreate()

  import spark.implicits._

  val crimes = spark
    .read
    .option("header", "true")
    .option("inferSchema", "true")
    .csv(crimeFilepath)
    .as[Crime]


  val offense_codes = spark
    .read
    .option("header", "true")
    .option("inferSchema", "true")
    .csv(offenseCodesFilePath)
    .withColumn("CRIME_TYPE", trim(substring_index($"NAME", "-", 1)))
    .as[OffenseCode]


  val offense_codes_br = spark.sparkContext.broadcast(offense_codes)

  val filteredCrimes = crimes
    .filter($"DISTRICT".isNotNull).cache()

  val crimesWithOffenceCodes = filteredCrimes
    .join(offense_codes_br.value, filteredCrimes("OFFENSE_CODE") === offense_codes_br.value("CODE"))
    .select("INCIDENT_NUMBER", "DISTRICT", "MONTH", "Lat", "Long", "CRIME_TYPE").cache()

  val crimesDistrictAnalytics = filteredCrimes
    .groupBy($"DISTRICT")
    .agg(expr("COUNT(INCIDENT_NUMBER) as crimes_total"),
      expr("AVG(Lat) as lat"),
      expr("AVG(Long) as lng")
    )


  val crimesByDistrictByMonth = filteredCrimes
    .groupBy($"DISTRICT", $"MONTH")
    .agg(expr("count(INCIDENT_NUMBER) as CRIMES_CNT")).createOrReplaceTempView("crimesByDistrictByMonth")

  val crimesDistrictMedian = spark.sql(
    "select " +
      " DISTRICT" +
      " ,percentile(CRIMES_CNT, 0.5) as crimes_monthly " +
      " from crimesByDistrictByMonth" +
      " group by DISTRICT")


  val crimesByDistrictByCrimeTypes = crimesWithOffenceCodes
    .groupBy($"DISTRICT", $"CRIME_TYPE")
    .agg(expr("count(INCIDENT_NUMBER) as CRIMES_CNT"))
    .selectExpr("*", "row_number() over(partition by DISTRICT order by CRIMES_CNT desc) as rn")
    .filter($"rn" <= 3)
    .drop($"rn")
    .drop($"CRIMES_CNT")
    .groupBy($"DISTRICT")
    .agg(concat_ws(", ", collect_list($"CRIME_TYPE")).alias("frequent_crime_types"))


  val finalResult =
    crimesDistrictAnalytics
      .join(crimesDistrictMedian, "DISTRICT")
      .join(crimesByDistrictByCrimeTypes, "DISTRICT")
      .select($"DISTRICT", $"crimes_total", $"crimes_monthly", $"frequent_crime_types", $"lat", $"lng")

  finalResult.repartition(1).write.mode("OVERWRITE").parquet(resultFilePath)


}
```

Пример build.sbt

```scala
lazy val root = (project in file(".")).

  settings(
    inThisBuild(List(
      organization := "com.example",
      scalaVersion := "2.11.8"
    )),
    name := "boston_crimes",
    version := "0.0.1",

    sparkVersion := "2.3.0",
    sparkComponents := Seq(),

    javacOptions ++= Seq("-source", "1.8", "-target", "1.8"),
    javaOptions ++= Seq("-Xms512M", "-Xmx2048M", "-XX:MaxPermSize=2048M", "-XX:+CMSClassUnloadingEnabled"),
    scalacOptions ++= Seq("-deprecation", "-unchecked"),
    parallelExecution in Test := false,
    fork := true,

    coverageHighlighting := true,

    libraryDependencies ++= Seq(
      "org.apache.spark" %% "spark-streaming" % "2.3.0" % "provided",
      "org.apache.spark" %% "spark-sql" % "2.3.0" % "provided",

      "org.scalatest" %% "scalatest" % "3.0.1" % "test",
      "org.scalacheck" %% "scalacheck" % "1.13.4" % "test",
      "com.holdenkarau" %% "spark-testing-base" % "2.3.0_0.9.0" % "test"
    ),

    // uses compile classpath for the run task, including "provided" jar (cf http://stackoverflow.com/a/21803413/3827)
    run in Compile := Defaults.runTask(fullClasspath in Compile, mainClass in (Compile, run), runner in (Compile, run)).evaluated,

    scalacOptions ++= Seq("-deprecation", "-unchecked"),
    pomIncludeRepository := { x => false },

   resolvers ++= Seq(
      "sonatype-releases" at "https://oss.sonatype.org/content/repositories/releases/",
      "Typesafe repository" at "http://repo.typesafe.com/typesafe/releases/",
      "Second Typesafe repo" at "http://repo.typesafe.com/typesafe/maven-releases/",
      Resolver.sonatypeRepo("public")
    ),

    pomIncludeRepository := { x => false },

    // publish settings
    publishTo := {
      val nexus = "https://oss.sonatype.org/"
      if (isSnapshot.value)
        Some("snapshots" at nexus + "content/repositories/snapshots")
      else
        Some("releases"  at nexus + "service/local/staging/deploy/maven2")
    }
  )
```

Пример класса Crime.scala

```scala
package com.example.entity

case class Crime(
                  INCIDENT_NUMBER: Option[String],
                  OFFENSE_CODE: Option[Int],
                  OFFENSE_CODE_GROUP: Option[String],
                  OFFENSE_DESCRIPTION: Option[String],
                  DISTRICT: Option[String],
                  REPORTING_AREA: Option[String],
                  SHOOTING: Option[String],
                  OCCURRED_ON_DATE: Option[String],
                  YEAR: Option[Int],
                  MONTH: Option[Int],
                  DAY_OF_WEEK: Option[String],
                  HOUR: Option[Int],
                  UCR_PART: Option[String],
                  STREET: Option[String],
                  Lat: Option[Double],
                  Long: Option[Double],
                  Location: Option[String]
                )
```

Пример класса OffenseCode.scala

```scala
package com.example.entity

case class OffenseCode(
                      CODE: Option[Int],
                      NAME: Option[String],
                      CRIME_TYPE: Option[String]
                      )
```

Чтобы построить uber-jar, выполните следующую команду в директории проекта:

```
$ sbt clean assembly
```

Запуск приложения

После того, как uber-jar готов, убедитесь, что вы поместили jar в ваш кластер, используя Spark-сервер. В директории проекта запустите программу, используя следующую команду:

```
$ spark3-submit --master local --class com.example.BostonCrimesMap path-to/boston_crimes.jar path-to/crime.csv path-to/offense_codes.csv path-to/result
```

Просмотр результатов

Все результаты сохраняются в директорию, указанную при вызове команды `spark3-submit`, и представлены в parquet-формате.

Пример результата выполнения

```
{"DISTRICT":"C6","crimes_total":23460,"crimes_monthly":1870,"frequent_crime_types":"SICK/INJURED/MEDICAL, DRUGS, INVESTIGATE PERSON","lat":42.21212258445548,"lng":-70.85561011772236}
{"DISTRICT":"B2","crimes_total":49945,"crimes_monthly":3986,"frequent_crime_types":"VERBAL DISPUTE, SICK/INJURED/MEDICAL, INVESTIGATE PERSON","lat":42.31600367732765,"lng":-71.0756993065433}
{"DISTRICT":"C11","crimes_total":42530,"crimes_monthly":3284,"frequent_crime_types":"SICK/INJURED/MEDICAL, INVESTIGATE PERSON, VERBAL DISPUTE","lat":42.29263740900067,"lng":-71.05125995734362}
{"DISTRICT":"E13","crimes_total":17536,"crimes_monthly":1368.5,"frequent_crime_types":"SICK/INJURED/MEDICAL, INVESTIGATE PERSON, DRUGS","lat":42.309803655709956,"lng":-71.09800478878388}
{"DISTRICT":"B3","crimes_total":35442,"crimes_monthly":2764,"frequent_crime_types":"VERBAL DISPUTE, INVESTIGATE PERSON, MISSING PERSON","lat":42.283059445201054,"lng":-71.07894914185492}
{"DISTRICT":"E5","crimes_total":13239,"crimes_monthly":1043.5,"frequent_crime_types":"SICK/INJURED/MEDICAL, INVESTIGATE PERSON, PROPERTY","lat":42.19796999447013,"lng":-71.00440862434752}
{"DISTRICT":"A15","crimes_total":6505,"crimes_monthly":499,"frequent_crime_types":"INVESTIGATE PERSON, VANDALISM, SICK/INJURED/MEDICAL","lat":42.17915525091085,"lng":-70.74472508958515}
{"DISTRICT":"A7","crimes_total":13544,"crimes_monthly":1092,"frequent_crime_types":"SICK/INJURED/MEDICAL, INVESTIGATE PERSON, VANDALISM","lat":42.36070260499386,"lng":-71.00394833039842}
{"DISTRICT":"D14","crimes_total":20127,"crimes_monthly":1607.5,"frequent_crime_types":"TOWED MOTOR VEHICLE, SICK/INJURED/MEDICAL, INVESTIGATE PERSON","lat":42.343507245109336,"lng":-71.13125461726499}
{"DISTRICT":"D4","crimes_total":41915,"crimes_monthly":3297.5,"frequent_crime_types":"PROPERTY, INVESTIGATE PERSON, SICK/INJURED/MEDICAL","lat":42.34124251790852,"lng":-71.07725024946983}
{"DISTRICT":"E18","crimes_total":17348,"crimes_monthly":1313,"frequent_crime_types":"SICK/INJURED/MEDICAL, INVESTIGATE PERSON, VERBAL DISPUTE","lat":42.26268061122604,"lng":-71.11891998757692}
{"DISTRICT":"A1","crimes_total":35717,"crimes_monthly":2775,"frequent_crime_types":"PROPERTY, SICK/INJURED/MEDICAL, WARRANT ARREST","lat":42.33123077259828,"lng":-71.01991881362002}
```

Пример запуска задач с RDD

Константин Алпашкин

Содержание

Назад к содержанию

- Операции с RDD
  - Создание
  - Модификация
  - Просмотр содержимого
- Пример

Spark resilient distributed dataset (RDD) — это отказоустойчивая коллекция, элементы которой распределены по узлам кластера и могут обрабатываться параллельно.

В данной статье показаны базовые примеры использования RDD на языке Scala, которые можно запускать в spark3-shell. Также Spark поддерживает другие языки, например Java, Python и R.

Операции с RDD

Создание

Существует два способа создания нового RDD, примеры представлены ниже.

Используя метод parallelize()

Для создания RDD из коллекции можно использовать метод `sparkContext.parallelize()` в коде программы драйвера.

```
val dataSeq = Seq(("foo", 0), ("bar", 1), ("buzz", 2))
val rdd=sc.parallelize(dataSeq)

val dataArr = Array(0, 1, 2, 3)
val rdd1=sc.parallelize(dataArr)

val emptyRDD = sc.parallelize(Seq.empty[String])
```

Из внешнего источника данных, например, HDFS

Данные для RDD можно получить из внешнего хранилища, например, HDFS, HBase или любого другого источника данных, реализующего Hadoop InputFormat. Ниже показан способ чтения данных из HDFS.

```
val rdd2 = sc.textFile("hdfs://adh/user/admin/testload/file_test.csv")
```

После создания объекта RDD вы можете выполнять основные операции Spark над RDD.

Модификация

Все операции RDD делятся на два типа:

- **Трансформации (transformations)**, которые возвращают новый RDD. Например, `map()`, `flatMap()`, `union()` и так далее.
- **Действия (actions)**, которые возвращают некое значение программе-драйверу после выполнения вычислений над RDD. Например, `collect()`, `count()` и так далее.

Все операции с RDD, включая создание и модификацию, являются "ленивыми" (lazy), то есть они не выполняются сразу. Вычисления производятся только тогда, когда какой-то результат должен быть возвращен программе-драйверу, например, вызов `rdd.take(10)`.

Полная справочная информация о поддерживаемых операциях доступна на странице Spark RDD API.

Просмотр содержимого

Для просмотра содержимого RDD используйте метод `rdd.take(x)`, который возвращает первые `x` элементов RDD.

ПРИМЕЧАНИЕ

Используйте с осторожностью конструкции типа `rdd.collect().foreach(println)`, так как это может привести к нехватке памяти на машине-драйвере Spark, поскольку `collect()` загружает данные RDD со всех узлов кластера на одну машину.

Пример

Следующий пример демонстрирует использование основных операций с RDD. Чтобы пример отработал без ошибок, соответствующий файл с данными (*test_lorem.txt*) должен быть доступен в HDFS. Создание тестового файла показано ниже.

1. В локальной файловой системе создайте файл *test_lorem.txt*:

```
echo "Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec, vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo. Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus elementum semper nisi. Aenean vulputate eleifend tellus." > test_lorem.txt
```

2. Создайте директорию в HDFS, установите права доступа и скопируйте тестовый файл в HDFS:

```
$ sudo -u hdfs hdfs dfs -mkdir /user/admin
$ sudo -u hdfs hdfs dfs -chown admin:admin /user/admin
$ sudo -u hdfs hdfs dfs -put /home/admin/spark-demo/test_lorem.txt /user/admin/test_lorem.txt
```

Необходимо убедиться, что пользователь `hdfs` имеет доступ к файлу *test_lorem.txt*.

Когда тестовые данные загружены в HDFS, вы можете выполнить следующий Scala-код непосредственно в spark3-shell (*/bin/spark-shell*).

```
val rdd = sc.textFile("hdfs://adh/user/admin/test_lorem.txt")
println("Created RDD:")
rdd.take(10)

val resultRDD = rdd.flatMap(line => line.split(" ")) ①
      .map(word => (word,1)) ②
      .reduceByKey(_+_) ③
      .sortBy(_._2, false) ④

resultRDD.take(100).foreach(println)

resultRDD.persist() ⑤
println("Writing RDD to a file: /user/admin/lorem_demo_result")
resultRDD.saveAsTextFile("hdfs://adh/user/admin/lorem_demo_result") ⑥
```

① Разбиение каждой строки текста исходного файла на отдельные слова.

② Создание кортежа `(<word>,1)` для каждого слова.

③ Редуцирование коллекции таким образом, чтобы остались лишь уникальные слова с количеством повторений.

④ Сортировка слов по количеству повторений.

⑤ Кеширование RDD в памяти. В этот момент выполняется синхронизация RDD по всем узлам кластера.

⑥ Запись RDD в HDFS. В HDFS создается директория с частями (parts) данных RDD.

В результате выполнения примера в консоль выводится следующий результат.

```
Created RDD:
val res0: Array[String] = Array(Lorem ipsum dolor sit amet, ...)

val resultRDD: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[9] at sortBy at <console>:4

(Aenean,3)
(vulputate,2)
(pede,2)
(nec,,2)
(Donec,2)
(justo,,2)
(montes,,1)
...

Writing RDD to a file: /user/admin/lorem_demo_result
```

Запись RDD в HDFS в примере выше можно проверить с помощью команды:

```
$ sudo -u hdfs hdfs dfs -ls /user/admin/lorem_demo_result
```

Пример вывода:

```
Found 3 items
-rw-r--r--   3 admin admin          0 2023-07-25 20:55 /user/admin/lorem_demo_result/_SUCCESS
-rw-r--r--   3 admin admin         64 2023-07-25 20:55 /user/admin/lorem_demo_result/part-00000
-rw-r--r--   3 admin admin        753 2023-07-25 20:55 /user/admin/lorem_demo_result/part-00001
```

Пример запуска задач с DataFrame

Константин Алпашкин

Содержание

Назад к содержанию

Операции с DataFrame
- Создание
- Модификация
- Просмотр содержимого
- Выполнение SQL-запросов

Пример

Spark DataFrame (DF) — это структура данных, организованная в именованные столбцы и распределенная по кластеру. Концептуально DF похож на таблицу в реляционной базе данных, но с определенными оптимизациями. Технически Spark DataFrame представляет собой DataSet, хранящий данные типа Row.

В данной статье показаны базовые примеры использования DF на языке Scala, которые можно запускать в spark3-shell. Также Spark поддерживает другие языки, например Java, Python и R.

Операции с DataFrame

Создание

Объект DataFrame может быть создан из различных источников, наиболее распространенные способы представлены ниже.

Из структурированных данных

Ниже показано создание DF из JSON-файла, хранящегося в HDFS.

```
val df = spark
    .read
    .json("hdfs://adh/user/admin/testload/people.json")

// read multiline JSON
val multiline_df = spark
    .read
    .option("multiline","true")
    .json("hdfs://adh/user/admin/testload/people_multiline.json")
```

В данном примере схема для DF определяется автоматически на основе структуры JSON-файла. Для получения дополнительной информации о создании DF из других источников, таких как Avro, Parquet, ORC и многих других, смотрите доступные источники данных.

Из RDD

Вы можете преобразовать RDD в DF с помощью метода `rdd.toDF()`. По умолчанию Spark генерирует схему автоматически на основе структуры RDD, однако вы также можете указать схему явно.

```
val rdd = sc.textFile("hdfs://adh/user/admin/testload/test_data_rdd.txt")
val df = rdd.toDF()
```

Из внешней базы данных, используя JDBC

Пример ниже демонстрирует получение DF из таблицы MySQL. Чтобы код примера заработал, в classpath должен быть доступен JAR MySQL-драйвера.

```
val df_mysql = spark.read.format("jdbc")
   .option("url", "jdbc:mysql://<host>:<port>/db_name")
   .option("driver", "com.mysql.jdbc.Driver")
   .option("dbtable", "<table_name>")
   .option("user", "<username>")
   .option("password", "<password>")
   .load()
```

Получение DF из других СУБД выглядит аналогично.

Помимо вышеупомянутых, существует множество других способов получения объекта DataFrame, например, с помощью методов `SparkSession.sql()` / `SparkSession.createDataFrame()` или запроса к таблице Hive/HBase и так далее. Более подробная информация об этих методах доступна в документации Spark.

Модификация

Все операции DataFrame делятся на два типа:

- **Трансформации (transformations)**, которые возвращают новый DF. Например, `map()`, `filter()`, `agg()`, `col()`, `join()` и так далее.
- **Действия (actions)**, которые возвращают некое значение программе-драйверу после выполнения вычислений над DF. Например, `collect()`, `count()` и так далее.

Все операции с DF являются "ленивыми" (lazy), то есть они не выполняются сразу. Вычисления производятся только тогда, когда какой-то результат должен быть возвращен программе-драйверу, например, вызов `df.show()`.

Полная справочная информация о поддерживаемых операциях доступна на странице Spark DataSet API.

Просмотр содержимого

Для просмотра содержимого DF используйте метод `df.show(x)`, который возвращает первые `x` строк DF.

> **ПРИМЕЧАНИЕ**
>
> Используйте с осторожностью конструкции типа `df.collect().foreach(println)`, так как это может привести к нехватке памяти на машине-драйвере Spark, поскольку `collect()` загружает данные DF со всех узлов кластера на одну машину.

Для получения информации о DF-схеме, используйте `df.printSchema()` как показано ниже.

```
root
 |-- age: long (nullable = true)
 |-- dept: string (nullable = true)
 |-- id: long (nullable = true)
 |-- lastname: string (nullable = true)
 |-- name: string (nullable = true)
```

Выполнение SQL-запросов

Модуль Spark SQL позволяет выполнять традиционные SQL-запросы к DF. Для этого необходимо создать временное представление (View) из DataFrame с помощью метода `createOrReplaceTempView()`. Затем вы можете отправлять SQL-запросы к этому представлению, используя метод `SparkSession.sql()`. Пример приведен ниже.

```
val df = spark
    .read
    .csv("hdfs://adh/user/admin/testload/employees.json")

df.createOrReplaceTempView("employees")

val df_sql = spark.sql("SELECT name, lastname, email FROM employees")
df_sql.show()
```

После создания временное представление ведет себя как обычная таблица SQL, однако оно нигде не сохраняется в виде файла. Представление привязано к сессии SparkSession, в которой оно было создано, и остается доступным до тех пор, пока существует объект SparkSession. Больше информации о выполнении SQL-запросов доступно в разделе SparkSession.sql() API.

Пример

Следующий пример демонстрирует использование основных операций с DF. Чтобы пример отработал без ошибок, соответствующий файл с данными (*employees.json*) должен быть доступен в HDFS. Создание тестового файла показано ниже.

1. В локальной файловой системе создайте файл *employees.json*:

```
echo -e
'{"id":1,"name":"Sarah","lastname":"Connor","dept":"hr","age":34}\n{"id":2,"name":"Michael","lastname":"Scott","dept":"sales","age
":47}\n{"id":3,"name":"Luke","lastname":"Skywalker","dept":"it","age":45}\n{"id":4,"name":"James","lastname":"Hetfield","dept":"sa
les","age":50}' > employees.json
```

2. Создайте директорию в HDFS, установите права доступа и скопируйте тестовый файл в HDFS:

```
$ sudo -u hdfs hdfs dfs -mkdir /user/admin
$ sudo -u hdfs hdfs dfs -chown admin:admin /user/admin
$ sudo -u hdfs hdfs dfs -put /home/admin/spark-demo/employees.json /user/admin/employees.json
```

Необходимо убедиться, что пользователь `hdfs` имеет доступ к файлу *employees.json*.

Когда тестовые данные загружены в HDFS, вы можете выполнить следующий Scala-код непосредственно в spark3-shell (*/bin/spark-shell*).

```
val df_employees = spark.read.json("hdfs://adh/user/admin/employees.json") ①

println("DataFrame created:")
df_employees.show()

println("DataFrame schema:")
df_employees.printSchema()

println("Show name-age selection from all employees:")
df_employees.select("name","age").show()

println("Get all employees older than 40")
df_employees.filter("age > 40").show()


val depts_data = Seq((1,"it"),(2,"sales"),(3,"hr"))
val rdd = sc.parallelize(depts_data)
val df_department = rdd.toDF("id","department") ②

println("DataFrame created:")
df_department.show()

val joined = df_employees.filter("age > 30").join(df_department, df_employees("dept") === df_department("department")) ③

println("Get number of employees per department")
joined.groupBy("department").count().show()

println("Get average age per department")
joined.groupBy("department").mean("age").show() ④
```

① Создание DF из JSON-файла в HDFS.

② Создание DF из RDD.

③ Фильтрация по столбцу `age` и выполнение `JOIN` для двух DF.

④ Использование `GROUP BY` с агрегацией по столбцу.

В результате выполнения примера в консоль выводится следующий результат:

```
DataFrame created:
+---+-----+---+---------+-------+
|age| dept| id| lastname|   name|
+---+-----+---+---------+-------+
| 34|   hr|  1|   Connor|  Sarah|
| 47|sales|  2|    Scott|Michael|
| 45|   it|  3|Skywalker|   Luke|
| 50|sales|  4| Hetfield|  James|
+---+-----+---+---------+-------+

DataFrame schema:
root
 |-- age: long (nullable = true)
 |-- dept: string (nullable = true)
 |-- id: long (nullable = true)
 |-- lastname: string (nullable = true)
 |-- name: string (nullable = true)

Show name-age selection from all employees:
+-------+---+
|   name|age|
+-------+---+
|  Sarah| 34|
|Michael| 47|
|   Luke| 45|
|  James| 50|
+-------+---+

Get all employees older than 40:
+---+-----+---+---------+-------+
|age| dept| id| lastname|   name|
+---+-----+---+---------+-------+
| 47|sales|  2|    Scott|Michael|
| 45|   it|  3|Skywalker|   Luke|
| 50|sales|  4| Hetfield|  James|
+---+-----+---+---------+-------+

DataFrame created:
+---+----------+
| id|department|
+---+----------+
|  1|        it|
|  2|     sales|
|  3|        hr|
+---+----------+

Get number of employees per department:
+----------+-----+
|department|count|
+----------+-----+
|     sales|    2|
|        hr|    1|
|        it|    1|
+----------+-----+

Get average age per department:
+----------+--------+
|department|avg(age)|
+----------+--------+
|     sales|    48.5|
|        hr|    34.0|
|        it|    45.0|
+----------+--------+
```

Spark и Hive

Константин Алпашкин

Содержание

Назад к содержанию

[Кастомная настройка](#)

По умолчанию Spark может работать с Hive из коробки. Настройки для взаимодействия с Hive находятся в домашней директории — */etc/spark3/conf*. Если ваше Spark-приложение взаимодействует с Hadoop, или же и с Hadoop и Spark, необходимо добавить файлы конфигурации Hadoop в Spark classpath.

Несколько одновременно запущенных приложений могут потребовать индивидуальной клиентской конфигурации Hadoop/Hive. Для каждого отдельного приложения вы можете модифицировать файлы *hdfs-site.xml*, *core-site.xml*, *yarn-site.xml* и поместить их в Spark classpath.

ПРИМЕЧАНИЕ

Для кластеров Spark с YARN вышеупомянутые конфигурационные файлы устанавливаются на уровне кластера и не могут быть изменены приложением.

Наиболее простым способом указания параметров для Spark Hadoop является нотация вида `spark.hadoop.*` и `spark.hive.*` для Hive. Например, добавление `spark.hadoop.abc.def=xyz` аналогично Hadoop-параметру `abc.def=xyz`, а `spark.hive.abc=xyz` эквивалентно Hive-параметру `hive.abc=xyz`. Параметры, заданные в таком виде, является обычными Spark-параметрами, которые можно объявлять в файле *$SPARK_HOME/conf/spark-defaults.conf*. По умолчанию домашней директорией для Spark является */etc/spark3/conf*, где вы можете хранить все настройки для Spark.

Иногда, чтобы не записывать определенные настройки в SparkConf, вы можете добавлять или переопределять определенные параметры при старте программы:

```
./bin/spark3-submit \
  --name "My app" \
  --master local[4] \
  --conf spark.eventLog.enabled=false \
  --conf "spark.executor.extraJavaOptions=-XX:+PrintGCDetails -XX:+PrintGCTimeStamps" \
  --conf spark.hadoop.abc.def=xyz \
  --conf spark.hive.abc=xyz
  myApp.jar
```

Описания всех Spark-параметров доступны на [странице документации Spark](#).

Кастомная настройка

Если вы хотите изменить настройку вашего Spark-приложения для Hive, существует 2 подхода:

- Добавить кастомные параметры в файл *spark-defaults.conf* и поместить этот файл в classpath Hive.
- Добавить кастомные параметры в конфигурационный файл *hive-site.xml*, который использует Hive. Все файлы конфигурации хранятся в домашней директории Spark — */etc/spark3/conf/hive-site.xml*.

Дополнительная информация о параметрах Hive для Spark доступна в разделе [Параметры Hive on Spark](#).

ADB Spark 3 Connector

Дарья Барышева

Назад к содержанию

Arenadata DB (ADB) — это распределенная аналитическая СУБД, построенная на базе MPP-системы с открытым исходным кодом Greenplum.

Apache Spark 3 — это открытый фреймворк, разработанный для распределенной обработки неструктурированных либо слабо структурированных данных.

ADB Spark 3 Connector обеспечивает высокоскоростной параллельный обмен данными между Spark 3 и Arenadata DB.

В следующих статьях содержится полное описание ADB Spark 3 Connector:

- Обзор
- Требования к установке
- Примеры использования
- Опции

ВАЖНО

Начиная с версии ADH 3.1.2.1.b1 коннектор для Spark 3 поставляется вместе с бандлом ADH. Для использования коннектора необходимо установить сервис Spark 3. Порядок работы с коннектором для Spark 2 описан в отдельном руководстве ADB Spark Connector.

Обзор ADB Spark 3 Connector

Михаил Серов

Содержание

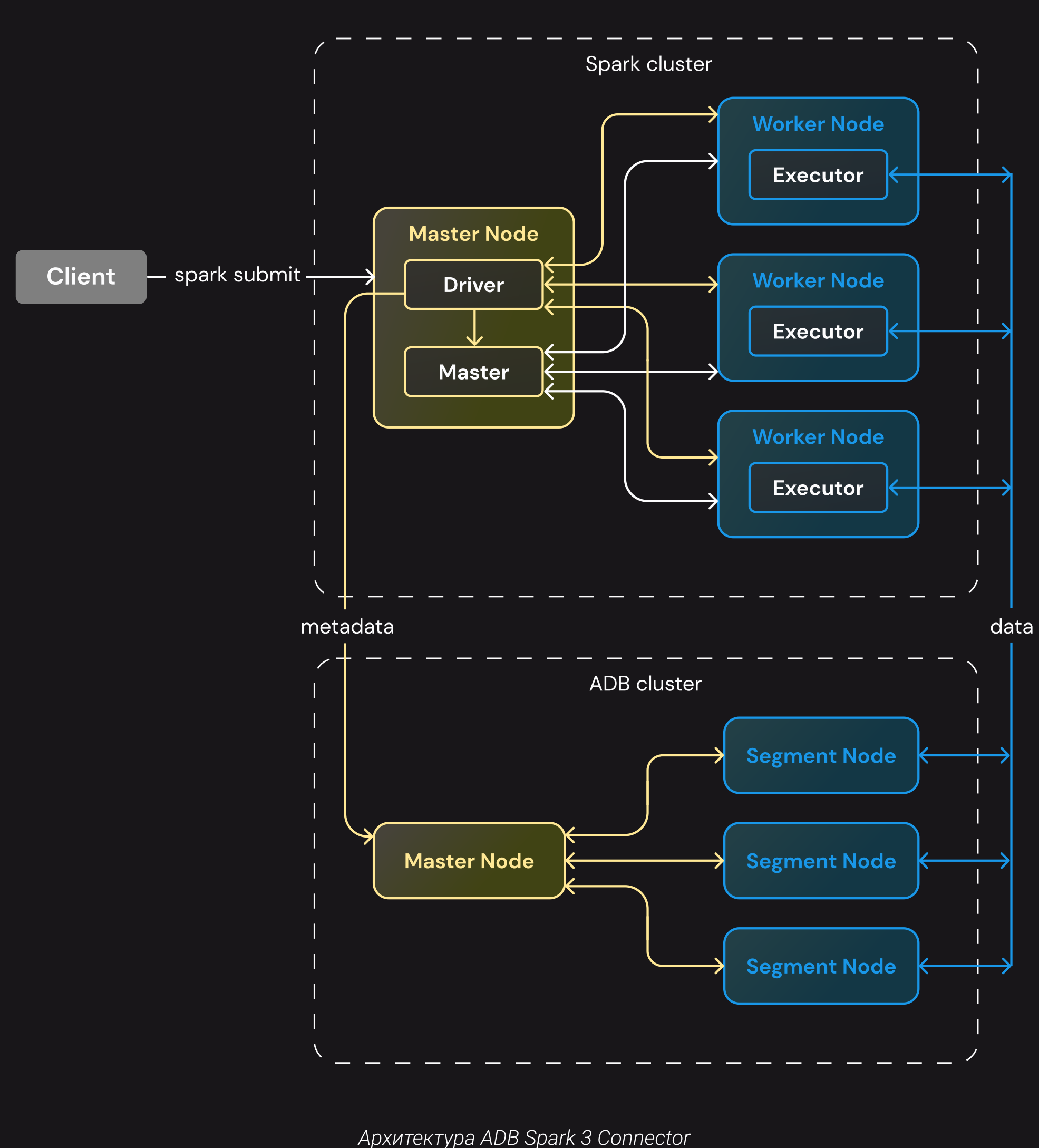
Назад к содержанию

- Архитектура
- Алгоритмы работы
  - Чтение данных
  - Запись данных
- Поддерживаемые типы данных
- Поддерживаемые драйверы JDBC
- Использование коннектора

ADB Spark 3 Connector предоставляет возможность высокоскоростного параллельного обмена данными между Apache Spark 3 и Arenadata DB.

Архитектура

Каждое приложение Spark 3 состоит из контролирующего процесса (driver) и некоторого количества распределённых worker-процессов (executors). Схема взаимодействия представлена ниже.



Архитектура ADB Spark 3 Connector

Алгоритмы работы

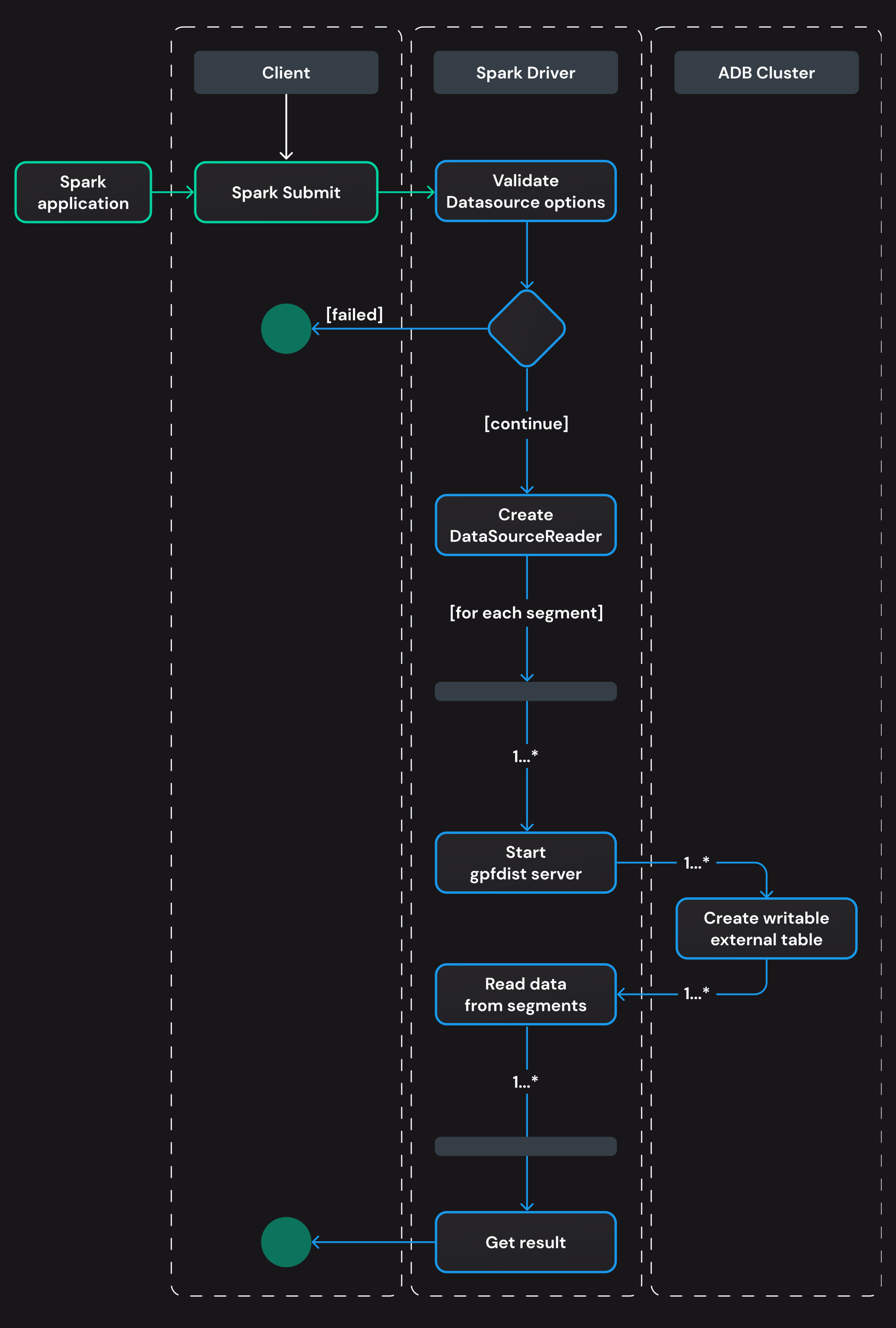
Чтение данных

ADB Spark 3 Connector инициализирует драйвер, чтобы загрузить таблицу ADB в Spark 3. Этот драйвер устанавливает соединение с ADB master через JDBC, чтобы получить необходимые метаданные о таблице. Метаданные содержат информацию о типе, структурах таблиц и ключе дистрибуции. Ключ дистрибуции позволяет вам распределить данные из таблицы на существующие ноды Spark 3.

Данные хранятся в сегментах ADB. Приложение Spark 3, которое использует коннектор, может перемащать данные из каждого сегмента в одну или несколько партиций Spark 3. Существует пять режимов партиционирования:

- **В соответствии с gp_segment_id**. Данные из таблиц ADB прочитываются и распределяются между партициями Spark 3 в зависимости от сегмента. Количество партиций Spark 3 соответствует количеству активных сегментов ADB. Этот режим партиционирования не требует дополнительных параметров и используется по умолчанию.
- **В соответствии с указанными столбцом и количеством партиций**. Данные распределяются между партициями Spark 3 внутри столбцевого диапазона согласно указанному количеству партиций. Количество партиций Spark 3 соответствует количеству рядов. Необходимо установить подходящие параметры (см. Опции ADB Spark 3 Connector). У этого режима партиционирования есть ограничения: в качестве столбца могут использоваться только поля таблицы типа `integer` или `date/time`.
- **В соответствии с указанным столбцом**. Данные распределяются между партициями Spark 3 в зависимости от уникальных значений в указанном столбце. Количество партиций Spark 3 соответствует количеству уникальных значений в указанном столбце. Необходимо установить подходящие параметры (см. Опции ADB Spark 3 Connector). Этот режим рекомендуется для случая малого и ограниченного набора значений.
- **В соответствии с указанным количеством партиций**. Данные распределяются между партициями Spark 3 в зависимости от указанного количества партиций. Необходимо установить подходящие параметры (см. Опции ADB Spark 3 Connector).
- **В соответствии с указанными хеш-функцией и количеством партиций**. Данные распределяются между партициями Spark 3 в зависимости от указанной хеш-функции и указанного количества партиций. Необходимо указать подходящие параметры (см. Опции ADB Spark 3 Connector). У этого режима партиционирования есть ограничения: только выражение, возвращающее целое число, может использоваться в качестве хеш-функции.

Некоторые задачи обработки данных распределяются между всеми executor-нодами Spark 3 вместе с соответствующими партициями, которые были созданы ранее. Обмен данными выполняется в параллельном режиме с использованием внешних writable-таблиц для каждой партиции.



Алгоритм чтения данных

Запись данных

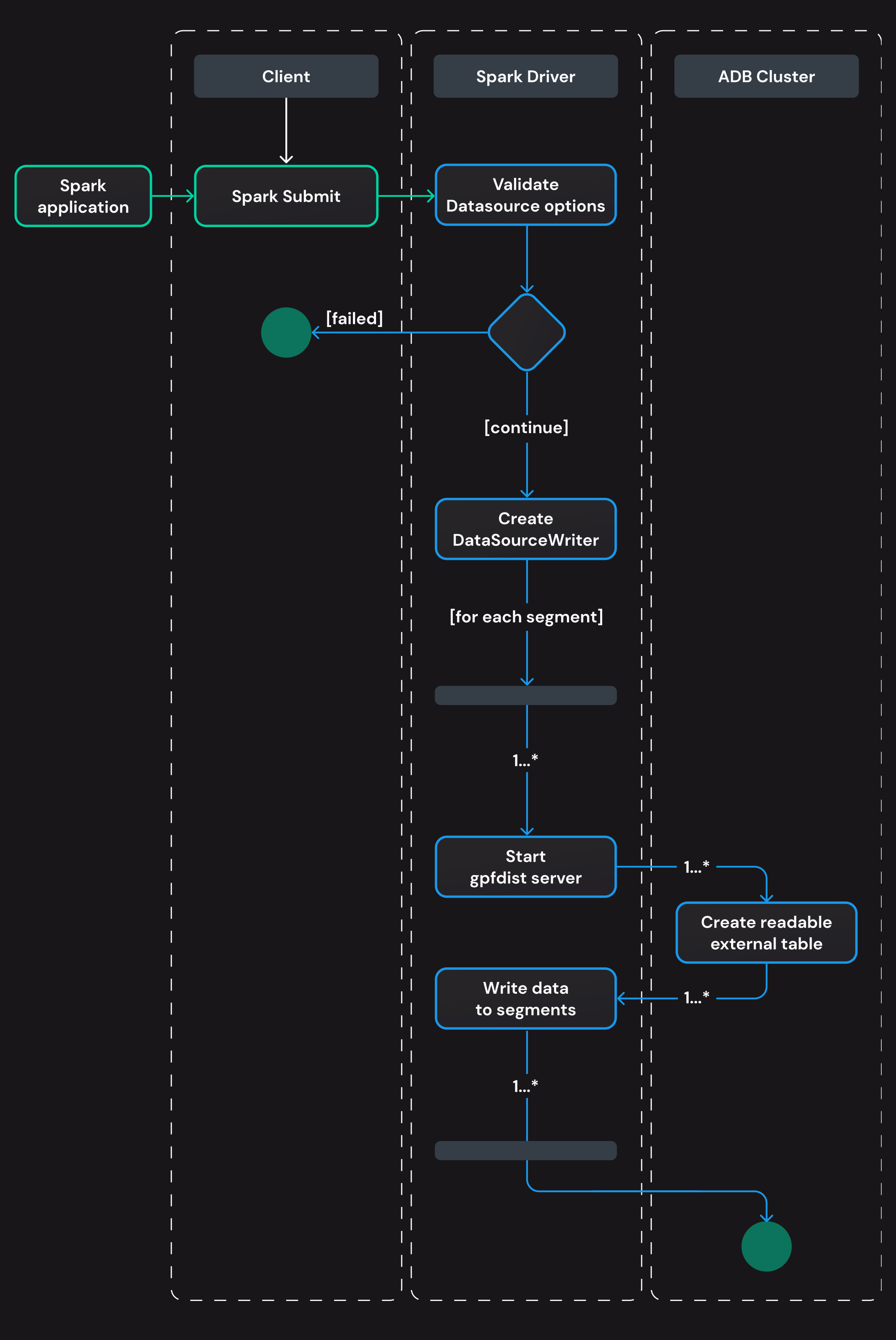
ADB Spark 3 Connector инициализирует драйвер для загрузки таблицы из Spark 3 в ADB. После этого драйвер устанавливает соединение с ADB master через JDBC и готовит загрузку ADB. Метод загрузки зависит от режима записи.

Доступны следующие режимы записи:

- `overwrite`. В зависимости от `spark.db.table.truncate` целевая таблица либо удаляется полностью, либо все данные очищаются (cleaned up) перед процессом записи.
- `append`. Дополнительные данные записываются в целевую таблицу.
- `errorIfExists`. Если целевая таблица существует, появляется ошибка.

Также доступно несколько опций для `spark.db.create.table`. Это позволяет определять дополнительные настройки, когда вы создаёте целевую таблицу (см. Опции ADB Spark 3 Connector).

Задача загрузки данных создаётся для каждой партиции Spark 3. Обмен данными происходит параллельно через внешние readable-таблицы для каждого сегмента.



Алгоритм записи данных

Поддерживаемые типы данных

Соответствия типов данных при их перемещениях представлены в таблицах ниже.

Перемещение из ADB в Spark 3

ADB	Spark 3
bigint	LongType
bigSerial	LongType
bit	StringType
bytea	BinaryType
boolean	BooleanType
char	StringType
date	DateType
decimal	DecimalType
float4	FloatType
float8	DoubleType
int	IntegerType
interval	CalendarIntervalType
serial	IntegerType
smallInt	ShortType
text	StringType
time	TimeStampType
timestamp	TimeStampType
timestamptz	TimeStampType
timetz	TimeStampType
varchar	StringType

Перемещение из Spark 3 в ADB

Spark 3	ADB
BinaryType	byte
BooleanType	boolean
CalendarIntervalType	interval
DateType	date
DecimalType	numeric
DoubleType	float8
FloatType	float4
IntegerType	int
LongType	bigint
ShortType	smallint
StringType	text
TimeStampType	timestamp

Поддерживаемые драйверы JDBC

Для взаимодействия ADB Spark 3 Connector и ADB используется JDBC-соединение. Драйвер PostgreSQL JDBC driver поставляется вместе с коннектором.

Строка подключения JDBC для стандартного драйвера имеет следующую структуру:

```
jdbc:postgresql://<master_ip>[:<port>]/<database_name>
```

где:

- `<master_ip>` — IP-адрес master-хоста ADB.
- `<port>` — номер порта для подключения (`5432` по умолчанию).
- `<database_name>` — название БД в ADB для подключения.

Дополнительно ADB Spark 3 Connector поддерживает использование сторонних драйверов JDBC. Чтобы использовать сторонний драйвер, выполните следующее:

1. Подготовьте строку для подключения ODBC к master-хосту ADB.
2. Загрузите JAR-файл со сторонним драйвером JDBC одним из следующих способов:
   - Используйте переменную окружения `--jars` при вызове spark3-shell или spark3-submit.
   - Выполните билд файла *uber-jar* со всеми зависимостями.
   - Установите JAR-файл со сторонним драйвером на всех нодах Spark 3.
3. Укажите полный путь к Java-классу драйвера, используя опцию коннектора `spark.adb.driver`.

Использование коннектора

Для передачи данных между Spark 3 и ADB с помощью ADB Spark 3 Connector можно использовать стандартные утилиты spark3-shell и spark3-submit. Примеры использования spark3-shell приведены в статье Примеры использования ADB Spark 3 Connector.

При разработке standalone-приложения Spark необходимо предварительно указать JAR-файл коннектора (*/usr/lib/spark3/jars/adb-spark-connector-assembly-release-<connector_version>-spark-<spark_version>.jar*) наряду с другими зависимостями приложения в файле *uber-jar*. Дополнительную информацию можно получить в статье Submitting Applications документации Spark.

Требования к установке ADB Spark 3 Connector

Михаил Серов

Содержание

Назад к содержанию

Чек-лист

Память

Сетевое соединение

Чек-лист

Перед установкой и использованием ADB Spark 3 Connector убедитесь в том, что выполнены следующие требования:

- Есть доступ к кластеру ADB.
- Есть доступ к кластеру Spark 3.
- Есть соединение между нодой ADB master и драйвером Spark 3.
- Есть соединение между нодой ADB master и каждой executor-нодой Spark 3.
- Есть соединение между нодой каждого сегмента ADB и каждой executor-нодой Spark 3.

Необходимо предоставить особые права пользователю, от имени которого будет осуществляться доступ из Spark 3 к таблицам ADB. Эти права включают создание внешних таблиц. Вы можете использовать следующий образец:

```
ALTER ROLE <role_name>
WITH    CREATEEXTTABLE(protocol='gpfdist',type='readable')
        CREATEEXTTABLE(protocol='gpfdist',type='writable');
```

где `<role_name>` — имя пользователя в ADB.

Также потребуется настроить доступ выбранного пользователя к таблицам ADB с хостов Spark 3 в файле *pg_hba.conf* (см. Примеры использования ADB Spark 3 Connector).

Память

Spark 3 работает нормально при любом количестве памяти между восемью и несколькими сотнями гигабайт на рабочую машину. Для Spark 3 мы рекомендуем аллоцировать, самое большее, 75% памяти — оставьте остальное для операционной системы и кеша буфера.

Количество требуемой памяти зависит от приложения. Чтобы определить количество памяти, необходимое приложению для определённого набора данных, загрузите часть вашего набора данных в Spark 3 RDD, затем перейдите на вкладку ***Storage*** в UI для мониторинга Spark 3 (*http://<driver-node>:4040*) и посмотрите количество памяти для этой части данных. На использование памяти влияют уровень хранения (storage level) и формат сериализации. В руководстве по настройке вы можете найти советы о том, как снизить использование памяти.

ПРИМЕЧАНИЕ

Виртуальная машина Java не всегда хорошо работает, если объём RAM превышает 200 ГБ. Если вы приобрели машины с большим объёмом RAM, вы можете запустить несколько виртуальных worker-машин Java на каждой ноде. Для standalone-режима Spark 3 вы можете установить число worker-машин на ноде с помощью переменной `SPARK_WORKER_INSTANCES` в скрипте *conf/spark-env.sh*. Вы также можете установить число ядер на worker-машине с помощью переменной `SPARK_WORKER_CORES`.

Сетевое соединение

ADB Spark 3 Connector использует протокол `gpfdist` для передачи данных между сегментами ADB и execution-нодами Spark 3. По умолчанию, коннектор запускает инстанс сервера `gpfdist` в диапазоне портов `[8080 .. 65536]`.

ВАЖНО

Удостоверьтесь, что в вышеуказанном диапазоне есть открытые порты. Порты должны быть доступны из ADB master.

Примеры использования ADB Spark 3 Connector

Дарья Барышева

Содержание

Назад к содержанию

- Предварительные требования
- Примеры использования: Scala
  - Чтение данных из ADB в Spark
  - Выполнение произвольного запроса SQL в ADB
  - Запись данных из Spark в ADB
- Примеры использования: PySpark
  - Чтение данных из ADB в Spark
  - Запись данных из Spark в ADB

Предварительные требования

В статье показаны примеры передачи данных между ADB и Spark 3 с использованием ADB Spark 3 Connector. При выполнении примеров соблюдены следующие условия:

- Кластер ADB развернут согласно руководству Online-установка.
- Кластер ADH развернут согласно руководству Online-установка. Минимальная версия ADH — 3.1.2.1.b1.
- Сервис Spark 3 установлен в кластере ADH.
- IP-адрес мастер-хоста ADB — `10.92.40.38`. Для входящих подключений PostgreSQL по умолчанию используется порт с номером `5432`.
- База данных с именем `adb` существует в кластере ADB.
- Пользователь с именем `adb_to_spark`, привилегиями `CREATEEXTTABLE` и паролем `123` создан в БД кластера ADB. Для добавления пользователя можно использовать следующий запрос:

```
CREATE ROLE adb_to_spark
WITH  CREATEEXTTABLE(protocol='gpfdist',type='readable')
      CREATEEXTTABLE(protocol='gpfdist',type='writable')
      LOGIN
      PASSWORD '123';
```

- Для пользователя `adb_to_spark` настроен доступ к базе данных `adb` с хостов кластера ADH. Для этой цели отредактирован файл *pg_hba.conf* следующим образом:

```
host    all  adb_to_spark       0.0.0.0/0     md5
```

Обратите внимание, что приведенная конфигурация предназначена для использования исключительно в тестовых целях. Вместо `0.0.0.0/0` можно перечислить IP-адреса хостов ADH-кластера, на которых установлен сервис Spark 3 (с подсетью).

> РЕКОМЕНДАЦИЯ
>
> Редактирование файла *pg_hba.conf* возможно в web-интерфейсе ADCM. Для этого заполните параметр ***Custom pg_hba section*** на вкладке ***Configuration*** сервиса ADB в кластере ADB. Чтобы применить изменения, нажмите ***Save*** и запустите сервисное действие ***Reconfigure***.

- Таблица `author` существует в базе данных `adb`. Для создания и заполнения таблицы предназначены следующие запросы:

```
CREATE TABLE author(id INT NOT NULL, name TEXT NOT NULL)
WITH (appendoptimized=true)
DISTRIBUTED BY(id);
```

```
INSERT INTO author(id, name) VALUES
(1,'Virginia Woolf'),
(2,'J.R.R. Tolkien'),
(3,'Harper Lee'),
(4,'J.D. Salinger'),
(5,'George Orwell'),
(6,'John Steinbeck'),
(7,'Margaret Mitchell'),
(8,'Alan Moore'),
(9,'Jack Kerouac'),
(10,'Ernest Hemingway');
```

- Пользователь `adb_to_spark` имеет необходимые права на доступ к таблице `author`:

```
GRANT SELECT, INSERT ON public.author TO adb_to_spark;
```

Для взаимодействия ADB Spark 3 Connector и ADB используется JDBC-соединение. Драйвер PostgreSQL JDBC driver поставляется вместе с коннектором.

Примеры использования: Scala

Примеры кода на Scala, описанные далее, предназначены для запуска в интерактивной оболочке для Scala — spark3-shell.

Чтение данных из ADB в Spark

1. Откройте spark3-shell на хосте, где установлен Spark 3:

```
$ sudo -u hdfs spark3-shell
```

2. Укажите настройки коннектора для чтения данных из предварительно созданной и заполненной таблицы ADB `author`:

```
val options = Map(
   "spark.adb.url" -> "jdbc:postgresql://10.92.40.38:5432/adb",
   "spark.adb.user" -> "adb_to_spark",
   "spark.adb.password" -> "123",
   "spark.adb.dbschema" -> "public",
   "spark.adb.dbtable" -> "author"
   )
```

Результат:

```
val options: scala.collection.immutable.Map[String,String] = HashMap(spark.adb.password -> 123, spark.adb.dbtable -> author,
spark.adb.user -> adb_to_spark, spark.adb.url -> jdbc:postgresql://10.92.40.38:5432/adb, spark.adb.dbschema -> public)
```

3. Создайте DataFrame:

```
val adb_author  = spark.read.format("adb").options(options).load()
```

Результат:

```
val adb_author: org.apache.spark.sql.DataFrame = [id: int, name: string]
```

4. Проверьте содержимое DataFrame:

```
adb_author.show()
```

Результат:

```
+---+-----------------+
| id|             name|
+---+-----------------+
|  4|    J.D. Salinger|
|  8|       Alan Moore|
|  5|    George Orwell|
| 10| Ernest Hemingway|
|  2|   J.R.R. Tolkien|
|  1|   Virginia Woolf|
|  7|Margaret Mitchell|
|  6|   John Steinbeck|
|  3|       Harper Lee|
|  9|     Jack Kerouac|
+---+-----------------+
```

5. Получите схему выбранной таблицы:

```
adb_author.printSchema()
```

Результат:

```
root
 |-- id: integer (nullable = false)
 |-- name: string (nullable = false)
```

6. Вычислите количество строк в таблице:

```
adb_author.count()
```

Результат:

```
val res9: Long = 10
```

Выполнение произвольного запроса SQL в ADB

Для отправки произвольного SQL-запроса из Spark в ADB выполните следующие шаги:

1. Установите дополнительные настройки Spark-сессии и включите вспомогательные функции:

```
spark.stop
val spark1 = org.apache.spark.sql.SparkSession.
   builder().
   master("local[*]").
   appName("spark_example").
   config("spark.adb.url","jdbc:postgresql://10.92.40.38:5432/adb").
   config("spark.adb.driver","org.postgresql.Driver").
   config("spark.adb.user","adb_to_spark").
   config("spark.adb.password","123").
   getOrCreate()
import io.arenadata.spark.adb.implicits._
```

Результат:

```
val spark1: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@20e87aa3
```

2. Создайте произвольную таблицу `test_spark` в ADB и заполните ее тестовыми данными:

```
spark1.executeAdbQueryOnMaster("CREATE TABLE public.test_spark(id INT);")
spark1.executeAdbQueryOnMaster("INSERT INTO public.test_spark VALUES(1);")
spark1.executeAdbQueryOnMaster("INSERT INTO public.test_spark VALUES(2);")
spark1.executeAdbQueryOnMaster("INSERT INTO public.test_spark VALUES(3);")
```

3. Создайте DataFrame на основе новой таблицы ADB `test_spark`:

```
val res = spark1.executeAdbSelectQueryOnMaster("SELECT * FROM public.test_spark;")
```

Результат:

```
val res: org.apache.spark.sql.DataFrame = [id: int]
```

4. Проверьте содержимое DataFrame:

```
res.show()
```

Результат:

```
+---+
| id|
+---+
|  2|
|  3|
|  1|
+---+
```

Запись данных из Spark в ADB

1. Установите настройки коннектора для записи данных в ADB:

```
val options_write = Map(
   "spark.adb.url" -> "jdbc:postgresql://10.92.40.38:5432/adb",
   "spark.adb.user" -> "adb_to_spark",
   "spark.adb.password" -> "123",
   "spark.adb.dbschema" -> "public",
   "spark.adb.dbtable" -> "test_spark_write")
```

Результат:

```
val options_write: scala.collection.immutable.Map[String,String] = HashMap(spark.adb.password -> 123, spark.adb.dbtable ->
test_spark_write, spark.adb.user -> adb_to_spark, spark.adb.url -> jdbc:postgresql://10.92.40.38:5432/adb, spark.adb.dbschema ->
public)
```

2. Выполните следующую команду для создания таблицы `test_spark_write` в ADB и загрузки в нее данных из созданного в предыдущем примере DataFrame `res`:

```
res.write.format("adb").options(options_write).mode(org.apache.spark.sql.SaveMode.Overwrite).save()
```

3. Проверьте, что данные успешно загружены в ADB (например, используя psql):

   - Вывод структуры таблицы:

```
\d+ public.test_spark_write
```

```
                   Table "public.test_spark_write"
 Column |  Type   | Modifiers | Storage | Stats target | Description
--------+---------+-----------+---------+--------------+-------------
 id     | integer |           | plain   |              |
Distributed by: (id)
```

   - Проверка содержимого таблицы:

```
SELECT * FROM public.test_spark_write;
```

```
 id
----
  2
  1
  3
(3 rows)
```

Примеры использования: PySpark

Примеры кода на Python, приведенные ниже, предназначены для запуска в оболочке PySpark shell, которая доступна на хостах с установленным компонентом Spark3. Для корректной работы примеров из статьи оболочка PySpark должна запускаться пользователем `hdfs`/`spark`, чтобы обеспечить достаточные права для взаимодействия с удаленным ADB-кластером. Ниже показаны команды для запуска оболочки PySpark соответствующим образом.

1. Создайте новый virtual environment, используя Python из */opt/pyspark3-python/*:

```
$ cd ~
$ mkdir pyspark-demo
$ /opt/pyspark3-python/bin/python3 -m venv pyspark-demo/venv
```

2. Активируйте virtual environment:

```
$ source pyspark-demo/venv/bin/activate
```

3. Запустите оболочку PySpark в активированной virtual environment:

```
$ sudo -u hdfs pyspark3
```

Пример вывода показан ниже:

```
(venv) [admin@ka-adh-1 ~]$ sudo -u hdfs pyspark3
Python 3.10.4 (main, Jun  1 2024, 01:11:04) [GCC 4.8.5 20150623 (Red Hat 4.8.5-44)] on linux
Type "help", "copyright", "credits" or "license" for more information.
Picked up _JAVA_OPTIONS: -Djava.io.tmpdir=/var/tmp
Picked up _JAVA_OPTIONS: -Djava.io.tmpdir=/var/tmp
...
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.4.3
      /_/

Using Python version 3.10.4 (main, Jun  1 2024 01:11:04)
Spark context Web UI available at http://ka-adh-1.ru-central1.internal:4142
Spark context available as 'sc' (master = yarn, app id = application_1724163954275_0023).
SparkSession available as 'spark'.
```

Чтение данных из ADB в Spark

1. В оболочке PySpark укажите параметры коннектора для чтения данных из предварительно созданной и заполненной таблицы ADB `author`:

```
opts_read = {
   "spark.adb.url": "jdbc:postgresql://10.92.40.38/adb",
   "spark.adb.user": "adb_to_spark",
   "spark.adb.password": "123",
   "spark.adb.dbschema": "public",
   "spark.adb.dbtable": "author"
}
```

2. Создайте DataFrame:

```
adb_author = spark.read.format("adb") \
            .options(**opts_read) \
            .load()
```

Результат:

```
24/08/28 19:58:59 INFO HikariDataSource: HikariPool-1 - Starting...
24/08/28 19:58:59 INFO HikariDataSource: HikariPool-1 - Start completed.
```

3. Проверьте содержимое DataFrame:

```
adb_author.show()
```

Результат:

```
+---+-----------------+
| id|             name|
+---+-----------------+
|  4|    J.D. Salinger|
|  8|       Alan Moore|
|  5|    George Orwell|
| 10| Ernest Hemingway|
|  2|   J.R.R. Tolkien|
|  1|   Virginia Woolf|
|  7|Margaret Mitchell|
|  6|   John Steinbeck|
|  3|       Harper Lee|
|  9|     Jack Kerouac|
+---+-----------------+
```

4. Получите схему выбранной таблицы:

```
adb_author.printSchema()
```

Результат:

```
root
 |-- id: integer (nullable = false)
 |-- name: string (nullable = false)
```

5. Вычислите количество строк в таблице:

```
adb_author.count()
```

Результат:

```
10
```

Запись данных из Spark в ADB

1. Установите настройки коннектора для записи данных в ADB-таблицу `test_pyspark_write`:

```
opts_write = {
   "spark.adb.url": "jdbc:postgresql://10.92.40.38:5432/adb",
   "spark.adb.user": "adb_to_spark",
   "spark.adb.password": "123",
   "spark.adb.dbschema": "public",
   "spark.adb.dbtable": "test_pyspark_write"
}
```

2. Создайте новый DataFrame и запишите его содержимое в тестовую таблицу ADB:

```
data = [
    (1, "foo"),
    (2, "bar"),
    (3, "buzz")
]
columns = ["id", "value"]
test_df = spark.createDataFrame(data=data, schema=columns)

test_df.write.format("adb") \
    .options(**opts_write) \
    .mode("overwrite") \
    .save()
```

Вывод:

```
24/08/28 17:34:29 INFO HikariDataSource: HikariPool-3 - Starting...
24/08/28 17:34:29 INFO HikariDataSource: HikariPool-3 - Start completed.
...
24/08/28 17:34:31 INFO OverwriteByExpressionExec: Data source write support io.arenadata.spark.adb.spark3.writer.
AdbBatchWrite@26b15eff is committing.
24/08/28 17:34:31 INFO OverwriteByExpressionExec: Data source write support io.arenadata.spark.adb.spark3.writer.
AdbBatchWrite@26b15eff committed.
```

3. Проверьте, что данные успешно загружены в ADB (например, используя psql):

   - Вывод структуры таблицы:

```
\d+ public.test_pyspark_write
```

```
                 Table "public.test_pyspark_write"
 Column |  Type  | Modifiers | Storage  | Stats target | Description
--------+--------+-----------+----------+--------------+-------------
 id     | bigint |           | plain    |              |
 value  | text   |           | extended |              |
Distributed by: (id)
```

   - Проверка содержимого таблицы:

```
SELECT * FROM public.test_pyspark_write;
```

```
 id | value
----+-------
  3 | buzz
  1 | foo
  2 | bar
(3 rows)
```

Опции ADB Spark 3 Connector

Михаил Серов

Назад к содержанию

Параметры чтения/записи, а также параметры, которые необходимы для подключения к базе данных ADB, задаются с помощью пар ключ/значение.

Для коннектора доступны следующие опции.

Опции ADB Spark 3 Connector

Ключ	Описание	Тип	Необходимость	Значение по умолчанию
spark.adb.url	Строка подключения к базе данных	Чтение/запись	Да	—
spark.adb.dbschema	Название схемы базы данных, которой принадлежит таблица	Чтение/запись	Да	public
spark.adb.dbtable	Название таблицы из базы данных	Чтение/запись	Да	—
spark.adb.driver	Полный путь к драйверу JDBC при использовании кастомного драйвера	Чтение/запись	Нет	org.postgresql.Driver
spark.adb.user	Пользователь/роль ADB	Чтение/запись	Да	—
spark.adb.password	Пароль пользователя в ADB	Чтение/запись	Нет	—
spark.adb.server.usehostname	Позволяет использовать название executor-ноды Spark 3 в качестве адреса сервера `gpfdist`	Чтение/запись	Нет	false
spark.adb.server.env.name	Название переменной окружения, значение которой определяет название executor-ноды Spark 3 или IP-адрес, на котором запущен процесс `gpfdist`	Чтение/запись	Нет	—
spark.adb.server.port	Номер или диапазон порта, на котором запущен процесс сервера `gpfdist` executor-ноды Spark 3	Чтение/запись	Нет	—
spark.adb.server.nic	Имя сетевого интерфейса, который необходимо использовать для разрешения FQDN `gpfdist`-сервера. Если параметр не указан, коннектор использует результат команды `hostname -f` для разрешения FQDN	Чтение/запись	Нет	—
spark.adb.pool.maxsize	Максимальное количество соединений в пуле соединений	Чтение/запись	Нет	4
spark.adb.pool.timeoutms	Время в миллисекундах, после которого неактивное (inactive) соединение считается бездействующим (idle)	Чтение/запись	Нет	10000
spark.adb.pool.minidle	Минимальное количество бездействующих соединений, поддерживающееся в пуле соединений	Чтение/запись	Нет	0
spark.adb.debugmode	Включает режим логирования событий в таблице ADB `adb_spark_debug_query_log`	Чтение/запись	Нет	false
spark.adb.partition.column	Название столбца таблицы, использующееся для партиционирования в Spark 3. Этот столбец должен быть типа данных `integer` или `date/time`	Чтение	Нет	—
spark.adb.partition.count	Количество партиций в Spark 3. Может быть задано либо независимо, либо совместно с `spark.add.partition.column` или `spark.adb.partition.hash`	Чтение	Нет	—
spark.adb.partition.hash	Выражение, использующееся в качестве ключа партиционирования при чтении данных в Spark 3. Задаётся совместно с `spark.adb.partition.count`. Это выражение должно возвращать значение типа данных `integer`	Чтение	Нет	—
spark.adb.table.truncate	Используется при записи в режиме `Overwrite`. Выполняет с таблицей операцию truncate в случае значения `true`, иначе — выполняет операцию drop	Запись	Нет	false
spark.adb.create.table.with	Используется при записи в режимах `Overwrite` и `errorIfExists`. Хранит параметры при создании таблицы с выражением `WITH`	Запись	Нет	—
spark.adb.create.table.distributedby	Используется при записи в режимах `Overwrite` и `errorIfExists`. Работает в качестве ключа дистрибуции при создании целевой таблицы с использованием выражения `DISTRIBUTED BY`	Запись	Нет	RANDOMLY
spark.adb.read.mode	Устанавливает режим чтения данных из ADB. Допустимые значения: ▪ `GPFDIST` — Spark получает данные из ADB с помощью утилиты `gpfdist`. ▪ `PARALLEL_CURSOR` — Spark считывает данные непосредственно из сегментов ADB, используя специальную реализацию ADB-курсора `gp_parallel_retrieve_cursor`. Использование этого курсора эффективно при параллельном чтении больших наборов данных, поскольку он предполагает меньшее количество преобразований.	Чтение	Нет	GPFDIST

ADQM Spark 3 Connector

Елена Дворядкина

Назад к содержанию

ADQM Spark 3 Connector — это многофункциональный коннектор с поддержкой параллельных операций чтения/записи между Apache Spark 3 и Arenadata QuickMarts (ADQM).

Обладает большой гибкостью в настройке и множеством функций, таких как:

- высокая скорость передачи данных;
- автоматическое формирование схемы данных;
- гибкое партиционирование;
- поддержка push-down операторов;
- поддержка batch-операций.

ВАЖНО

Начиная с версии ADH 3.2.4.2.b1 коннектор для Spark 3 поставляется вместе с бандлом ADH. Для использования коннектора необходимо установить сервис Spark 3. Порядок работы с коннектором для Spark 2 описан в отдельном руководстве ADQM Spark Connector.

Обзор ADQM Spark 3 Connector

Елена Дворядкина

Содержание

Назад к содержанию

Архитектура

Чтение данных

Запись данных

Поддерживаемые движки таблиц

Поддерживаемые типы данных

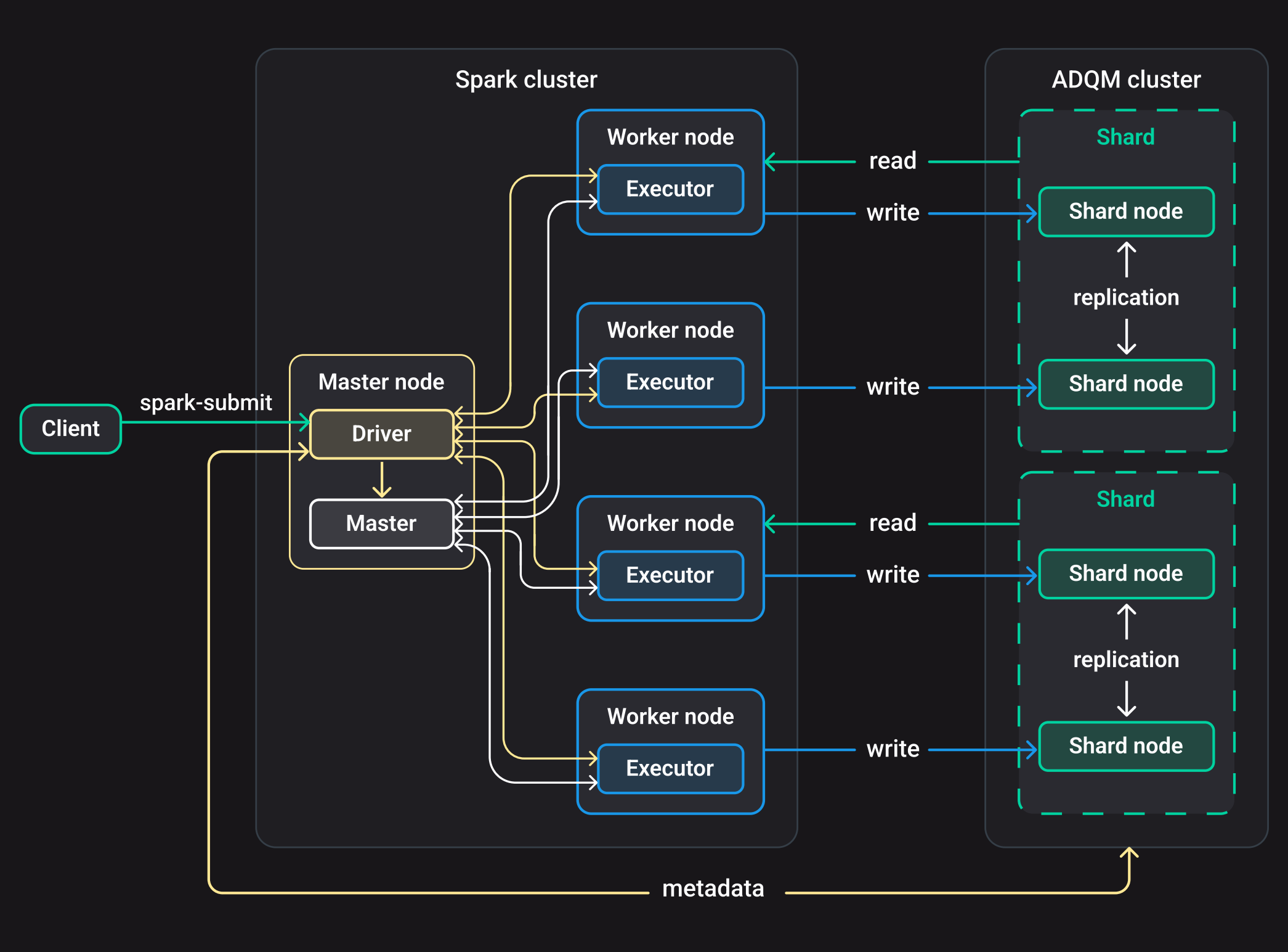
Arenadata QuickMarts (ADQM) — столбцовая система управления базами данных для онлайн-обработки запросов, которая основана на открытой (open-source) СУБД ClickHouse.

Apache Spark 3 — фреймворк с открытым исходным кодом для реализации распределенной обработки неструктурированных и слабоструктурированных данных, входящий в экосистему проектов Hadoop.

ADQM Spark 3 Connector предоставляет возможность высокоскоростного параллельного обмена данными между Apache Spark 3 и Arenadata QuickMarts.

Архитектура

Каждое приложение Spark 3 состоит из управляющего процесса — драйвера (driver) — и набора распределенных worker-процессов — исполнителей (executors). Схема взаимодействия представлена ниже.



Spark cluster
ADQM cluster
Client
spark-submit
Master node
Driver
Master
Worker node
Executor
Worker node
Executor
Worker node
Executor
Worker node
Executor
read
write
write
read
write
write
Shard
Shard node
replication
Shard node
Shard
Shard node
replication
Shard node
metadata

Архитектура ADQM Spark 3 Connector

Чтение данных

Для загрузки таблицы ADQM в Spark 3 на первичном этапе коннектор инициализирует драйвер через строку подключения, которая задана опцией `spark.adqm.url`. При этом если указана опция `spark.adqm.cluster.name`, драйвер, подключаясь к `spark.adqm.url` с помощью JDBC, также считывает метаинформацию о всех шардах и репликах указанного кластера ADQM, что позволяет в дальнейшем эффективно в параллель считывать данные с таблиц шардов по имеющимся исполнителям Spark 3.

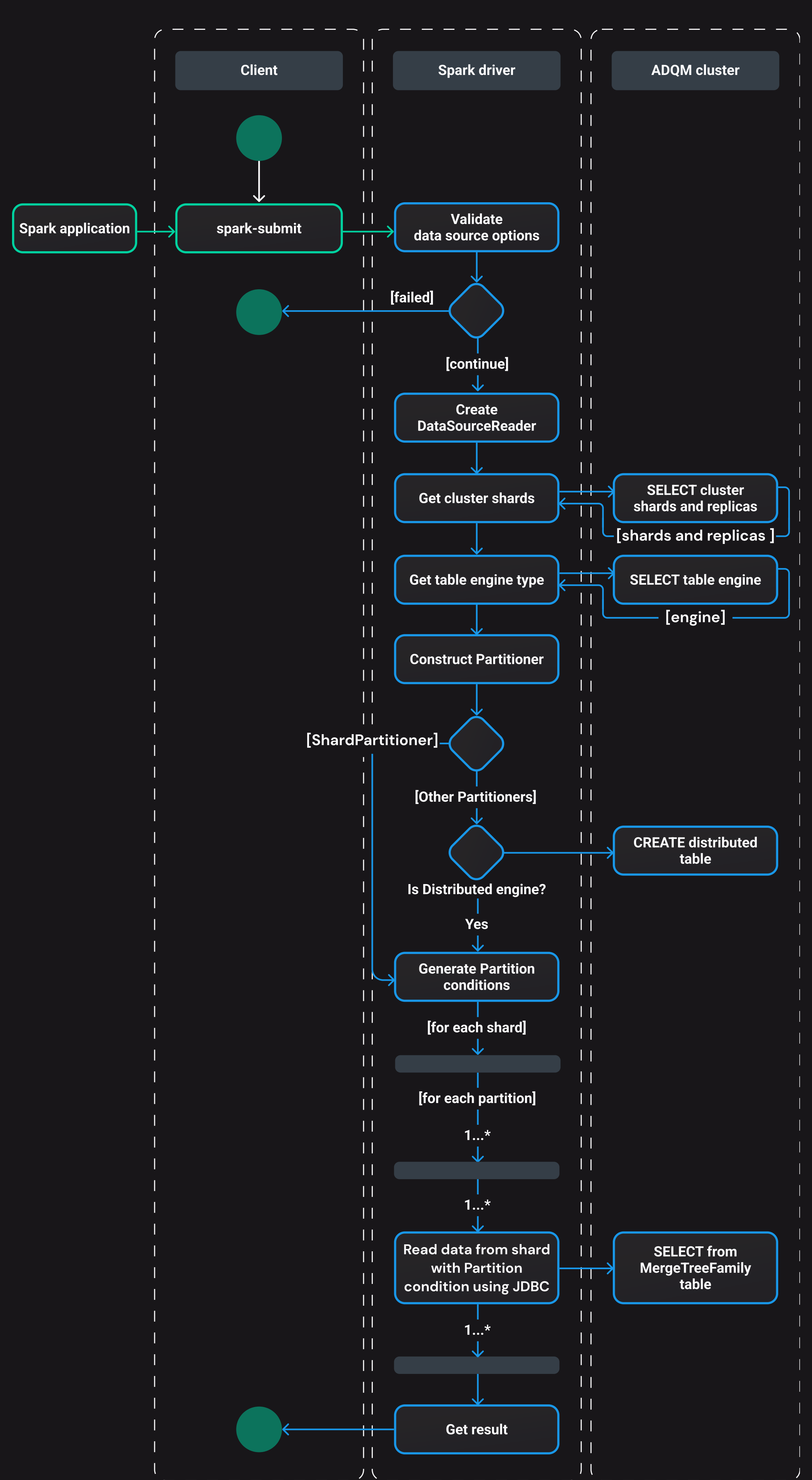
Данные в ADQM хранятся на шардах. Приложение Spark 3, используя коннектор, может получать данные с каждого шарда в одну или несколько партиций Spark 3. В случае чтения из Distributed-таблиц, чтение происходит в параллель на каждом из шардов из соответствующей MergeTree-таблицы.

На текущий момент существует три режима партиционирования:

> **ПРИМЕЧАНИЕ**
>
> Информация справедлива для случаев, когда опция `spark.adqm.cluster.name` указана и не указана.

- **По shard_id**. Данные из таблиц ADQM считываются и распределяются по партициям Spark 3 в соответствии с номером шарда. Количество партиций Spark 3 соответствует числу активных шардов ADQM. Этот режим партиционирования не требует дополнительных параметров и используется по умолчанию.
- **По указанному столбцу и указанному количеству партиций**. Данные разбиваются по партициям Spark 3 согласно диапазонам значений указанного столбца на указанное число партиций. Количество партиций Spark 3 соответствует количеству диапазонов. Необходимо указать параметры `spark.adqm.partition.column` и `spark.adqm.partition.count` (см. Опции ADQM Spark 3 Connector). Данный режим имеет ограничения: в качестве столбца могут быть использованы только поля таблицы типа integer или date/time.
- **По указанному столбцу**. Данные разбиваются по партициям Spark 3 в соответствии с уникальными значениями указанного столбца. Количество партиций Spark 3 соответствует количеству уникальных значений в указанном столбце. Необходимо установить параметр `spark.adqm.partition.column` (см. Опции ADQM Spark 3 Connector). Рекомендуется использовать данный режим при небольшом, ограниченном наборе значений.

При такой архитектуре на каждый исполнитель Spark 3 назначается задача по обработке данных и соответствующая ему партиция из ранее созданных. Обмен данными выполняется в параллельном режиме для каждой партиции с каждым шардом (если указана опция `spark.adqm.cluster.name` — с каждым шардом данного кластера, в ином случае — с шардами, указанными в опции `spark.adqm.url`).



Client
Spark driver
ADQM cluster
Spark application
spark-submit
Validate data source options
[failed]
[continue]
Create DataSourceReader
Get cluster shards
SELECT cluster shards and replicas
[shards and replicas]
Get table engine type
SELECT table engine
[engine]
Construct Partitioner
[ShardPartitioner]
[Other Partitioners]
CREATE distributed table
Is Distributed engine?
Yes
Generate Partition conditions
[for each shard]
[for each partition]
1...*
1...*
Read data from shard with Partition condition using JDBC
SELECT from MergeTreeFamily table
1...*
Get result

Алгоритм чтения данных

Запись данных

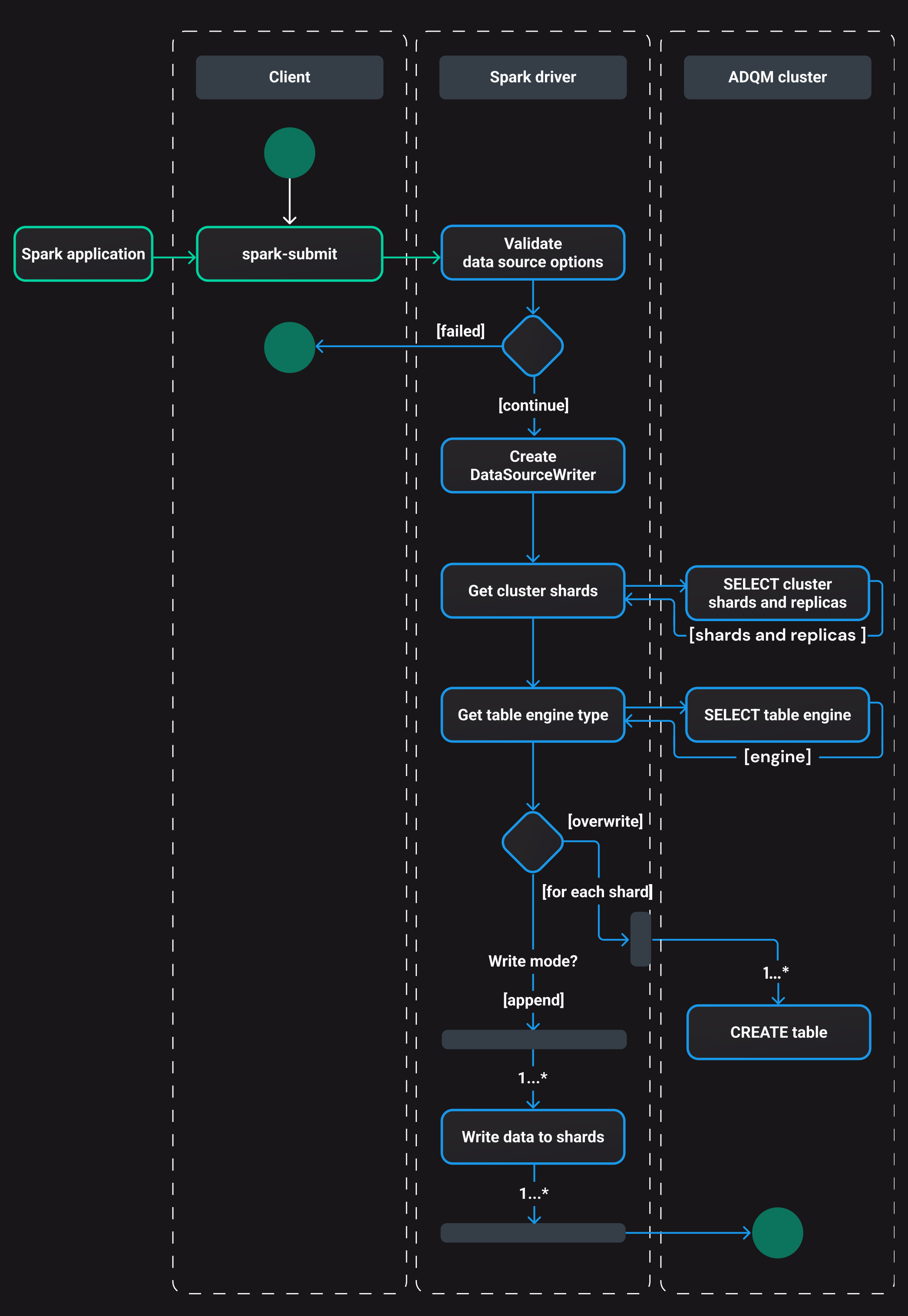
Для загрузки таблицы из Spark 3 в ADQM на первичном этапе коннектор инициализирует драйвер через строку подключения `spark.adqm.url`. При этом если указана опция `spark.adqm.cluster.name`, драйвер, подключаясь к `spark.adqm.url` с помощью JDBC, также считывает метаинформацию о всех шардах и репликах указанного кластера ADQM. В зависимости от режима записи производится подготовка каждого шарда к загрузке данных (создание или очистка таблиц).

На текущий момент поддерживаются следующие режимы записи:

- `append`. Дополнительные данные записываются в целевую таблицу. При указании опции `spark.adqm.cluster.name` осуществляется попытка записи в каждый из шардов кластера, в ином случае — только в те шарды, которые указаны в `spark.adqm.url`. Если на каком-либо из шардов отсутствует таблица, в которую предполагается дозапись, процесс загрузки завершается ошибкой. В этом режиме на всех шардах должна быть предварительно создана целевая таблица.
- `errorIfExists`. Если целевая таблица существует, появляется ошибка. В этом режиме предполагается, что целевой таблицы нет, она создается в процессе загрузки. Необходимо указать информацию о движке создаваемой целевой таблицы с помощью опции `spark.adqm.create.table.engine`. Если указана опция `spark.adqm.cluster.name`, то целевая таблица с указанным движком создается на каждом из шардов кластера, в ином случае — целевая таблица создается на каждом из шардов, заданных в опции `spark.adqm.url`.
- `overwrite`. Данные в таблице перезаписываются. В зависимости от `spark.adqm.table.truncate` целевая таблица либо удаляется полностью, либо в ней все данные очищаются перед процессом записи. В случае полного удаления таблицы процесс записи аналогичен режиму `errorIfExists`, в случае очистки — аналогичен режиму `append`.

Также доступен ряд опций `spark.adqm.create.table`, позволяющих указать дополнительные настройки при создании целевой таблицы (см. Опции ADQM Spark 3 Connector).

Далее для каждой партиции Spark 3 создается задача по загрузке данных. Обмен данных выполняется в параллельном режиме с каждым шардом (если указана опция `spark.adqm.cluster.name` — с каждым из шардов данного кластера, в ином случае — с шардами, указанными в опции `spark.adqm.url`).



Client
Spark driver
ADQM cluster
Spark application
spark-submit
Validate data source options
[failed]
[continue]
Create DataSourceWriter
Get cluster shards
SELECT cluster shards and replicas
[shards and replicas]
Get table engine type
SELECT table engine
[engine]
[overwrite]
[for each shard]
1...*
CREATE table
Write mode?
[append]
1...*
Write data to shards
1...*

Алгоритм записи данных

Поддерживаемые движки таблиц

Движок таблицы (**table engine**) — сущность ADQM, которая определяет, как данные хранятся и каким образом их записывать, откуда их читать и какой к ним имеется доступ, а также задает параметры запросов, индексов и репликации (см. Движки таблиц в документации ADQM).

ADQM Spark 3 Connector поддерживает следующие движки таблиц ADQM:

- **Движки семейства MergeTree**. Предназначены для быстрой записи огромного объема данных по частям относительно заданных правил. Поддерживается основной движок MergeTree, а также его специальные версии:
  - AggregatingMergeTree;
  - CollapsingMergeTree;
  - VersionedCollapsingMergeTree;
  - ReplacingMergeTree;
  - SummingMergeTree;
  - ReplicatedMergeTree;
  - ReplicatedAggregatingMergeTree;
  - ReplicatedCollapsingMergeTree;
  - ReplicatedVersionedCollapsingMergeTree;
  - ReplicatedReplacingMergeTree;
  - ReplicatedSummingMergeTree.
- **Движок Distributed**. Не хранит данные, а позволяет обрабатывать запросы распределенно на нескольких серверах, выполняя параллельное чтение автоматически (подробнее о создании и использовании таблиц типа Distributed в статье Шардирование документации ADQM).

Поддерживаемые типы данных

Соответствия типов данных при перемещениях данных представлены в таблицах ниже.

Перемещение из ADQM в Spark 3

ADQM	Spark 3
Date	DateType
DateTime	TimestampType
DateTime64	TimestampType
Decimal	DecimalType
Decimal128	DecimalType
Decimal256	DecimalType
Decimal32	DecimalType
Decimal64	DecimalType
Float32	FloatType
Float64	DoubleType
IPv4	LongType
Int16	IntegerType
Int32	IntegerType
Int64	LongType
Int8	ShortType
String	StringType
UInt16	IntegerType
UInt32	LongType
UInt64	LongType
UInt8	ShortType
UUID	StringType
float8	DoubleType

Перемещение из Spark 3 в ADQM

Spark 3	ADQM
BinaryType	String
BooleanType	UInt8
ByteType	Int8
DateType	Date
DecimalType	Decimal
DoubleType	Float64
FloatType	Float32
IntegerType	Int32
LongType	Int64
ShortType	Int16
StringType	String
TimestampType	DateTime64

Требования к установке

Елена Дворядкина

Содержание

Назад к содержанию

Чек-лист

Поддерживаемые платформы и версии

Память

Чек-лист

Убедитесь в том, что выполнены следующие требования:

- Есть доступ к кластеру ADQM.
- Есть доступ к кластеру Spark 3.
- Есть соединение между всеми шардами ADQM и драйвером Spark 3.
- Есть соединение между всеми шардами ADQM и каждым узлом-исполнителем Spark 3.

Поддерживаемые платформы и версии

- ADQM начиная с версии 20.8.11.17.
- Spark 3.3.x, Spark 3.4.x.
- Scala 2.13.
- ClickHouse Native JDBC 2.5.4.

Память

Spark 3 работает нормально при любом количестве памяти между восемью и несколькими сотнями гигабайт на рабочую машину. Для Spark 3 рекомендуется аллоцировать, самое большее, 75% памяти — оставьте остальное для операционной системы и кеша буфера.

Количество требуемой памяти зависит от приложения. Чтобы определить количество памяти, необходимое приложению для определенного набора данных, загрузите часть вашего набора данных в Spark 3 RDD, затем перейдите на вкладку ***Storage*** в UI для мониторинга Spark 3 (*http://<driver-node>:4040*) и посмотрите количество памяти для этой части данных. На использование памяти влияют уровень хранения (storage level) и формат сериализации. В руководстве по настройке вы можете найти советы о том, как снизить использование памяти.

ПРИМЕЧАНИЕ

Виртуальная машина Java не всегда хорошо работает, если объем RAM превышает 200 ГБ. Если вы приобрели машины с большим объемом RAM, вы можете запустить несколько виртуальных worker-машин Java на каждой ноде. Для standalone-режима Spark 3 вы можете установить число worker-машин на ноде с помощью переменной `SPARK_WORKER_INSTANCES` в скрипте *conf/spark-env.sh*. Вы также можете установить число ядер на worker-машине с помощью переменной `SPARK_WORKER_CORES`.

Пример использования

Елена Дворядкина

Содержание

Назад к содержанию

Пример загрузки данных из ADQM в Spark 3

Пример выполнения произвольного запроса SQL на узле ADQM

Пример загрузки данных из Spark 3 в ADQM

Прежде чем приступить к использованию ADQM Spark 3 Connector, необходимо уточнить следующую информацию:

- имя хоста ADQM;
- порт для взаимодействия с ADQM;
- реквизиты пользователя для подключения;
- имя базы данных, к которой необходимо подключиться;
- имя таблицы, к которой нужно получить доступ.

В примерах, приведенных в этой статье, используются следующие тестовые данные:

- хост ADQM — `10.92.17.146`;
- порт — `9000`;
- пользователь ADQM — `default` (без пароля);
- база данных — `default`;
- таблица для чтения из ADQM — `users`:

```
┌─user_id─┬─name───┬───role──┐
│       1 │ john   │    admin│
│       2 │ mary   │   author│
│       3 │ andrew │   author│
│       4 │ harry  │ reviewer│
│       5 │ ann    │view only│
└─────────┴────────┴─────────┘
```

Для взаимодействия ADQM Spark 3 Connector и ADQM используется соединение JDBC. Драйвер ClickHouse Native JDBC поставляется вместе с коннектором.

Команды выполняются в интерактивной оболочке для Scala — spark3-shell.

Пример загрузки данных из ADQM в Spark 3

1. Укажите настройки коннектора для чтения данных из таблицы ADQM `default.users`:

```
val options_read = Map(
    "spark.adqm.url" -> "jdbc:clickhouse://10.92.17.146:9000",
    "spark.adqm.user" -> "default",
    "spark.adqm.dbschema" -> "default",
    "spark.adqm.dbtable" -> "users")
```

2. Создайте DataFrame:

```
val adqm_users = spark.read.format("adqm").options(options_read).load()
```

Результат выполнения команды:

```
adqm_users: org.apache.spark.sql.DataFrame = [user_id: int, name: string ... 1 more field]
```

3. Проверьте содержимое DataFrame:

```
adqm_users.show()
```

```
+-------+------+---------+
|user_id|  name|     role|
+-------+------+---------+
|      1|  john|    admin|
|      2|  mary|   author|
|      3|andrew|   author|
|      4| harry| reviewer|
|      5|   ann|view only|
+-------+------+---------+
```

Пример выполнения произвольного запроса SQL на узле ADQM

1. Отредактируйте настройки Spark-сессии и включите вспомогательные функции:

```
spark.stop
val spark1 = org.apache.spark.sql.SparkSession.
    builder().
    master("local[*]").
    appName("spark_example").
    config("spark.adqm.url","jdbc:clickhouse://10.92.17.146:9000").
    config("spark.adqm.user","default").
    getOrCreate()
import io.arenadata.spark.adqm.implicits._
```

2. Создайте произвольную таблицу в ADQM и заполните ее тестовыми данными:

```
spark1.executeAdqmQueryOnShard("create table default.test_spark(id integer) ENGINE  = MergeTree() ORDER BY id;")
spark1.executeAdqmQueryOnShard("insert into default.test_spark values(1);")
spark1.executeAdqmQueryOnShard("insert into default.test_spark values(2);")
spark1.executeAdqmQueryOnShard("insert into default.test_spark values(3);")
```

3. Создайте DataFrame на основе новой таблицы ADQM:

```
val test = spark1.executeAdqmSelectQueryOnShard("select * from default.test_spark;")
```

```
test: org.apache.spark.sql.DataFrame = [id: int]
```

4. Проверьте содержимое DataFrame:

```
test.show()
```

```
+---+
| id|
+---+
|  1|
|  2|
|  3|
+---+
```

5. Проверьте в ADQM (например, используя консольный клиент clickhouse-client), что таблица создалась и заполнилась данными:

```
SELECT * FROM test_spark;
```

```
┌─id─┐
│  1 │
│  2 │
│  3 │
└────┘
```

Пример загрузки данных из Spark 3 в ADQM

1. Укажите настройки коннектора:

```
val options_write = Map(
    "spark.adqm.url" -> "jdbc:clickhouse://10.92.17.146:9000",
    "spark.adqm.user" -> "default",
    "spark.adqm.dbschema" -> "default",
    "spark.adqm.dbtable" -> "test_spark_write",
    "spark.adqm.create.table.engine" -> "MergeTree")
```

2. Выполните следующую команду, чтобы создать в ADQM таблицу `test_spark_write` и записать в нее данные из созданного в предыдущем примере DataFrame:

```
test.write.format("adqm").options(options_write).mode(org.apache.spark.sql.SaveMode.Overwrite).save()
```

3. Проверьте в ADQM, что в новую таблицу добавились данные:

```
SELECT * FROM test_spark_write;
```

```
┌─id─┐
│  1 │
│  2 │
│  3 │
└────┘
```

Опции ADQM Spark 3 Connector

Елена Дворядкина

Назад к содержанию

Необходимые параметры подключения к базе данных ADQM, а также параметры чтения/записи задаются с помощью пар ключ/значение.

Для ADQM Spark 3 Connector доступны следующие опции.

Опции ADQM Spark 3 Connector

Ключ	Описание	Тип операции	Обязательность	Значение по умолчанию
spark.adqm.url	Строка подключения к базе данных с помощью драйвера, использующего протокол TCP	Чтение/запись	Да	—
spark.adqm.dbschema	Имя схемы базы данных, которой принадлежит таблица	Чтение/запись	Да	default
spark.adqm.dbtable	Имя таблицы базы данных	Чтение/запись	Да	—
spark.adqm.user	Пользователь/роль ADQM	Чтение/запись	Да	—
spark.adqm.password	Пароль пользователя в ADQM	Чтение/запись	Нет	—
spark.adqm.cluster.name	Имя кластера ADQM	Чтение/запись	Нет	—
spark.adqm.jdbc.pool.maxsize	Максимальное количество подключений в пуле подключений	Чтение/запись	Нет	4
spark.adqm.jdbc.pool.timeout	Время в миллисекундах, по истечении которого неактивное соединение считается бездействующим	Чтение/запись	Нет	10000
spark.adqm.jdbc.pool.minidle	Минимальное количество бездействующих соединений, поддерживаемое в пуле соединений	Чтение/запись	Нет	0
spark.adqm.partition.column	Имя столбца таблицы, используемого в Spark 3 для разделения по партициям. Столбец должен быть типа integer или date/time	Чтение	Нет	—
spark.adqm.partition.count	Количество партиций в Spark 3. Может быть указано только совместно с `spark.adqm.partition.column`	Чтение	Нет	—
spark.adqm.insert.batchsize	Размер батча (batch) при вставке в целевую таблицу	Запись	Нет	10000
spark.adqm.insert_distributed_sync	Режим вставки в ADQM: `0` — вставка в асинхронном режиме; `1` — вставка в синхронном режиме	Запись	Нет	0
spark.adqm.table.truncate	Используется при записи в режиме `overwrite`. Выполняет операцию truncate table в случае значения `true`, иначе — операцию drop table	Запись	Нет	false

Для режима `overwrite` с установленным в `true` значением опции `spark.adqm.table.truncate` и для режима `errorIfExists` есть дополнительные опции, описанные в таблице ниже. В иных режимах данные опции игнорируются.

Дополнительные опции ADQM Spark 3 Connector

Ключ	Описание	Обязательность
spark.adqm.create.table.engine	Движок целевой таблицы	Да
spark.adqm.create.table.storage_policy	Политика хранения целевой таблицы	Нет
spark.adqm.create.table.distributed.sharding_key	Ключ шардирования для целевой таблицы Distributed	Нет
spark.adqm.create.table.merge_tree.sorting_key	Ключ сортировки для целевой таблицы семейства MergeTree. При указанной опции `spark.adqm.create.table.merge_tree.primary_key` значение `primaryKey`, в ином случае — `tuple()`	Нет
spark.adqm.create.table.merge_tree.sampling_key	Ключ сэмплирования для целевой таблицы семейства MergeTree	Нет
spark.adqm.create.table.merge_tree.primary_key	Первичный ключ для целевой таблицы семейства MergeTree	Нет
spark.adqm.create.table.merge_tree.partition_key	Ключ партиционирования для целевой таблицы семейства MergeTree	Нет
spark.adqm.create.table.summing_merge_tree.columns	Столбцы целевой таблицы (Replicated)SummingMergeTree	Нет
spark.adqm.create.table.replacing_merge_tree.version	Столбец с версией записи данных для целевой таблицы (Replicated)ReplacingMergeTree	Нет
spark.adqm.create.table.collapsing_merge_tree.sign	Признак таблицы (Replicated)CollapsingMergeTree/(Replicated)VersionedCollapsingMergeTree	Нет
spark.adqm.create.table.collapsing_merge_tree.version	Версия целевой таблицы (Replicated)CollapsingMergeTree/(Replicated)VersionedCollapsingMergeTree	Нет

PySpark

Константин Алпашкин

Содержание

Назад к содержанию

Требования к кластеру

Использование PySpark и spark3-submit

Использование PySpark shell

Действия в защищенном кластере

PySpark — это Python-обертка над Spark, которая позволяет вызывать нативные методы Spark из Python-кода. Ядром PySpark является библиотека Py4J, которая позволяет интерпретатору Python получить доступ к объектам Spark в JVM.

В данной статье показаны примеры использования PySpark через `spark3-submit` и PySpark shell, доступные с сервисом Spark3.

Требования к кластеру

Примеры, описанные в данной статье, предназначены для выполнения в кластере со следующими характеристиками:

- Узлы кластера — CentOS 7 машины. Для других OS команды/пути могут отличаться.
- Установлен кластер ADH 3.1.2.1.b1 или более поздней версии. В кластере установлен сервис Spark3.
  В процессе установки ADH-кластера ADCM также устанавливает интерпретатор Python3 в директорию */opt/pyspark3-python/* на тех хостах, где расположен компонент Spark3 Client. Данный интерпретатор Python используется для всех примеров, описанных в разделе, однако возможно использование другого интерпретатора Python версии 3.10+.
- В HDFS создана директория для тестовой загрузки данных. Пользователь, от имени которого запускается PySpark, должен быть назначен владельцем (owner) этой директории.

```
$ sudo -u hdfs hdfs dfs -mkdir /user/admin
$ sudo -u hdfs hdfs dfs -chown admin:admin /user/admin
```

Использование PySpark и spark3-submit

Ниже показан запуск Spark-задачи на Python с использованием `spark3-submit` в режиме кластера.

1. На всех узлах кластера установите переменную окружения `PYSPARK_PYTHON`, которая позволяет Spark вызывать нужный Python-интерпретатор. Для этого в ADCM перейдите на страницу ***Clusters** → **<имя_кластера>** → **Services** → **Spark3** → **Primary configuration*** и в поле ***spark-env.sh*** добавьте строку `export PYSPARK_PYTHON=/opt/pyspark3-python/bin/python3` (поле становится видимым после выбора опции ***Show advanced***). Затем сохраните настройки и перезапустите сервис Spark3.

2. Создайте тестовый файл *test.py*.

```
from pyspark import SparkConf
from pyspark import SparkContext
from pyspark.sql import SparkSession

conf = SparkConf()
conf.setMaster('yarn')
conf.setAppName('spark-yarn')
sparkSession = SparkSession.builder.appName("spark-yarn").config(conf=conf).getOrCreate()

data = [('First', 1), ('Second', 2), ('Third', 3), ('Fourth', 4), ('Fifth', 5)]
df = sparkSession.createDataFrame(data) ①
df.write.csv("hdfs:///user/admin/tmp/test") ②
quit()
```

① Создание DataFrame из коллекции.

② Запись содержимого DataFrame в HDFS в виде CSV-файла.

3. Запустите `spark3-submit`, передавая *test.py* в качестве аргумента. В данном примере `spark3-submit` необходимо запускать от пользователя `admin`.

```
$ /bin/spark3-submit \
    --deploy-mode cluster \ ①
    --master yarn \ ②
    test.py
```

① Запуск Spark в режиме кластера.

② Master URL для запуска Spark в YARN-кластере. Адрес кластера определяется на основе переменных окружения `HADOOP_CONF_DIR` или `YARN_CONF_DIR`.

4. Проверьте запись данных в HDFS:

```
$ sudo -u hdfs hdfs dfs -ls /user/admin/tmp/test/
```

Части (parts) DataFrame сохранены в HDFS:

```
Found 3 items
-rw-r--r--   3 admin admin          0 2023-07-10 12:51 /user/admin/tmp/test/_SUCCESS
-rw-r--r--   3 admin admin         17 2023-07-10 12:51 /user/admin/tmp/test/part-00000-1a966f98-6c1a-467b-9564-dbbd65dd32a2-c000.
csv
-rw-r--r--   3 admin admin         25 2023-07-10 12:51 /user/admin/tmp/test/part-00001-1a966f98-6c1a-467b-9564-dbbd65dd32a2-c000.
csv
```

Использование PySpark shell

PySpark shell позволяет запускать Spark-задачи в интерактивном режиме. Ниже показана установка PySpark shell и запуск задачи Spark в локальном режиме.

1. Создайте новый virtual environment, используя Python из */opt/pyspark3-python/*:

```
$ cd ~
$ mkdir pyspark-demo
$ /opt/pyspark3-python/bin/python3 -m venv pyspark-demo/venv
```

2. Активируйте virtual environment:

```
$ source pyspark-demo/venv/bin/activate
```

1. Запустите PySpark:

```
$ pyspark3
```

Вывод имеет следующий вид:

```
Python 3.10.4 (main, Sep  7 2023, 08:17:33) [GCC 4.8.5 20150623 (Red Hat 4.8.5-44)] on linux
Type "help", "copyright", "credits" or "license" for more information.
Picked up _JAVA_OPTIONS: -Djava.io.tmpdir=/var/tmp
Picked up _JAVA_OPTIONS: -Djava.io.tmpdir=/var/tmp
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/spark3/jars/log4j-slf4j-impl-2.17.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/09/28 08:37:05 WARN HiveConf: HiveConf of name hive.metastore.event.db.notification.api.auth does not exist
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.3.2
      /_/

Using Python version 3.10.4 (main, Sep  7 2023 08:17:33)
Spark context Web UI available at http://ka-adh-1.ru-central1.internal:4040
Spark context available as 'sc' (master = yarn, app id = application_1695647776253_0020).
SparkSession available as 'spark'.
```

2. Выполните следующий код непосредственно в PySpark shell. В данном примере создается Spark DataFrame с последующей записью в HDFS через NameNode.

```
from pyspark import SparkConf
from pyspark import SparkContext
from pyspark.sql import SparkSession

conf = SparkConf()
conf.setMaster('yarn')
conf.setAppName('spark-yarn')
sparkSession = SparkSession.builder.appName("spark-yarn").config(conf=conf).getOrCreate()

data = [('First', 1), ('Second', 2), ('Third', 3), ('Fourth', 4), ('Fifth', 5)]
df = sparkSession.createDataFrame(data)
df.write.csv("hdfs://<active-nn-host>:8020/user/admin/tmp/test-nn") ①
quit()
```

① `<active-nn-host>` должен указывать на активную NameNode. Чтобы определить активную NameNode, используйте команду:

```
$ sudo -u hdfs hdfs haadmin -getAllServiceState
```

Пример вывода:

```
ka-adh-1.ru-central1.internal:8020                 standby
ka-adh-3.ru-central1.internal:8020                 active
```

3. Проверьте успешную запись данных в HDFS:

```
$ sudo -u hdfs hdfs dfs -ls /user/admin/tmp/test-nn/
```

Данные сохранены в HDFS:

```
-rw-r--r--   3 admin hadoop          9 2023-03-10 16:53 /user/admin/tmp/test-nn/part-00001-dedba941-113e-4fd6-871d-e87dd3291e57-
c000.csv
-rw-r--r--   3 admin hadoop          8 2023-03-10 16:53 /user/admin/tmp/test-nn/part-00002-dedba941-113e-4fd6-871d-e87dd3291e57-
c000.csv
-rw-r--r--   3 admin hadoop         17 2023-03-10 16:53 /user/admin/tmp/test-nn/part-00003-dedba941-113e-4fd6-871d-e87dd3291e57-
c000.csv
```

Действия в защищенном кластере

Если в кластере настроена безопасность, а именно:

- установлен Apache Ranger в ADPS;
- керберизованы ADPS и ADH;
- ноды кластера объединены в домен и используют SSSD;
- активированы плагины HDFS/YARN;
- настроены политики Ranger для HDFS/YARN-плагинов.

В таком кластере процедура использования PySpark полностью **идентична** вышеописанной, главными требованиями остаются:

1. Группа пользователя/домен должны иметь доступ к HDFS-директориям, определенным в списке управления доступом Ranger ACL.
2. Перед началом работы необходимо получить тикет Kerberos, используя `kinit` как показано ниже:

```
$ kinit myuser@EXAMPLE_REALM.COM
```

Использование Spark Connect

Константин Алпашкин

Содержание

Назад к содержанию

- Компонент Spark3 Connect
- Использование Spark Connect в приложениях Spark
  - Использование spark3-submit
  - Использование PySpark shell
- Аутентификация

В данной статье показаны примеры удаленного взаимодействия с кластером Spark3 через Spark Connect.

Компонент Spark3 Connect

Чтобы использовать Spark Connect, в вашем ADH-кластере должен быть установлен соответствующий компонент (Spark3 Connect) сервиса Spark3. Этот компонент можно добавить с помощью действий сервиса Spark3 в ADCM. После добавления в кластер компонент готов к работе из коробки и не требует настройки.

Использование Spark Connect в приложениях Spark

Код клиентского приложения Spark, в котором используется Spark Connect, почти не отличается от обычного приложения Spark. Главное отличие — использование **удаленной** Spark-сессии (объект SparkSession), которая отвечает за соединение с удаленным сервером Spark Connect. При использовании удаленной сессии код приложения Spark выполняется в YARN-кластере, где установлен сервер Spark Connect. После создания удаленной `SparkSession` взаимодействие с объектом сессии ведется таким же образом, как если бы это была обычная сессия Spark. Все коммуникации с удаленным сервером Spark Connect выполняются автоматически с помощью клиентской библиотеки Spark Connect.

Основные способы создания удаленной сессии показаны ниже.

Использование spark3-submit

При использовании */bin/spark3-submit* для запуска клиентских Spark-приложений создание удаленной сессии в коде приложения выглядит следующим образом:

```
from pyspark import SparkConf
from pyspark import SparkContext
from pyspark.sql import SparkSession

conf = SparkConf()
conf.setMaster('yarn')
conf.setAppName('spark-yarn')
sparkSession = SparkSession
    .builder
    .appName("spark-yarn")
    .config(conf=conf)
    .remote("sc://<sc_host>:<sc_port>") ①
    .getOrCreate()

data = [('First', 1), ('Second', 2), ('Third', 3)]
df = sparkSession.createDataFrame(data)
df.write.csv("hdfs:///user/admin/tmp/test") ②
quit()
```

① Создание удаленной сессии Spark. `<sc_host>:<sc_port>` — это gRPC-эндпоинт компонента Spark3 Connect. Актуальный адрес эндпоинта доступен в ADCM на странице ***Clusters → <your_cluster_name> → Services → Spark3 → Info***.

② Запись содержимого DataFrame в HDFS. Все вычисления и операции записи выполняются в кластере Spark3, к которому относится сервер Spark Connect.

После запуска такого приложения в выводе содержится строка, сигнализирующая о соединении с удаленным сервером Spark Connect.

```
Client connected to the Spark Connect server at ka-adh-1.ru-central1.internal:15002
```

ПРИМЕЧАНИЕ

В приложении Spark одновременно может существовать только 1 объект `SparkSession`. Вы можете удалить существующую сессию с помощью метода SparkSession.stop() и создать новый объект удаленной сессии как показано в примере выше.

Использование PySpark shell

При использовании PySpark shell (*/usr/bin/pyspark3*) объект `SparkSession` создается автоматически при запуске.

```
...
SparkSession available as 'spark'.
```

Для того чтобы сессия автоматически создавалась удаленной, используйте флаг `--remote` при запуске */usr/bin/pyspark3*. Например:

```
./bin/pyspark --remote "sc://<sc_host>:<sc_port>"
```

Где `<sc_host>:<sc_port>` — это gRPC-эндпоинт компонента Spark3 Connect. Актуальный адрес эндпоинта доступен в ADCM на странице ***Clusters → <your_cluster_name> → Services → Spark3 → Info***.

Аутентификация

Spark Connect не предоставляет встроенных средств аутентификации, а gRPC-канал между клиентскими приложениями и сервером Spark Connect является незащищенным. При необходимости одним из способов защитить данный канал связи является использование gRPC-шлюзов.

Работа с Livy

Константин Алпашкин

Содержание

Назад к содержанию

- Требования к кластеру
  - Добавление прав доступа
  - Активация имперсонации в Livy
- Пример создания сессии
- Просмотр статуса сессии
- Удаление сессии
- Частые проблемы

Apache Livy — это компонент сервиса Spark3, позволяющий взаимодействовать с кластером Spark через HTTP. Livy предоставляет REST API, с помощью которого можно вызывать функции Spark, например:

- создание Spark-задачи;
- синхронное/асинхронное получение результатов;
- управление Spark Context.

Данный раздел демонстрирует основные операции с Livy, такие как создание/просмотр/удаление сессии, а также содержит информацию об использовании имперсонации в Livy. Все примеры в данном разделе относятся к типу сессий Batch, однако REST API для работы с интерактивными (Interactive) сессиями очень схож. Batch-сессии предназначены для асинхронного, неинтерактивного выполнения Spark-задач и создаются автоматически при запуске приложения в batch-режиме.

Больше подробной информации доступно на странице Apache Livy REST API.

> **РЕКОМЕНДАЦИЯ**
>
> Рекомендуется использовать сервер Livy, поставляемый с сервисом Spark3. Такой сервер использует порт `8999` и содержит последние исправленные баги.

Требования к кластеру

Для выполнения примеров, представленных в этом разделе, необходимо подготовить кластер ADH.

Добавление прав доступа

Сначала добавьте необходимые права доступа директориям Hadoop и скопируйте JAR-файлы с примерами Spark в HDFS.

```
$ sudo -u hdfs hdfs dfs -mkdir /user/livyDemoUser
$ sudo -u hdfs hdfs dfs -chown livyDemoUser:hadoop /user/livyDemoUser
$ sudo -u hdfs hdfs dfs -chmod -R a+rwx /user/livyDemoUser
$ sudo -u hdfs hdfs dfs -chmod -R a+rwx /var/log/spark3/apps
$ sudo -u hdfs hdfs dfs -mkdir /user/livyDemoUser/spark_examples
$ sudo -u hdfs hdfs dfs -put /usr/lib/spark3/examples/jars/* /user/livyDemoUser/spark_examples
```

Активация имперсонации в Livy

Когда Livy получает HTTP-запрос и создает новую задачу YARN/Spark, то по умолчанию новый процесс задачи запускается под пользователем `livy`. Имперсонация Livy позволяет заменить имя пользователя и запустить процесс от имени другого пользователя. Чтобы включить имперсонацию, выполните следующие действия:

1. В разделе настроек ADCM ***<YourClusterName>*** → ***Services*** → ***HDFS*** → ***Custom core-site.xml*** укажите свойства `hadoop.proxyuser.livy.groups = *` и `hadoop.proxyuser.livy.hosts = *`. Разрешение всех групп/хостов (`*`) является плохой практикой с точки зрения безопасности и используется только для демонстрационного примера.
2. Убедитесь, что файл */etc/livy/conf/livy.conf* (находится на том же хосте, где установлен сервер Livy) содержит параметр `livy.impersonation.enabled = True`.
3. Выполните рестарт сервисов HDFS, YARN и Spark.
4. Если включен плагин Ranger, необходимо добавить соответствующие политики для пользователя `livyDemoUser`, который фигурирует в примерах.

Пример создания сессии

Ниже показан пример создания новой сессии (Batch), которая запускает задачу SparkPi:

1. Создайте JSON-файл, который будет отправлен на сервер Livy в теле запроса HTTP.

```
{
    "file": "/user/livyDemoUser/spark_examples/spark-examples_2.11-2.3.2.3.1.0.15-1.jar", ①
    "className": "org.apache.spark.examples.SparkPi", ②
    "name": "test-livy-1", ③
    "args": [10], ④
    "proxyUser": "livyDemoUser" ⑤
}
```

① Путь к JAR-файлу с примером Spark.

② Основной класс приложения SparkPi.

③ Имя новой сессии. Имя должно быть уникальным для всего кластера.

④ Список аргументов командной строки, передаваемых приложению Spark.

⑤ Имя пользователя для имперсонации. От этого пользователя будет запущен новый процесс задачи Spark.

2. Отправьте POST-запрос для создания новой batch-сессии.

```
$ curl -X POST 'http://<livyServerHost>:8999/batches'
-H "X-Requested-By: livyDemoUser" ①
-d @request.json -H 'Content-Type: application/json'
```

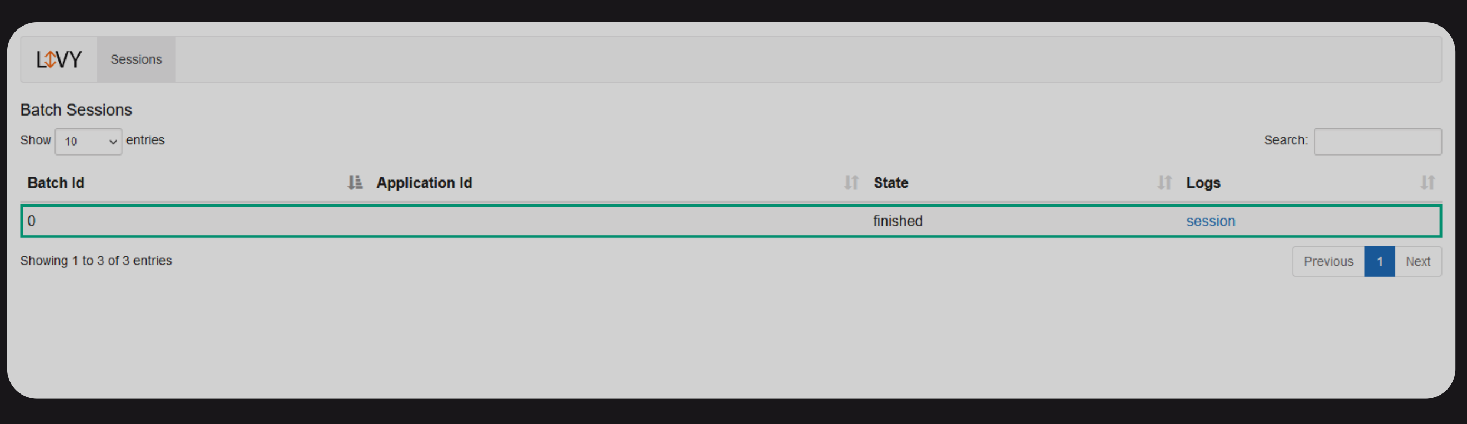
① Обязательный HTTP-заголовок, необходимый для CSRF-защиты

> **ПРИМЕЧАНИЕ**
>
> Если ваш кластер ADH керберизирован, необходимо активировать механизм SPNEGO для curl. Для этого добавьте флаг `--negotiate -v -u:` к вашим запросам curl.

Ответ от Livy-сервера имеет следующий вид:

```
{"id":1,"name":"test-livy-2","state":"starting","appId":null,"appInfo":{"driverLogUrl":null,"sparkUiUrl":null},"log":["stdout:
","\nstderr: ","\nYARN Diagnostics: "]}
```

Получив запрос, Livy создает новую сессию с именем `test-livy-1` (значение поля `name` в request.json). Новая сессия отображается в интерфейсе Livy.



Batch Sessions

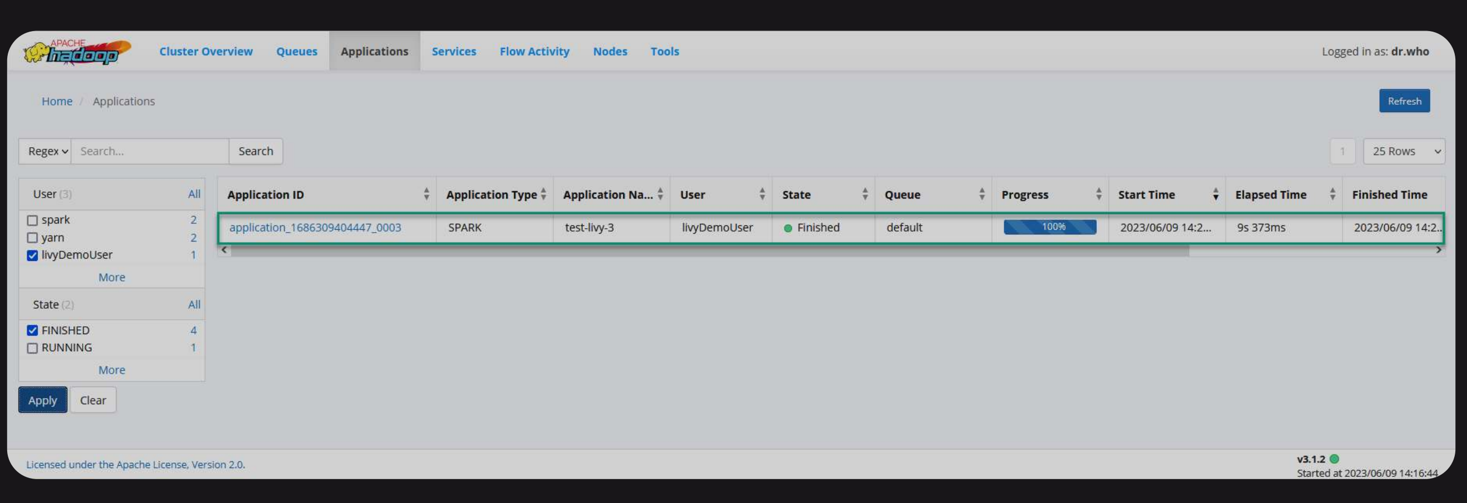
Show 10 entries Search:

Batch Id	Application Id	State	Logs
0		finished	session

Showing 1 to 3 of 3 entries

Новая сессия в Livy UI

Также новая сессия видна в интерфейсе YARN(UI2).



Application ID	Application Type	Application Na...	User	State	Queue	Progress	Start Time	Elapsed Time	Finished Time
application_1686309404447_0003	SPARK	test-livy-3	livyDemoUser	Finished	default	100%	2023/06/09 14:2...	9s 373ms	2023/06/09 14:2...

Новая сессия в YARN

Просмотр статуса сессии

Для получения статуса batch-сессии выполните следующий запрос.

```
$ curl -X GET 'http://<livyServerHost>/batches/x'
```

Где `x` — это ID сессии, например `123`. Полученный ответ имеет следующий вид:

```
{"id":2,"name":"test-livy-4","state":"starting","appId":null,"appInfo":{"driverLogUrl":null,"sparkUiUrl":null},"log": [...]}
```

Для получения списка всех сессий выполните следующий запрос.

```
$ curl -X GET 'http://<livyServerHost>/batches'
```

Ответ содержит список всех сессий, например:

```
{"from":0,"total":4,"sessions":[{"id":0,"name":"test-livy-1","state":"finished","appId":null,"appInfo":
{"driverLogUrl":null,"sparkUiUrl":null},"log":[...]},...]}
```

Удаление сессии

Чтобы удалить batch-сессию, выполните следующий запрос.

```
$ curl -X DELETE 'http://<livyServerHost>:8999/batches/x'
-H 'X-Requested-By: livyDemoUser'
```

Где `x` — это ID batch-сессии, которую нужно удалить.

Частые проблемы

Ниже описаны наиболее частые проблемы, с которыми вы можете столкнуться при работе с Livy:

- Ошибка имперсонации.
  Вы можете увидеть следующую ошибку в логах Livy:

```
org.apache.hadoop.security.authorize.AuthorizationException: User: livy is not allowed to impersonate <username> ...
```

  Данная ошибка сигнализирует о некорректной настройке имперсонации. Подробности настройки описаны в секции Активация имперсонации в Livy.

- Ошибка `HTTP ERROR: 400` в ответе curl.
  Данная ошибка говорит об отсутствии обязательного HTTP-заголовка для защиты от CSRF. Добавьте заголовок (`-H "X-Requested-By: <userName>"`) при отправке запросов curl.
- Ошибки, связанные с политиками Ranger.
  Если плагин Ranger включен, пользователь не может создавать новые сессии в дефолтной очереди YARN. Для корректной работы необходимо добавить соответствущие политики Ranger для YARN.

Планирование задач Spark

Константин Алпашкин

Содержание

Назад к содержанию

- Планирование на уровне кластера
  - Статическое выделение ресурсов
  - Динамическое выделение ресурсов
  - Использование внешнего shuffle-сервиса
- Планирование на уровне приложения
  - Режимы FIFO и справедливого распределения
  - Пул справедливого планировщика

При использовании Spark в режиме кластера существует две области, где применимо планирование и распределение ресурсов:

- Планирование на уровне кластера. Предполагает планирование отдельных Spark-приложений, которые выполняются на разных узлах кластера.
- Планирование на уровне приложения. Предполагает планирование параллельных задач Spark, например `collect`, `save` и других действий, выполняющихся внутри одного приложения Spark.

Планирование на уровне кластера

Для запуска Spark в режиме кластера используется кластерный менеджер YARN, задачей которого является планирование Spark-задач и выделение для них ресурсов на каждом узле кластера.

Статическое выделение ресурсов

Один из способов управления распределением ресурсов в YARN — это указание фиксированного объема ресурсов при запуске скрипта `spark3-submit` (статическое распределение ресурсов). Ниже показан пример запуска Spark в YARN, где с помощью специальных параметров указан объем выделяемых ресурсов.

```
$ ./bin/spark3-submit --class <app-class-name> \
    --master yarn \ ①
    --deploy-mode cluster \ ②
    --driver-memory 4g \ ③
    --num-executors ④
    --executor-memory 2g \ ⑤
    --executor-cores 1 \ ⑥
    ...
```

① Тип менеджера кластера.

② Режим деплоя, при котором код программы драйвера выполняется на worker-узлах.

③ Максимальное количество памяти, выделяемое для программы-драйвера.

④ Количество Spark executor, выделяемых на кластере.

⑤ Максимальное количество памяти, выделяемое на один executor.

⑥ Количество ядер ЦПУ, выделяемое на один executor.

Динамическое выделение ресурсов

Альтернативным подходом к распределению ресурсов кластера является встроенная в Spark функция динамического распределения ресурсов. Данная опция позволяет Spark динамически контролировать количество executor в зависимости от нагрузки. Для активации этой функции в ADH-сервисах Spark/Spark3, используйте действие Switch dynamic resource allocation в интерфейсе ADCM.

Использование внешнего shuffle-сервиса

После включения динамического распределения ресурсов следующим шагом может стать переход на внешний shuffle-сервис для улучшения надежности и ресурсопотребления. Основной задачей Spark shuffle-сервиса (Shuffle Service, SS) является обмен блоками данных между несколькими Spark executor или между executor и Spark-драйвером на этапе перетасовки (shuffle). По умолчанию Spark использует реализацию SS для YARN, которая запускается на каждом NodeManager YARN-кластера. Данная реализация SS доступна в */usr/lib/spark3/yarn/spark-{version}-yarn-shuffle.jar*. Однако эта реализация SS имеет один недостаток — в случае сбоя Spark executor удаляются все shuffle-данные, записанные на диск.

Для обеспечения максимальной надежности и скорости можно использовать внешний SS. Ниже описаны шаги по его подключению к кластеру Spark:

1. Добавьте JAR с вашей реализацией SS в classpath, например */usr/lib/spark3/**. В качестве примера для создания своей реализации SS можно использовать */usr/lib/spark3/yarn/spark-{version}-yarn-shuffle.jar*.
2. В ADCM на странице ***Clusters* → *<YOUR_CLUSTER_NAME>* → *Services* → *YARN* → *Primary configuration*** укажите имя вашего класса SS в поле `yarn.nodemanager.aux-services.spark_shuffle.class`.
3. Для каждого NodeManager увеличьте размер heap с помощью параметра `YARN_HEAPSIZE` (по умолчанию 1000) в файле *etc/hadoop/yarn-env.sh*. Это необходимо для избежания проблем с GC на этапе перетасовки.
4. Выполните рестарт YARN.

Вы также можете указать конфигурацию SS в файле *spark-shuffle-site.xml* и поместить этот файл в classpath SS. Подробная информация о конфигурационных параметрах SS доступна в документации Spark.

Планирование на уровне приложения

Внутри одного приложения Spark (здесь и далее под "приложением" стоит понимать один экземпляр SparkContext) планирование и распределение ресурсов также применимо, поскольку несколько Spark-задач могут выполняться параллельно, если они запущены из разных потоков. В данной статье "задача" означает действие Spark, например `reduce`, `save`, `collect` и прочие, а также любые другие задачи, необходимые для выполнения этого действия.

Режимы FIFO и справедливого распределения

По умолчанию задачи Spark выполняются в порядке FIFO. Каждая задача разбивается на этапы (например, фазы map/reduce), и задача, которая запустилась первой, получает приоритет на все доступные ресурсы, пока не завершены все ее этапы. Затем приоритет получает вторая задача и так далее. Если стоящей в начале очереди задаче не требуются все доступные ресурсы, то последующие задачи могут стартовать параллельно. Однако, если задачи в начале очереди являются трудоемкими, то выполнение последующих задач может существенно задержаться.

Помимо FIFO, Spark может выполнять задачи в режиме справедливого распределения (fair sharing mode). В этом режиме Spark распределяет задачи по принципу round-robin, так что все задачи получают примерно равные доли ресурсов кластера. Таким образом маленькие задачи, запущенные во время выполнения большой задачи, могут сразу же начать получать ресурсы, не дожидаясь завершения большой задачи, и при этом демонстрировать хорошее время отклика. По умолчанию режим справедливого распределения отключен, его можно включить, установив для экземпляра SparkContext значение `spark.scheduler.mode=FAIR`, как показано ниже:

```
val conf = new SparkConf()
          .setMaster(...)
          .setAppName(...)
conf.set("spark.scheduler.mode", "FAIR")
val sc = new SparkContext(conf)
```

Пул справедливого планировщика

Spark позволяет группировать задачи в пулы (pool) и устанавливать различные параметры планирования (например, вес) для каждого пула. Это может быть полезно для создания высокоприоритетных пулов для важных задач.

По умолчанию все задачи попадают в дефолтный пул. Чтобы направить задачу в определенный пул, необходимо установить локальное свойство `spark.scheduler.pool` для вашего экземпляра SparkContext. Это необходимо сделать в том же потоке, из которого запускаются задачи, например:

```
sc.setLocalProperty("spark.scheduler.pool", "mPool")
```

По умолчанию каждый пул получает равную долю ресурсов кластера, а внутри каждого пула задачи выполняются в порядке FIFO. Например, если создать один пул для каждого пользователя, то каждый пользователь получит равную долю ресурсов кластера, а запросы от каждого пользователя будут выполняться один за другим.

Поведение пулов планировщика можно настроить с помощью следующих параметров:

- `schedulingMode`. Определяет, порядок выполнения задач в пуле. Допустимые значения:
  - `FIFO` — задачи выполняются по очереди, является значением по умолчанию.
  - `FAIR` — ресурсы пула распределяются справедливо между задачами.
- `weight`. Устанавливает приоритет пула по отношению к другим пулам. Например, если установить вес пула `2`, то такой пул получит в 2 раза больше ресурсов, чем другие активные пулы. Используя большие значения веса (например, `1000`), можно реализовать приоритет для нескольких пулов. По умолчанию вес равен `1`.
- `minShare`. Кроме веса, для каждого пула может быть задана минимальная доля (в виде количества ядер ЦПУ). Планировщик Spark всегда пытается удовлетворить минимальные доли всех активных пулов, прежде чем перераспределять дополнительные ресурсы в соответствии с весами. Таким образом, свойство `minShare` является еще одним способом обеспечить быстрое получение пулом определенного количества ресурсов (например, 10 ядер) без предоставления ему высокого приоритета. По умолчанию для каждого пула свойство `minShare` равно `0`.

Приведенные выше свойства необходимо указать в файле *fairscheduler.xml*, аналогичном */etc/spark3/conf.dist/fairscheduler.xml.template*, как показано ниже.

```
<allocations>
  <pool name="production">
    <schedulingMode>FAIR</schedulingMode>
    <weight>1</weight>
    <minShare>2</minShare>
  </pool>
  <pool name="test">
    <schedulingMode>FIFO</schedulingMode>
    <weight>2</weight>
    <minShare>3</minShare>
  </pool>
</allocations>
```

Затем следует либо поместить файл *fairscheduler.xml* в classpath, либо установить свойство SparkConf `spark.scheduler.allocation.file`, которое указывает на файл *fairscheduler.xml*. Например:

```
conf.set("spark.scheduler.allocation.file", "file:///path/to/local-file")
conf.set("spark.scheduler.allocation.file", "hdfs:///path/to/hdfs-file")
```

Более детальная информация о свойствах пула для справедливого планировщика Spark доступна в документации Spark.

Логирование в Spark

Константин Алпашкин

Содержание

Назад к содержанию

- Просмотр логов
  - Текстовые логи
  - Веб-интерфейс History Server
- Настройка логирования

По умолчанию логирование в Spark включено, и все логи хранятся в HDFS-директории */var/log/spark3/apps*. Внутри этой директории Spark создает поддиректории с логами отдельных Spark-приложений.

Такие компоненты Spark, как History Server и Livy, также генерируют логи и сохраняют их в локальной файловой системе в следующих директориях:

- */var/log/spark3/history* — хранит логи History Server.
- */var/log/livy-spark3* — хранит логи Livy.

Просмотр логов

Существует два способа просмотра логов Spark:

- В виде обычных текстовых файлов.
- Используя веб-интерфейс History Server. В этом случае логи Spark должны храниться в HDFS, чтобы History Server мог получить доступ ко всем логам, которые генерируются на различных узлах кластера.

Текстовые логи

Spark сохраняет логи в виде JSON-файлов, которые состоят из отдельных записей, называемых **событиями (events)**. Для просмотра событий в текстовом виде выполните следующие действия:

1. Выведите содержимое базовой директории с логами (`spark.eventLog.dir`).

```
$ sudo -u hdfs hdfs dfs -ls /var/log/spark3/apps
```

В выводе отображаются лог-файлы различных Spark-приложений.

```
Found 3 items
-rwxrwx---   3 admin hadoop      57123 2023-08-24 13:32 /var/log/spark3/apps/application_1690315534352_0005
-rw-rw----   3 admin hadoop     126205 2023-08-24 14:52 /var/log/spark3/apps/application_1690315534352_0006
-rw-rw----   3 admin hadoop     235626 2023-08-25 09:14 /var/log/spark3/apps/application_1690315534352_0007
```

2. Выведите содержимое определенного лог-файла с помощью следующей команды.

```
$ sudo -u hdfs hdfs dfs -tail /var/log/spark3/apps/application_1690315534352_0007
```

Содержимое лог-файла имеет следующий вид:

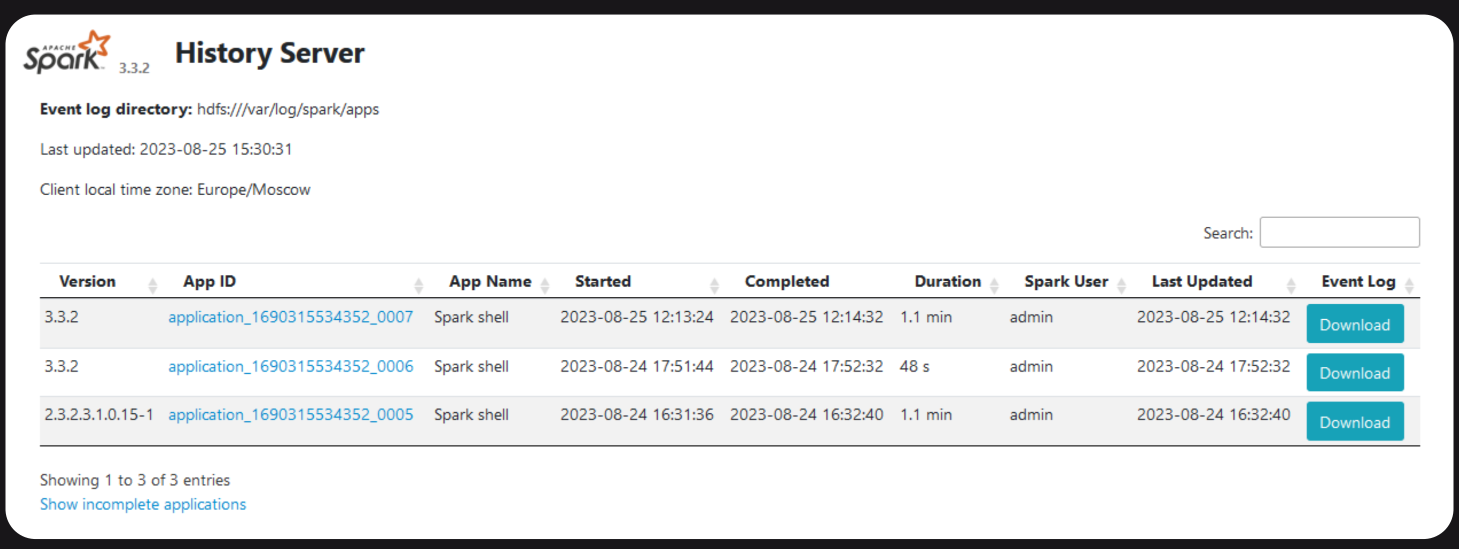
```
... ID":"2","Executor Info":{"Host":"ka-adh-2.ru-central1.internal","Total Cores":1,"Log Urls":{"stdout":"http://ka-adh-2.ru-
central1.internal:8042/node/containerlogs/container_e01_1690315534352_0006_01_000003/admin/stdout?
start=-4096","stderr":"http://ka-adh-2.ru-central1.
internal:8042/node/containerlogs/container_e01_1690315534352_0006_01_000003/admin/stderr?start=-4096"},"Attributes":
{"NM_HTTP_ADDRESS":"ka-adh-2.ru-central1.
internal:8042","USER":"admin","LOG_FILES":"stderr,stdout","NM_HTTP_PORT":"8042","CLUSTER_ID":"","NM_PORT":"8041","HTTP_SCHEME":"h
ttp://","NM_HOST":"ka-adh-2.ru-central1.internal","CONTAINER_ID":"container_e01_1690315534352_0006_01_000003"},"Resources":
{},"Resource Profile Id":0}}
{"Event":"SparkListenerBlockManagerAdded","Block Manager ID":{"Executor ID":"2","Host":"ka-adh-2.ru-central1.
internal","Port":46065},"Maximum Memory":384093388,"Timestamp":1692888721218,"Maximum Onheap Memory":384093388,"Maximum Offheap
Memory":0}
{"Event":"SparkListenerApplicationEnd","Timestamp":1692888752753}
```

Веб-интерфейс History Server

Компонент Spark3 History Server предоставляет веб-интерфейс для визуализации событий завершенных Spark-приложений. Веб-интерфейс доступен по адресу *http://<history-host>:<history-port>*, где:

- <history-host> — адрес хоста, на котором установлен компонент Spark3 History Server.
- <history-port> — порт по умолчанию (`18092`). Номер порта можно изменить с помощью параметра `spark.history.ui.port` в ADCM.

Веб-интерфейс History Server выглядит следующим образом.



Event log directory: hdfs:///var/log/spark/apps

Last updated: 2023-08-25 15:30:31

Client local time zone: Europe/Moscow

Version	App ID	App Name	Started	Completed	Duration	Spark User	Last Updated	Event Log
3.3.2	application_1690315534352_0007	Spark shell	2023-08-25 12:13:24	2023-08-25 12:14:32	1.1 min	admin	2023-08-25 12:14:32	Download
3.3.2	application_1690315534352_0006	Spark shell	2023-08-24 17:51:44	2023-08-24 17:52:32	48 s	admin	2023-08-24 17:52:32	Download
2.3.2.3.1.0.15-1	application_1690315534352_0005	Spark shell	2023-08-24 16:31:36	2023-08-24 16:32:40	1.1 min	admin	2023-08-24 16:32:40	Download

Showing 1 to 3 of 3 entries
Show incomplete applications

Веб-интерфейс Spark History Server

Spark3 History Server ищет лог-файлы событий в директории, которая задается параметром `spark.history.fs.logDirectory` в ADCM (по умолчанию это директория *hdfs:///var/log/spark/apps*). Обычно параметры `spark.history.fs.logDirectory` и `spark.eventLog.dir` должны указывать на одну и ту же директорию, чтобы History Server имел доступ ко всем лог-файлам, которые генерирует Spark.

Настройка логирования

Под капотом Spark использует библиотеку `Log4j` для ведения логов. Технически вы можете управлять процессом логирования, внося изменения в файл *log4j.properties* (файл можно создать из шаблона */etc/spark3/conf.dist/log4j.properties.template* и сохранить в этой же директории). Описание работы с *log4j.properties* выходит за рамки данной статьи, однако ниже приведены ADCM-параметры сервиса Spark3, с помощью которых можно настроить логирование для Spark.

Параметр	Описание
spark.eventLog.enabled	Включает/отключает логирование Spark
spark.eventLog.dir	Базовая директория для хранения лог-файлов Spark. Чтобы History Server имел доступ ко всем лог-файлам, которые генерирует Spark, данный параметр должен указывать на ту же директорию, что и `spark.history.fs.logDirectory`
spark.history.fs.cleaner.enabled	Активирует очистку логов событий в хранилище для компонента History Server
spark.history.store.path	Путь в локальной файловой системе для кеширования history-данных приложений Spark
spark.history.ui.port	Порт, на котором доступен веб-интерфейс Spark History Server
spark.history.fs.logDirectory	Директория, из которой History Server подбирает лог-файлы событий

Оптимизация производительности в Spark

Конастантин Алпашкин

Содержание

Назад к содержанию

- Кеширование данных в памяти
- Использование подсказок JOIN
- Использование подсказок разбиения
- Адаптивное выполнение запросов
  - Объединение разделов после перетасовки
  - Преобразование sort-merge JOIN в broadcast hash JOIN
  - Преобразование sort-merge JOIN в shuffled hash JOIN
  - Оптимизация перекосов в JOIN

В этом разделе описаны методы оптимизации производительности Spark, а также приведены соответствующие конфигурационные параметры для тонкой настройки этих функций.

ПРИМЕЧАНИЕ

В данной статье "Spark" подразумевает ADH-сервис Spark3.

Параметры конфигурации Spark, описанные в статье, можно указать с помощью `sparkSession.setConf("property=value")` или с помощью команд Spark SQL типа `SET property=value`. Больше информации о параметрах, используемых для настройки производительности, доступно в документации Spark.

Кеширование данных в памяти

Spark SQL позволяет кешировать таблицы в памяти, используя специальный столбцовый формат, с помощью методов `spark.catalog.cacheTable("tableName")` или `dataFrame.cache()`. После вызова этих методов Spark будет сканировать только необходимые столбцы и автоматически сжимать данные для минимизации использования памяти и нагрузки на GC. Чтобы удалить таблицу из памяти, используйте `spark.catalog.uncacheTable("tableName")` или `dataFrame.unpersist()`.

Свойство	Описание	Значение по умолчанию
spark.sql.inMemoryColumnarStorage.compressed	При значении `true` Spark автоматически подбирает кодек сжатия для каждого столбца, основываясь на статистике о данных в столбце	true
spark.sql.inMemoryColumnarStorage.batchSize	Устанавливает размер пакета при кешировании столбца. Установка большего размера пакета позволяет более эффективно использовать память и обеспечивает лучший уровень сжатия, однако увеличивает риск OOM-исключений	10000
spark.sql.files.maxPartitionBytes	Максимальное количество байт, помещаемых в один раздел (partition) при чтении файла. Данный параметр актуален при работе с файловыми данными типа Parquet, ORC и JSON	134217728 (128 МБ)
spark.sql.files.openCostInBytes	Влияет на количество создаваемых разделов, в которые будут записаны данные при чтении. Расчетная "стоимость" открытия файла, измеряемая в количестве байт, которые могут быть прочитаны за раз. Данный параметр используется при записи нескольких файлов в один раздел. Рекомендуется оценивать стоимость с запасом — в таком случае разделы с маленькими файлами обрабатываются быстрее, чем разделы с бОльшими файлами (запланированные на выполнение первыми). Данный параметр актуален при работе с файловыми данными типа Parquet, ORC и JSON	4194304 (4 МБ)
spark.sql.files.minPartitionNum	Желаемое (однако не гарантируемое) минимальное количество разделов при разбиении файла. Если параметр не задан, используется значение свойства `spark.default.parallelism`. Данный параметр актуален при работе с файловыми данными типа Parquet, ORC и JSON	Значение свойства `spark.default.parallelism`
spark.sql.broadcastTimeout	Тайм-аут (в секундах) на выполнение операции трансляции при выполнении broadcast JOIN	300
spark.sql.autoBroadcastJoinThreshold	Максимальный размер таблицы (в байтах), перед тем как данные таблицы будут транслированы (broadcast) на все worker-узлы при выполнении JOIN. Установка значения `-1` отключает трансляцию данных	10485760 (10 МБ)
spark.sql.shuffle.partitions	Устанавливает количество разделов, используемых при перетасовке данных в операциях JOIN/агрегации	200
spark.sql.sources.parallelPartitionDiscovery.threshold	Устанавливает предельное значение для активации параллельного листинга при чтении входных путей (input paths). Если количество входных путей превышает это значение, Spark выполняет листинг, используя распределенную задачу. В остальных случаях Spark использует последовательный листинг. Данный параметр актуален при работе с файловыми данными типа Parquet, ORC и JSON	32
spark.sql.sources.parallelPartitionDiscovery.parallelism	Устанавливает значение максимального параллелизма листинга для входных путей задачи. Если количество входных путей больше, чем значение данного параметра, оно будет урезано до значения параметра. Данный параметр актуален при работе с файловыми данными типа Parquet, ORC и JSON	10000

Использование подсказок JOIN

Операции JOIN в Spark SQL — это преобразования, подразумевающие массовую перетасовку данных по сети, которые при неправильной конфигурации могут ухудшать производительность. Используя подсказки для JOIN-стратегий (JOIN hints), можно "рекомендовать" Spark использовать определенную стратегию JOIN для конкретной таблицы. Вы можете значительно улучшить производительность JOIN-операций, выбирая стратегии, которые лучше всего подходят для конкретных таблиц. Ниже показан пример объединения двух таблиц с использованием подсказки `BROADCAST`.

```
val joined_table = spark.table("src").join(spark.table("records").hint("broadcast"), "key")
```

Если для таблицы указано несколько подсказок, Spark применяет подсказки со следующим приоритетом:

1. BROADCAST
2. MERGE
3. SHUFFLE_HASH
4. SHUFFLE_REPLACE_NL

ПРИМЕЧАНИЕ

Использование подсказки не дает гарантии, что Spark будет использовать именно рекомендуемую стратегию, поскольку указанная стратегия может поддерживать не все типы JOIN.

Использование подсказок разбиения

В Spark SQL подсказки разбиения (partition hints) позволяют "рекомендовать" определенную стратегию для выполнения разбиения (partitioning). Основные подсказки разбиения показаны ниже:

- **COALESCE**. Позволяет контролировать количество выходных файлов. Принимает количество партиций в качестве параметра.
- **REPARTITION**. Используется для переразбиения на нужное количество партиций. В качестве параметров может принимать количество партиций, имена столбцов, оба значения, либо использоваться без параметров.
- **REPARTITION_BY_RANGE**. Принимает имена столбцов и количество партиций (опционально).
- **REBALANCE**. Используется для перераспределения результатов запроса с целью сделать партиции примерно одинакового размера, без существенного перекоса.

Примеры подсказок разбиения показаны ниже.

```
SELECT /*+ COALESCE(5) */ * FROM t
SELECT /*+ REPARTITION(5) */ * FROM t
SELECT /*+ REPARTITION(c) */ * FROM t
SELECT /*+ REPARTITION(5, c) */ * FROM t
SELECT /*+ REPARTITION */ * FROM t
SELECT /*+ REPARTITION_BY_RANGE(c) */ * FROM t
SELECT /*+ REPARTITION_BY_RANGE(5, c) */ * FROM t
SELECT /*+ REBALANCE */ * FROM t
SELECT /*+ REBALANCE(5) */ * FROM t
SELECT /*+ REBALANCE(c) */ * FROM t
SELECT /*+ REBALANCE(5, c) */ * FROM t
```

Адаптивное выполнение запросов

Адаптивное выполнение запросов (Adaptive Query Execution, AQE) — это техника оптимизации, которая на основе статистики выполнения подбирает наиболее эффективный план выполнения запроса. AQE включена по умолчанию и может быть отключена с помощью свойства `spark.sql.adaptive.enabled`. Основные функции AQE описаны ниже.

Объединение разделов после перетасовки

Эта функция AQE объединяет разделы (partitions coalescence) после перетасовки на основе выходной статистики, что позволяет не указывать вручную количество разделов для перетасовки, которое бы подходило вашему набору данных. Вместо этого Spark сам подберет подходящее число разделов во время выполнения (при условии, что задано достаточно большое начальное число разделов с помощью свойства `spark.sql.adaptive.coalescePartitions.initialPartitionNum`).

Эта функция активируется, если заданы параметры `spark.sql.adaptive.enabled=true` и `spark.sql.adaptive.coalescePartitions.enabled=true`.

Свойство	Описание	Значение по умолчанию
spark.sql.adaptive.coalescePartitions.enabled	Если указано значение `true` и `spark.sql.adaptive.enabled=true`, Spark будет объединять смежные разделы перетасовки в соответствии с целевым размером раздела (определяется свойством `spark.sql.adaptive.advisoryPartitionSizeInBytes`), чтобы избежать слишком большого количества маленьких задач	true
spark.sql.adaptive.coalescePartitions.parallelismFirst	Если значение равно `true`, то при слиянии смежных разделов Spark игнорирует целевой размер, указанный в `spark.sql.adaptive.advisoryPartitionSizeInBytes` (по умолчанию равен 64 МБ), и учитывает только минимальный размер раздела, определяемый свойством `spark.sql.adaptive.coalescePartitions.minPartitionSize` (по умолчанию 1 МБ). Это позволяет добиться максимального эффекта параллелизма. Данные действия позволяют избежать ухудшения производительности при включенном AQE. Рекомендуется использовать значение `false` и соблюдать целевой размер, устанавливаемый свойством `spark.sql.adaptive.advisoryPartitionSizeInBytes`	true
spark.sql.adaptive.coalescePartitions.minPartitionSize	Минимальный размер разделов перетасовки после слияния. Значение должно составлять не более 20% от значения параметра `spark.sql.adaptive.advisoryPartitionSizeInBytes`. Этот параметр может быть полезен, если целевой размер игнорируется при слиянии разделов (поведение по умолчанию)	1 МБ
spark.sql.adaptive.coalescePartitions.initialPartitionNum	Начальное количество разделов перетасовки перед слиянием. Если значение не задано, используется значение параметра `spark.sql.shuffle.partitions`. Данный параметр имеет эффект только если установлены `spark.sql.adaptive.enabled=true` и `spark.sql.adaptive.coalescePartitions.enabled=true`	—
spark.sql.adaptive.advisoryPartitionSizeInBytes	Рекомендуемый размер (в байтах) разделов перетасовки при адаптивной оптимизации (`spark.sql.adaptive.enabled=true`). Данный параметр имеет эффект, если Spark объединяет небольшие разделы перетасовки или разбивает разделы перетасовки с перекосом	64 МБ

Преобразование sort-merge JOIN в broadcast hash JOIN

AQE автоматически преобразует JOIN сортировочного слияния (sort-merge JOIN) в транслируемый хеш-JOIN (broadcast hash JOIN), если статистика любой из сторон JOIN меньше порога адаптивного транслируемого хеш-JOIN (определяется параметром `spark.sql.adaptive.autoBroadcastJoinThreshold`). Это не так эффективно, как использование изначально транслируемого хеш-JOIN, но все же более эффективно, чем JOIN слияния, так как позволяет сохранить сортировку обеих сторон JOIN и локально обработать файлы перетасовки для уменьшения нагрузки на сеть (при условии, что `spark.sql.adaptive.localShuffleReader.enabled=true`).

Свойство	Описание	Значение по умолчанию
spark.sql.adaptive.autoBroadcastJoinThreshold	Устанавливает максимальный размер (в байтах) для таблицы, которая будет транслироваться на все worker-узлы при выполнении JOIN. Установка значения `-1` отключает трансляцию. Значение по умолчанию берется из параметра `spark.sql.autoBroadcastJoinThreshold`. Обратите внимание, что данный параметр используется только в AQE	—

Преобразование sort-merge JOIN в shuffled hash JOIN

AQE автоматически преобразует JOIN сортировочного слияния (sort-merge JOIN) в перетасованный хеш-JOIN (shuffled hash JOIN), если все разделы после сортировки меньше предельного значения, определяемого параметром `spark.sql.adaptive.maxShuffledHashJoinLocalMapThreshold`.

Свойство	Описание	Значение по умолчанию
spark.sql.adaptive.maxShuffledHashJoinLocalMapThreshold	Устанавливает максимальный размер (в байтах) для каждого раздела, который может быть использован для построения локальной хеш-таблицы. Если значение параметра больше или равно `spark.sql.adaptive.advisoryPartitionSizeInBytes`, и все разделы также не превышают значение данного параметра, Spark будет использовать перетасованный хеш-JOIN вместо сортированного JOIN, независимо от значения `spark.sql.join.preferSortMergeJoin`	0

Оптимизация перекосов в JOIN

Перекос данных (data skew) часто приводит к замедлению JOIN-операций. Данная функция AQE динамически обрабатывает перекос в JOIN сортировочного слияния (sort-merge JOIN), разбивая (и при необходимости реплицируя) перекошенные задачи на примерно одинаковые по размеру. Функция активируется при установке параметров `spark.sql.adaptive.enabled=true` и `spark.sql.adaptive.skewJoin.enabled=true`.

Свойство	Описание	Значение по умолчанию
spark.sql.adaptive.skewJoin.enabled	Если установлено значение `true` и `spark.sql.adaptive.enabled=true`, Spark динамически обрабатывает перекос в JOIN сортировочного слияния, разбивая (и реплицируя при необходимости) перекошенные разделы	true
spark.sql.adaptive.skewJoin.skewedPartitionFactor	Раздел считается перекошенным, если его размер (в байтах) больше данного коэффициента, умноженного на медианный размер раздела, а также если размер раздела больше значения `spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes`	5
spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	Раздел считается перекошенным, если его размер (в байтах) больше данного предельного значения, а также если размер раздела больше значения параметра `spark.sql.adaptive.skewJoin.skewedPartitionFactor`, умноженного на медианный размер раздела. Значение этого свойства должно быть больше, чем `spark.sql.adaptive.advisoryPartitionSizeInBytes`	256 МБ

Работа с таблицами Iceberg в Spark

Константин Алпашкин

Содержание

Назад к содержанию

- Каталоги Iceberg
  - Типы каталогов
  - SparkCatalog и SparkSessionCatalog
  - Параметры конфигурации каталогов
- Команды DDL
  - Создание таблицы Iceberg
  - Создание партиционированной таблицы
  - CREATE TABLE: CTAS/RTAS
  - DROP TABLE
  - ALTER TABLE
- Выборка данных
  - Time travel
  - Таблицы метаданных
- Запись данных
  - Запись данных в ветку
- Roll back

Apache Iceberg — это открытый, высокопроизводительный формат для больших аналитических таблиц. Сервис ADH Spark3 поддерживает данный формат, позволяя вам работать с таблицами Iceberg через Spark.

ПРИМЕЧАНИЕ

Spark3 поддерживает Iceberg-таблицы начиная с версии ADH 3.2.4.3.

В сервисе Spark3 поддержка Iceberg-таблиц включена по умолчанию. В этой статье для демонстрации примеров работы с Iceberg-таблицами используется Spark SQL — модуль, позволяющий выполнять традиционные ANSI SQL-команды на объектах Spark DataFrame.

Вы можете выполнять запросы Spark SQL, используя оболочку *spark3-sql*. Для запуска *spark3-sql* выполните следующую команду на хосте, где установлен компонент Spark3 Client.

```
$ spark3-sql
```

В данной статье все SQL-примеры предназначены для запуска в оболочке *spark3-sql*.

Каталоги Iceberg

Архитектура Iceberg добавляет в Spark API понятие каталогов (catalogs), которые позволяют обращаться к Iceberg-таблицам по имени. Каталог Iceberg — это именованное хранилище для таблиц и их метаданных, разделенное на пространства имен. Имена каталогов и пространства имен используются в SQL-запросах для идентификации таблиц, принадлежащих к разным каталогам, аналогично нотации `SELECT … FROM <database_name>.<table_name>`. Для доступа к конкретной Iceberg-таблице используется следующий синтаксис:

```
SELECT ... FROM <catalog_name>.<namespace>.<table_name> ...
```

Типы каталогов

Iceberg предоставляет несколько бэкендов каталогов, таких как REST, Hive, JDBC и прочие. Ниже показан пример создания Iceberg-каталога типа Hive, который загружает таблицы из Hive Metastore:

```
$ spark3-sql \
    --conf spark.sql.catalog.test_catalog_hive=org.apache.iceberg.spark.SparkCatalog \
    --conf spark.sql.catalog.test_catalog_hive.type=hive
```

Для обращения к таблицам внутри такого каталога необходимо добавить к именам таблиц префикс `test_catalog_hive`.

ПРИМЕЧАНИЕ

Параметры конфигурации каталогов, указанные с помощью `--conf` в примере выше, также можно указать в ADCM (***Clusters → <clusterName> → Services → Spark3 → Primary Configuration → Custom spark-defaults.conf***).

В сервисе Spark3 каталог по умолчанию называется `spark-catalog`. Данный каталог поддерживает одноуровневые пространства имен, которые соответствуют именам баз данных Hive. Примеры запросов в данной статье предназначены для выполнения в дефолтном каталоге `spark-catalog` с пространством имен `default`.

SparkCatalog и SparkSessionCatalog

Iceberg предоставляет 2 реализации каталогов для движка Spark:

- `org.apache.iceberg.spark.SparkCatalog`. Позволяет работать только с Iceberg-таблицами, используя Hive Metastore или warehouse-директорию в HDFS.
- `org.apache.iceberg.spark.SparkSessionCatalog`. Добавляет поддержку Iceberg-таблиц к функционалу встроенного Spark-каталога. При работе с обычными таблицами (не Iceberg), `SparkSessionCatalog` делегирует выполнение встроенному каталогу Spark. В сервисе Spark3 данная реализация используется по умолчанию, позволяя работать как с Iceberg-таблицами, так и с обычными (не Iceberg) таблицами.

Параметры конфигурации каталогов

Каталоги Iceberg можно настроить с помощью конфигурационных свойств `spark.sql.catalog.<catalog-name>.*`. Общие параметры конфигурации для Hive и Hadoop представлены в таблице ниже.

Свойство	Описание
spark.sql.catalog.<catalog-name>.type	Определяет реализацию Iceberg-каталога. Доступны следующие значения ▪ `hive` (HiveCatalog); ▪ `hadoop` (HadoopCatalog); ▪ `rest` (RESTCatalog); ▪ `glue` (GlueCatalog); ▪ `jdbc` (JdbcCatalog); ▪ `nessie` (NessieCatalog); ▪ пустое значение, если используется кастомная реализация каталога.
spark.sql.catalog.<catalog-name>.catalog-impl	Кастомная реализация каталога Iceberg. Если не задано значение `spark.sql.catalog.<catalog-name>.type`, значение свойства `spark.sql.catalog.<catalog-name>.catalog-impl` не должно быть пустым
spark.sql.catalog.<catalog-name>.io-impl	Пользовательская реализация FileIO
spark.sql.catalog.<catalog-name>.metrics-reporter-impl	Пользовательская реализация MetricsReporter
spark.sql.catalog.<catalog-name>.default-namespace	Пространство имен, используемое по умолчанию для каталога. Значение по умолчанию: `default`
spark.sql.catalog.<catalog-name>.uri	Hive Metastore URL для каталогов Hive (*thrift://host:port*) либо REST URL для каталогов типа REST
spark.sql.catalog.<catalog-name>.warehouse	Базовый путь к warehouse-директории. Например, *hdfs://nn:8020/warehouse/path*
spark.sql.catalog.<catalog-name>.cache-enabled	Включает или отключает кеширование для каталога. По умолчанию используется `true` (кеширование активно)
spark.sql.catalog.<catalog-name>.cache.expiration-interval-ms	Время, по истечении которого кешированные записи каталога будут удалены. Данный параметр актуален, если `spark.sql.catalog.<catalog-name>.cache-enabled=true`. Установка значения `-1` отключает срок действия кеша, а использование значения `0` полностью отключает механизм кеширования, независимо от значения `cache-enabled`. Значение по умолчанию: `30000` (30 секунд)
spark.sql.catalog.<catalog-name>.table-default.propertyKey	Устанавливает значение по умолчанию для свойства таблицы Iceberg с ключом `propertyKey`. Это значение будет использовано для всех таблиц, создаваемых в текущем каталоге, если только оно не переопределено
spark.sql.catalog.<catalog-name>.table-override.propertyKey	Устанавливает значение для свойства таблицы Iceberg с ключом `propertyKey`, которое не может быть переопределено

Дополнительная информация о создании и настройке каталогов доступна в документации Iceberg.

Команды DDL

Создание таблицы Iceberg

Для создания Iceberg-таблицы добавьте предложение `USING iceberg` к стандартному оператору `CREATE TABLE`.

Пример:

```
CREATE TABLE spark_catalog.default.transactions(
    txn_id int,
    acc_id int,
    txn_value double,
    txn_date date)
USING iceberg;
```

Выполните команду `DESCRIBE FORMATTED`, чтобы увидеть информацию о только что созданной таблице:

```
DESCRIBE FORMATTED spark_catalog.default.transactions;
```

Пример вывода:

```
txn_id                  int
acc_id                  int
txn_value               double
txn_date                date

# Metadata Columns
_spec_id                int
_partition              struct<>
_file                   string
_pos                    bigint
_deleted                boolean

# Detailed Table Information
Name                    spark_catalog.default.transactions
Type                    MANAGED
Location                hdfs://adh/apps/hive/warehouse/transactions
Provider                iceberg
Owner                   hdfs
Table Properties        [current-snapshot-id=none,format=iceberg/parquet,format-version=2,write.parquet.compression-codec=zstd]
```

Значение `Provider` в выводе указывает на то, что таблица была создана как Iceberg-таблица.

Создание партиционированной таблицы

Чтобы создать таблицу с партициями, используйте синтаксис, как показано в следующем примере:

```
CREATE TABLE spark_catalog.default.transactions_partitioned (
    txn_id bigint,
    acc_id int,
    txn_value double,
    txn_date date)
USING iceberg
PARTITIONED BY (acc_id);
```

Вы также можете использовать выражения преобразования (transform expressions) в предложении `PARTITIONED BY` для создания скрытых партиций вместо явного указания партиции. Пример приведен ниже.

```
CREATE TABLE spark_catalog.default.transactions_partitioned_transform (
    id bigint,
    acc_id int,
    txn_value double,
    txn_date timestamp)
USING iceberg
PARTITIONED BY (day(txn_date)); ①
```

① `day(txn_date)` — выражение преобразования для создания партиций по дням.

Чтобы проверить разбиение таблицы на партиции, запишите в таблицу тестовые данные с помощью следующей команды:

```
INSERT INTO spark_catalog.default.transactions_partitioned_transform VALUES
(1, 1002, 10.00, cast('2023-01-01' as timestamp)),
(2, 1001, 20.00, cast('2023-01-02' as timestamp));
```

Затем проверьте содержимое warehouse-директории в HDFS:

```
$ hdfs dfs -ls -R /apps/hive/warehouse/transactions_partitioned_transform
```

Пример вывода:

```
drwxrwxr-x   - hdfs hadoop          0 2024-06-19 16:06 /apps/hive/warehouse/transactions_partitioned_transform/data
drwxrwxr-x   - hdfs hadoop          0 2024-06-19 16:06
/apps/hive/warehouse/transactions_partitioned_transform/data/txn_date_day=2023-01-01
-rw-r--r--   3 hdfs hadoop       1206 2024-06-19 16:06
/apps/hive/warehouse/transactions_partitioned_transform/data/txn_date_day=2023-01-01/00000-12-a029426f-ca10-4efc-b87c-
389694c218bc-0-00001.parquet
drwxrwxr-x   - hdfs hadoop          0 2024-06-19 16:06
/apps/hive/warehouse/transactions_partitioned_transform/data/txn_date_day=2023-01-02
-rw-r--r--   3 hdfs hadoop       1206 2024-06-19 16:06
/apps/hive/warehouse/transactions_partitioned_transform/data/txn_date_day=2023-01-02/00000-12-a029426f-ca10-4efc-b87c-
389694c218bc-0-00002.parquet
drwxrwxr-x   - hdfs hadoop          0 2024-06-19 16:06 /apps/hive/warehouse/transactions_partitioned_transform/metadata
-rw-r--r--   3 hdfs hadoop       1381 2024-06-19 16:06 /apps/hive/warehouse/transactions_partitioned_transform/metadata/00000-
28c5f350-f14a-4fcb-b509-10a327fa19cb.metadata.json
-rw-r--r--   3 hdfs hadoop       2519 2024-06-19 16:06 /apps/hive/warehouse/transactions_partitioned_transform/metadata/00001-
7c83b7d2-0a43-463e-b180-6589a0f696af.metadata.json
-rw-r--r--   3 hdfs hadoop       7157 2024-06-19 16:06 /apps/hive/warehouse/transactions_partitioned_transform/metadata/28884384-
0539-4ca5-8dcb-95d00de4419e-m0.avro
-rw-r--r--   3 hdfs hadoop       4264 2024-06-19 16:06 /apps/hive/warehouse/transactions_partitioned_transform/metadata/snap-
2743337418634934372-1-28884384-0539-4ca5-8dcb-95d00de4419e.avro
```

Согласно выводу, тестовые данные были распределены по нескольким HDFS-директориям (партициям) с именами *txn_date_day={date}*, каждая из которых представляет отдельный день.

CREATE TABLE: CTAS/RTAS

Вы можете использовать синтаксис `CREATE TABLE AS <tbl_name>` (CTAS) и `REPLACE TABLE AS <tbl_name>` (RTAS) для создания/перезаписи таблицы Iceberg на основе уже существующей таблицы. Примеры приведены ниже.

```
CREATE TABLE spark_catalog.default.transactions_ctas USING iceberg
AS SELECT txn_id, txn_value
FROM spark_catalog.default.transactions;
```

```
REPLACE TABLE spark_catalog.default.transactions_rtas USING iceberg
AS SELECT txn_id, txn_value
FROM spark_catalog.default.transactions
WHERE txn_value < 50;
```

Операции создания/перезаписи таблицы с использованием синтаксиса CTAS/RTAS являются атомарными при использовании реализации каталога `SparkCatalog`. При использовании `SparkSessionCatalog` (используется по умолчанию в Spark3) атомарность операций CTAS/RTAS не гарантируется.

DROP TABLE

Для удаления Iceberg-таблицы используйте стандартный синтаксис:

```
DROP TABLE <table_name> [PURGE]
```

Необязательный флаг `PURGE` определяет, нужно ли удалять файлы данных/метаданных таблиц из хранилища (Ozone, HDFS и прочих). Если `PURGE` не установлен, из каталога удаляются только метаданные таблицы.

По умолчанию команда `DROP TABLE` не удаляет пустые директории из хранилища (даже с установленным флагом `PURGE`). Например:

```
hdfs dfs -ls -R /apps/hive/warehouse/my_test_table
drwxrwxr-x - hdfs hadoop 0 2025-05-29 15:22 /apps/hive/warehouse/my_test_table/data
drwxrwxr-x - hdfs hadoop 0 2025-05-29 15:22 /apps/hive/warehouse/my_test_table/metadata
```

Если необходимо, чтобы при выполнении `DROP TABLE` также удалялись директории таблиц, используйте следующие свойства при создании таблицы:

```
CREATE TABLE <table_name>(id int) USING ICEBERG
TBLPROPERTIES (
    'external.table.purge'='TRUE' ①
    'drop.base-directory.enabled'='TRUE', ②
 );
```

① Позволяет `DROP TABLE` удалять файлы данных/метаданных (эквивалент `PURGE`).

② Позволяет `DROP TABLE` удалять из хранилища (HDFS, Ozone и прочих) пустые директории таблиц.

ALTER TABLE

Сервис Spark3 предоставляет богатую поддержку операций `ALTER TABLE` для Iceberg-таблиц. Ниже показаны примеры основных операций `ALTER TABLE`. Более подробную информацию о командах `ALTER TABLE` для Iceberg-таблиц можно найти в документации Iceberg.

ALTER TABLE … RENAME TO

Следующая команда переименовывает таблицу Iceberg. Команда изменяет только метаданные таблицы, не затрагивая данные таблицы.

```
ALTER TABLE spark_catalog.default.transactions
RENAME TO spark_catalog.default.transactions_new;
```

ALTER TABLE … SET TBLPROPERTIES

Следующий пример устанавливает формат записи ORC, используя параметры таблиц Iceberg.

```
ALTER TABLE spark_catalog.default.transactions
SET TBLPROPERTIES ('write.format.default'='orc');
```

РЕКОМЕНДАЦИЯ

Чтобы очистить свойство таблицы, используйте команду `UNSET`.

ALTER TABLE … ADD COLUMN(S)

Вы можете добавить столбец в существующую таблицу Iceberg, используя предложение `ADD COLUMN`.

Пример:

```
ALTER TABLE spark_catalog.default.transactions
ADD COLUMN (
    description string comment "A transaction's description"
);
```

Вывод команды `DESCRIBE` после выполнения команды:

```
DESCRIBE spark_catalog.default.transactions;
txn_id                  int
acc_id                  int
txn_value               double
txn_date                date
description             string                  A transaction's description
```

Чтобы добавить несколько столбцов одной командой, используйте предложение `ADD COLUMNS`, разделив столбцы запятой.

Пример:

```
ALTER TABLE spark_catalog.default.transactions
ADD COLUMNS (
    is_committed boolean,
    is_acknowledged boolean
);
```

Вывод команды `DESCRIBE` после выполнения команды:

```
DESCRIBE spark_catalog.default.transactions;
txn_id                  int
acc_id                  int
txn_value               double
txn_date                date
description             string                  A transaction's description
is_committed            boolean
is_acknowledged         boolean
```

Для добавления столбца с композитным типом данных struct, например `struct<x, y>`, используйте следующий синтаксис:

```
ALTER TABLE spark_catalog.default.transactions
ADD COLUMN processor_metadata struct<x: double, y: double>;
```

Больше информации об использовании составных типов данных (массивы, карты и т. д.) доступно в документации Iceberg.

ALTER TABLE … ALTER COLUMN

Iceberg позволяет изменять определение столбцов. Например, вы можете изменить тип данных столбца, сделать столбец nullable, добавить комментарий и так далее. Iceberg позволяет расширить тип данных столбца, если расширение безопасно, например:

- `int` → `bigint`
- `float` → `double`
- `decimal(i,j)` → `decimal(i1,j)` при `i1` > `i`

```
ALTER TABLE spark_catalog.default.transactions
ALTER COLUMN txn_id TYPE bigint;
```

Вывод команды `DESCRIBE` после выполнения команды:

```
DESCRIBE spark_catalog.default.transactions;
txn_id                  bigint
acc_id                  int
txn_value               double
txn_date                date
description             string                  A transaction's description
is_committed            boolean
is_acknowledged         boolean
processor_metadata      struct<x:double,y:double>
```

Чтобы сделать поле nullable, используйте следующий синтаксис.

```
ALTER TABLE spark_catalog.default.transactions
ALTER COLUMN acc_id DROP NOT NULL;
```

Обратное изменение nullable-столбца на non-nullable недопустимо, поскольку Iceberg не располагает информацией о существующих записях со значениями `NULL`.

ALTER TABLE … ADD PARTITION FIELD

Iceberg поддерживает эволюцию схемы партиционирования, позволяя изменять партиции в существующих таблицах. Изменение схемы партиционирования является операцией над метаданными, которая не затрагивает существующие данные таблицы. После изменения схемы новые данные будут записаны в соответствии с обновленной схемой партиционирования, однако уже существующие данные будут соответствовать старой схеме. В старых файлах данных будут содержаться значения `NULL` для новых полей партиций.

Пример:

```
ALTER TABLE spark_catalog.default.transactions_partitioned
ADD PARTITION FIELD txn_date;
```

Выполните команду `DESCRIBE`, чтобы убедиться, что схема партиционирования таблицы была изменена. Пример вывода показан ниже.

```
DESCRIBE spark_catalog.default.transactions_partitioned;
txn_id                  bigint
acc_id                  int
txn_value               decimal(10,2)
txn_date                date
# Partition Information
# col_name              data_type               comment
acc_id                  int
txn_date                date
```

ВНИМАНИЕ

Изменение партиций в Iceberg-таблицах следует выполнять с осторожностью. Некоторые изменения могут привести к неожиданным результатам и к повреждению метаданных таблицы. Больше подробной информации доступно в разделах ADD PARTITION, DROP PARTITION и REPLACE PARTITION.

ALTER TABLE … WRITE ORDERED BY

Вы можете указать порядок сортировки для таблицы Iceberg, который используется для автоматической сортировки данных, записываемых в эту таблицу. Например, операция Spark MERGE INTO использует порядок сортировки таблицы.

```
ALTER TABLE spark_catalog.default.transactions
WRITE ORDERED BY txn_id;
```

ПРИМЕЧАНИЕ

Порядок записи в таблицу не гарантирует порядок данных при выборке. Он влияет только на то, как данные записываются в таблицу.

Чтобы отменить порядок сортировки таблицы, используйте ключевое слово `UNORDERED`, как показано в примере ниже.

```
ALTER TABLE spark_catalog.default.transactions
WRITE UNORDERED;
```

ALTER TABLE … CREATE BRANCH

Вы можете создавать ветки (branches) для Iceberg-таблиц с помощью оператора `CREATE BRANCH`. Ниже приведены несколько примеров.

```
ALTER TABLE spark_catalog.default.transactions CREATE BRANCH `test-branch`; ①
ALTER TABLE spark_catalog.default.transactions CREATE BRANCH IF NOT EXISTS `test-branch` RETAIN 7 DAYS; ②
ALTER TABLE spark_catalog.default.transactions CREATE OR REPLACE BRANCH `test-branch`; ③
ALTER TABLE spark_catalog.default.transactions CREATE BRANCH `test-branch` AS OF VERSION 123 ④
```

① Создание ветки с политикой хранения по умолчанию.

② Создание ветки, если таковой не существует, и установка времени хранения 7 дней.

③ Создание или замена ветки, если ветка уже существует.

④ Создание ветки из снепшота `123` с использованием политики хранения по умолчанию. Снепшот `123` должен быть создан до выполнения этой команды.

Для замены и удаления веток используйте команды REPLACE BRANCH и DROP BRANCH.

ALTER TABLE … CREATE TAG

Вы можете создавать теги (tags) для "маркировки" состояния таблицы и возврата к этому состоянию таблицы в будущем. Ниже приведены несколько примеров.

```
ALTER TABLE spark_catalog.default.transactions CREATE TAG `test-tag`; ①
ALTER TABLE spark_catalog.default.transactions CREATE TAG IF NOT EXISTS `test-tag` RETAIN 365 DAYS; ②
ALTER TABLE spark_catalog.default.transactions CREATE OR REPLACE TAG `test-tag`; ③
ALTER TABLE spark_catalog.default.transactions CREATE TAG `test-tag` AS OF VERSION 123 ④
```

① Создание тега с политикой хранения по умолчанию.

② Создание тега (если такого тега не существует) из текущего снепшота, и установка времени хранения 365 дней.

③ Создание или замена тега, если такой тег уже существует.

④ Создание тега из снепшота `123` с использованием политики хранения по умолчанию. Снепшот `123` должен быть создан до выполнения этой команды.

Запрос данных из таблицы с помощью тега `test-tag` выглядит следующим образом:

```
SELECT * FROM spark_catalog.default.transactions
VERSION AS OF 'test-tag';
```

Для замены и удаления тегов используйте команды REPLACE TAG и DROP TAG.

Выборка данных

Чтобы запросить данные из таблицы Iceberg с помощью Spark SQL, необходимо указать имя каталога, пространство имен и имя таблицы.

Пример:

```
SELECT * FROM spark_catalog.default.transactions;
```

Где:

- `spark_catalog` — имя каталога по умолчанию.
- `default` — имя пространства имен по умолчанию.
- `transactions` — имя запрашиваемой таблицы.

Чтобы загрузить таблицу в виде DataFrame, используйте метод `table()`. Пример PySpark:

```
df = spark.table("spark_catalog.default.transactions")
```

Time travel

Iceberg поддерживают функцию **time travel**, которая позволяет запрашивать данные из определенного снепшота таблицы, созданного в прошлом, используя идентификатор снепшота или временную метку.

РЕКОМЕНДАЦИЯ

Информация о всех снепшотах, доступных для определенной таблицы, хранится в таблице метаданных `snapshots`.

В следующем сценарии показано, как выполнять time-travel запросы с помощью Spark.

1. Создайте тестовую таблицу с помощью *spark3-sql*:

```
CREATE TABLE spark_catalog.default.transactions_tt(
    txn_id int,
    acc_id int,
    txn_value double,
    txn_date date)
USING iceberg;
```

2. Запишите тестовые данные в таблицу:

```
INSERT INTO spark_catalog.default.transactions_tt VALUES
(1, 1002, 100.00, cast('2023-01-01' as date)),
(2, 1001, 50.00, cast('2023-01-02' as date));
```

3. Выполните запрос к таблице метаданных `snapshot`:

```
SELECT * FROM spark_catalog.default.transactions_tt.snapshots;
```

Вывод команды имеет следующий вид:

```
2024-06-27 14:54:05.664 6216997363030961663     NULL    append  hdfs://adh/apps/hive/warehouse/transactions_tt/metadata/snap-
6216997363030961663-1-f55143c4-0351-4f6a-8978-f0c2fa5666b7.avro    {"added-data-files":"2","added-files-size":"2254","added-
records":"2","changed-partition-count":"1","spark.app.id":"application_1719409181169_0020","total-data-files":"2","total-delete-
files":"0","total-equality-deletes":"0","total-files-size":"2254","total-position-deletes":"0","total-records":"2"}
```

Вывод содержит информацию о единственном доступном снепшоте (`6216997363030961663`). Данный снепшот был создан сразу после выполнения операции `INSERT`.

4. Добавьте в таблицу дополнительные тестовые данные:

```
INSERT INTO spark_catalog.default.transactions_tt VALUES
(3, 1003, 150.00, cast('2024-01-03' as date));
```

5. Выполните запрос к таблице метаданных `snapshot` еще раз:

```
SELECT * FROM spark_catalog.default.transactions_tt.snapshots;
```

Результат:

```
2024-06-27 14:54:05.664 6216997363030961663     NULL    append  hdfs://adh/apps/hive/warehouse/transactions_tt/metadata/snap-
6216997363030961663-1-f55143c4-0351-4f6a-8978-f0c2fa5666b7.avro    {"added-data-files":"2","added-files-size":"2254","added-
records":"2","changed-partition-count":"1","spark.app.id":"application_1719409181169_0020","total-data-files":"2","total-delete-
files":"0","total-equality-deletes":"0","total-files-size":"2254","total-position-deletes":"0","total-records":"2"}
2024-06-27 14:56:20.781 3775233323500475562     6216997363030961663     append
hdfs://adh/apps/hive/warehouse/transactions_tt/metadata/snap-3775233323500475562-1-6500b975-c67e-4171-8c5d-70623933abc3.avro
{"added-data-files":"1","added-files-size":"1126","added-records":"1","changed-partition-count":"1","spark.app.
id":"application_1719409181169_0020","total-data-files":"3","total-delete-files":"0","total-equality-deletes":"0","total-files-
size":"3380","total-position-deletes":"0","total-records":"3"}
```

Теперь в выводе отображается информация о еще одном снепшоте. Данный снепшот был создан второй операцией `INSERT`.

6. Выполните time-travel запрос, используя идентификатор или временную метку более старого снепшота. Для этого вы можете использовать предложения `TIMESTAMP AS OF <timestamp>` или `VERSION AS OF <id>`. Предложение `VERSION AS OF` может принимать идентификатор снепшота, имя ветки или имя тега.

```
SELECT * FROM spark_catalog.default.transactions_tt VERSION AS OF 6216997363030961663; ①
SELECT * FROM spark_catalog.default.transactions_tt TIMESTAMP AS OF '2024-06-27 14:55:20.780'; ②
```

① Выборка данных с использованием идентификатора снепшота.

② Выборка данных по временной метке. В этом примере значение временной метки подобрано таким образом, чтобы выборка осуществлялась из более старого снепшота.

Вывод:

```
1       1002    100.0   2023-01-01
2       1001    50.0    2023-01-02
```

В выводе содержатся только те данные, которые присутствовали в таблице после первой операции `INSERT`.

7. Выполните обычный запрос `SELECT`:

```
SELECT * FROM spark_catalog.default.transactions_tt;
```

Вывод:

```
1       1002    100.0   2023-01-01
2       1001    50.0    2023-01-02
3       1003    150.0   2024-01-03
```

В отличие от предыдущих запросов с использованием time-travel, вывод содержит все данные таблицы.

Таблицы метаданных

Iceberg хранит несколько таблиц с метаданными, например `history`, `files`, `snapshots` и так далее. В этих таблицах хранится ценная информация о снепшотах, файлах, операциях и т. д., которая может быть полезна для построения запросов, связанных с time-travel, ветками и тегами. Вы можете просматривать таблицы метаданных, добавляя имя таблицы метаданных после имени исходной таблицы.

Пример:

```
SELECT * FROM spark_catalog.default.transactions.snapshots;
```

Пример вывода:

```
2024-06-21 14:30:36.001 7505187482383765765     NULL    append  hdfs://adh/apps/hive/warehouse/transactions/metadata/snap-
7505187482383765765-1-be783958-0d56-4b2c-b002-54cdae7a349d.avro       {"changed-partition-count":"0","total-data-
files":"0","total-delete-files":"0","total-equality-deletes":"0","total-files-size":"0","total-position-deletes":"0","total-
records":"0"}
2024-06-21 16:09:08.645 5529064852849657419     NULL    append  hdfs://adh/apps/hive/warehouse/transactions/metadata/snap-
5529064852849657419-1-af755165-d1e2-476e-85a9-40ea179c2e9f.avro       {"added-data-files":"2","added-files-size":"2522","added-
records":"9","changed-partition-count":"1","spark.app.id":"application_1719409181169_0020","total-data-files":"2","total-delete-
files":"0","total-equality-deletes":"0","total-files-size":"2522","total-position-deletes":"0","total-records":"9"}
```

Запись данных

Вы можете записывать данные в таблицы Iceberg, используя традиционный SQL-синтаксис, например INSERT INTO, INSERT OVERWRITE, UPDATE. Кроме того, Iceberg поддерживает операции на уровне строк (row-level operations), такие как `MERGE INTO` и `DELETE FROM`, примеры использования которых показаны далее в статье. При записи в таблицу Iceberg типы данных Spark преобразуются в типы Iceberg в соответствии с таблицей преобразования типов Spark. Подробная информация о поддерживаемых операциях записи доступна в документации Iceberg.

MERGE INTO

Команда `MERGE INTO` выполняет обновление данных на уровне строк. Эта команда позволяет изменять содержимое целевой таблицы (target table), используя обновления в исходной таблице (source table). Условие обновления определяется предложением `ON`, которое используется аналогично условию `JOIN`.

Синтаксис команды `MERGE INTO` показан ниже.

```
MERGE INTO <target_table>
   USING <source_table>
   ON <merge_condition>
   WHEN MATCHED <matched_condition> THEN <matched_action> |
   WHEN NOT MATCHED THEN <not_matched_action>
```

Обновления строк в целевой таблице определяются с помощью предложения `WHEN [NOT] MATCHED <matched_condition> THEN <matched_action>`. Строки в целевой таблице можно обновлять и удалять, а те строки исходной таблицы, которые не совпадают с целевой, можно добавлять в целевую таблицу. Рассмотрим следующий пример, демонстрирующий использование `MERGE INTO`.

Предположим, что есть две таблицы Iceberg со следующим содержимым:

spark_catalog.default.transactions

```
txn_id  acc_id  txn_value txn_date
1       1001    25.0      2023-02-03
2       1001    50.0      2023-03-03
3       1002    10.0      2023-04-01
```

spark_catalog.default.transactions_copy

```
txn_id  acc_id  txn_value txn_date
1       1003    20.0      2023-02-03
3       1001    50.0      2023-03-03
5       1002    100.0     2023-04-02
```

Следующая команда `MERGE INTO` удаляет все строки в таблице `spark_catalog.default.transactions_copy`, которые имеют совпадение в таблице `spark_catalog.default.transactions` по полю `txn_id`.

```
MERGE INTO spark_catalog.default.transactions_copy USING spark_catalog.default.transactions
  ON spark_catalog.default.transactions_copy.txn_id = spark_catalog.default.transactions.txn_id
  WHEN MATCHED THEN DELETE;
```

После выполнения этой команды содержимое таблиц имеет следующий вид.

spark_catalog.default.transactions

```
txn_id  acc_id  txn_value txn_date
1       1001    25.0       2023-02-03
2       1001    50.0       2023-03-03
3       1002    10.0       2023-04-01
```

spark_catalog.default.transactions_copy

```
txn_id  acc_id  txn_value txn_date
5       1002    100.0     2023-04-02
```

Следующая команда `MERGE INTO` обновляет все строки в таблице `spark_catalog.default.transactions`, если поле `acc_id` совпадает с `acc_id` таблицы `spark_catalog.default.transactions_copy`. Для строк с совпадающими значениями `acc_id` команда устанавливает значение `txn_value=0`.

```
MERGE INTO spark_catalog.default.transactions USING spark_catalog.default.transactions_copy
  ON spark_catalog.default.transactions.acc_id = spark_catalog.default.transactions_copy.acc_id
  WHEN MATCHED THEN UPDATE SET txn_value=0;
```

После выполнения этой команды содержимое таблиц имеет следующий вид.

spark_catalog.default.transactions

```
txn_id  acc_id  txn_value txn_date
1       1001    25.0      2023-02-03
2       1001    50.0      2023-03-03
3       1002    10.0      2023-04-01
```

spark_catalog.default.transactions_copy

```
txn_id  acc_id  txn_value txn_date
5       1002    0.0       2023-04-02
```

DELETE FROM

Операция `DELETE FROM` позволяет удалять данные из таблиц Iceberg на уровне строк. В запросах на удаление указывается фильтр для поиска удаляемых строк, как показано ниже. Если фильтр удаления соответствует всему разделу таблицы, Iceberg изменит только метаданные таблицы. Если фильтр соответствует отдельным строкам, Iceberg перезапишет только релевантные файлы данных.

Примеры:

```
DELETE FROM spark_catalog.default.transactions
WHERE txn_date > '2023-03-03';
```

```
DELETE FROM spark_catalog.default.transactions
WHERE txn_value < (SELECT min(txn_value) FROM spark_catalog.default.transactions_copy);
```

Запись данных в ветку

Чтобы записать данные в определенную ветку (branch) Iceberg-таблицы, необходимо указать идентификатор ветки в запросе.

ПРИМЕЧАНИЕ

Перед выполнением записи целевая ветка должна быть создана.

Например, чтобы записать данные в ветку с именем `testbranch`, используйте следующий синтаксис:

```
INSERT INTO spark_catalog.default.transactions.branch_testbranch VALUES
(1, 1003, 20.00, cast('2023-02-03' as date));
```

Если имя ветки содержит не только алфавитно-цифровые символы (например, `test-branch`), имя ветки должно быть заключено в обратные кавычки, как показано в примере ниже.

```
INSERT INTO spark_catalog.default.transactions.`branch_test-branch` VALUES
(1, 1003, 20.00, cast('2023-02-03' as date));
```

Чтобы получить данные из созданной выше ветки `testbranch`, используйте команду:

```
SELECT * FROM spark_catalog.default.transactions
VERSION AS OF 'testbranch';
```

Помимо записи данных в ветку с помощью `INSERT`, вы также можете использовать команды `UPDATE`, `DELETE` и `MERGE INTO`.

Примеры:

```
UPDATE spark_catalog.default.transactions.branch_testbranch
SET acc_id = 1005
WHERE acc_id = 1003
```

```
DELETE FROM spark_catalog.default.transactions.branch_testbranch
WHERE txn_id = 1;
```

```
MERGE INTO spark_catalog.default.transactions.branch_testbranch USING spark_catalog.default.transactions_copy
  ON spark_catalog.default.transactions.branch_testbranch.txn_id = spark_catalog.default.transactions_copy.txn_id
  WHEN MATCHED THEN DELETE;
```

Roll back

Iceberg предоставляет функцию rollback, которая позволяет откатить таблицу к более раннему состоянию. Данная функция может быть полезна, например, для отмены случайных или ошибочных изменений в таблицах. Данные таблицы можно откатить до нужного состояния, пока целевой снепшот доступен в метаданных Iceberg. Следующий пример демонстрирует базовое использование rollback-функции.

1. Создайте тестовую таблицу Iceberg со следующим содержимым:

```
txn_id  acc_id  txn_value txn_date
1       1002    10.0      2024-01-01
2       1001    20.0      2024-01-02
```

Для этого выполните запросы в *spark3-sql*:

```
CREATE TABLE spark_catalog.default.transactions_rollback_demo (
    txn_id int,
    acc_id int,
    txn_value double,
    txn_date date)
USING iceberg;
```

```
INSERT INTO spark_catalog.default.transactions_rollback_demo VALUES
(1, 1002, 10.00, cast('2024-01-01' as timestamp)),
(2, 1001, 20.00, cast('2024-01-02' as timestamp));
```

2. Удалите одну строку из тестовой таблицы. На следующих шагах будет выполнен откат данной операции.

```
DELETE FROM spark_catalog.default.transactions_rollback_demo
WHERE txn_id = 2;
```

3. Запросите информацию о снепшотах таблицы. Для этого выполните запрос к базе метаданных, как показано ниже:

```
SELECT * FROM spark_catalog.default.transactions_rollback_demo.snapshots;
```

Пример вывода:

```
2024-07-17 07:59:56.31  6142059163774284478     NULL    append
hdfs://adh/apps/hive/warehouse/transactions_rollback_demo/metadata/snap-6142059163774284478-1-c2c3b442-648b-4e94-8b14-
594ec65f66c5.avro{"added-data-files":"2","added-files-size":"2254","added-records":"2","changed-partition-count":"1","spark.app.
id":"application_1720628013974_0014","total-data-files":"2","total-delete-files":"0","total-equality-deletes":"0","total-files-
size":"2254","total-position-deletes":"0","total-records":"2"}
2024-07-17 08:02:55.859 8329113954968361656     6142059163774284478     delete
hdfs://adh/apps/hive/warehouse/transactions_rollback_demo/metadata/snap-8329113954968361656-1-375044c0-90fa-44f5-a6c6-
72ebc12cc473.avro {"changed-partition-count":"1","deleted-data-files":"1","deleted-records":"1","removed-files-
size":"1127","spark.app.id":"application_1720628013974_0015","total-data-files":"1","total-delete-files":"0","total-equality-
deletes":"0","total-files-size":"1127","total-position-deletes":"0","total-records":"1"}
```

Вывод показывает, что для тестовой таблицы доступны два снепшота. Первый был создан сразу после вставки данных, а второй — после операции удаления.

4. Откатите содержимое таблицы до состояния, которое предшествовало операции удаления:

```
CALL spark_catalog.system.rollback_to_snapshot('spark_catalog.default.transactions_rollback_demo', <snapshot-id>);
```

Где `<snapshot-id>` — ID снепшота, который был создан до выполнения операции DELETE.

РЕКОМЕНДАЦИЯ

Вместо ID можно также указать временную метку для отката к нужному снепшоту.

5. Запросите данные из таблицы, чтобы проверить результат выполнения rollback-функции.

```
SELECT * FROM spark_catalog.default.transactions_rollback_demo;
1       1002    10.0    2023-01-01
2       1001    20.0    2023-01-02
```

Вывод показывает, что удаленная строка была восстановлена.

Управление сервисом Spark3 через ADCM

Константин Алпашкин

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Spark3. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Spark3, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Spark3
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Manage Ranger plugin	Включает или отключает плагин Ranger Spark3
Switch dynamic allocation	Включает/отключает динамическое выделение ресурсов для Spark3

Действия с компонентами Spark3

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Обзор SSM UI

Константин Алпашкин, Елена Костюченко

Содержание

Назад к содержанию

Cluster Info

Rules

Actions

Audit

SSM предоставляет веб-интерфейс (AD.SSM), где вы можете управлять правилами SSM (rules), выполнять SSM-действия (actions), отслеживать состояние хранилища, просматривать логи действий и так далее.

По умолчанию веб-интерфейс доступен по адресу *http://<host>:7045/*, где `<host>` — это хост ADH с установленным компонентом SSM Server. Вы можете изменить номер порта (`7045` по умолчанию) с помощью параметра конфигурации `smart.server.http.address`.

SSM UI включает следующие основные разделы:

- Cluster Info
- Rules
- Actions
- Audit

В нижней части левой панели доступны опции:

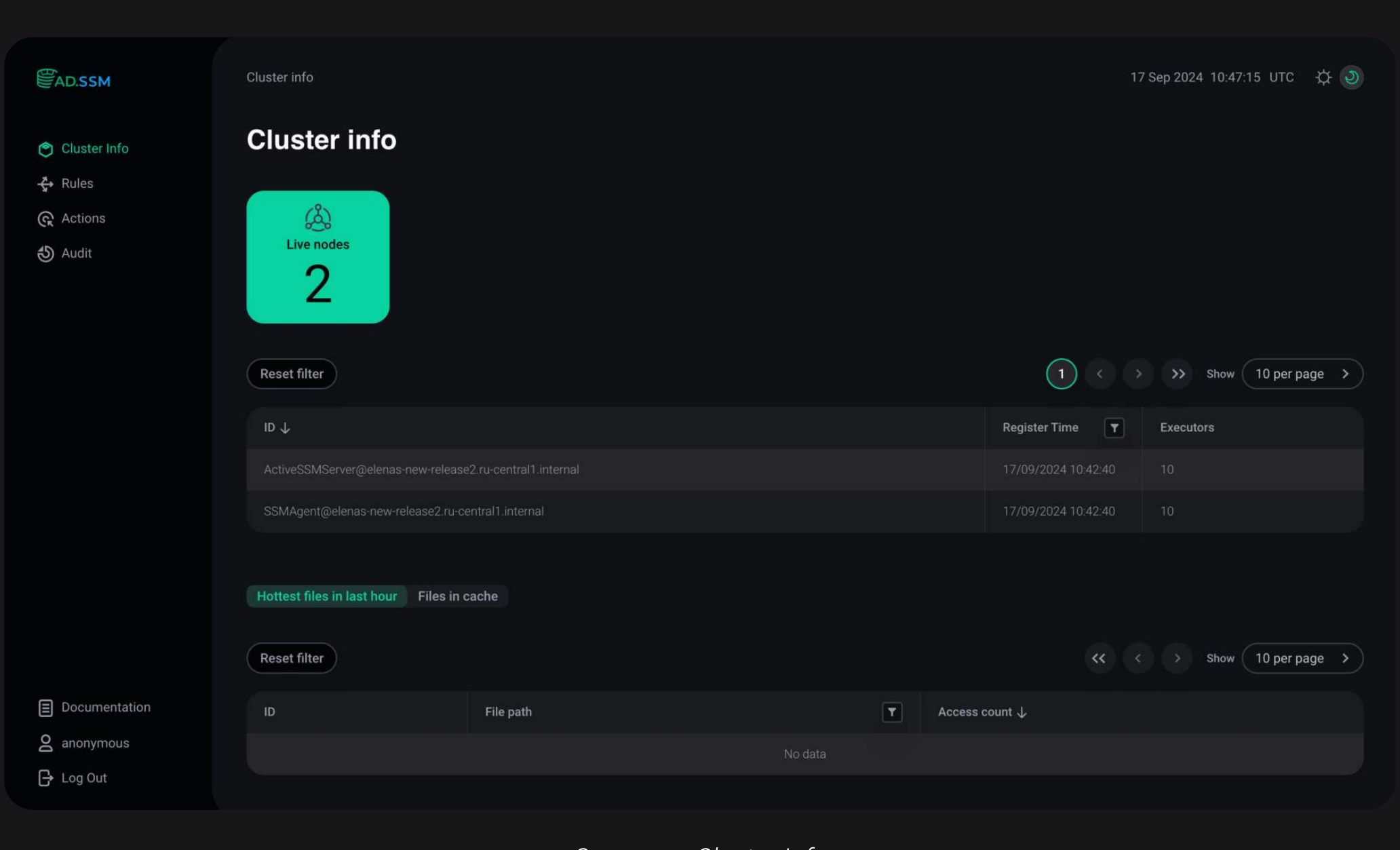
- 📄 ***Documentation*** — ссылка на документацию SSM.
- 👤 — индикатор профиля пользователя, который показывает имя пользователя, под которым выполнен вход в UI.
- ⇥ ***Log Out*** — опция для завершения сеанса текущего пользователя и возвращения к экрану входа.

Cluster Info

На этой странице отображаются три таблицы:

- Таблица с информацией об активности компонентов SSM Server/SSM Agent.
- Таблицы статистики по файлам отображают информацию о всех файлах и файлах в кеше:
  - ***Hottest files in last hour*** — отображает самые популярные (часто используемые) файлы за последний час со счетчиком `accessCount`.
  - ***Files in cache*** — отображает информацию о кешированных файлах.

Чтобы посмотреть одну из таблиц статистики файлов, выберите опцию ***Hottest files in last hour*** (самые популярные файлы за последний час) или ***Files in cache*** (файлы в кеше) над таблицей.



ID	Register Time	Executors
ActiveSSMServer@elenas-new-release2.ru-central1.internal	17/09/2024 10:42:40	10
SSMAgent@elenas-new-release2.ru-central1.internal	17/09/2024 10:42:40	10

ID	File path	Access count
No data		

Страница Cluster Info

Таблица Nodes

Поле	Описание
ID	Идентификатор ноды в формате `<component>@<host>`. Здесь: - `<component>` — имя компонента, установленного на хосте; - `<host>` — полное доменное имя хоста (FQDN).
Register Time	Время запуска компонента в формате `DD/MM/YYYY HH:mm:ss`
Executors	Количество исполнителей (executors) на ноду

Нажмите на заголовок столбца, чтобы сортировать строки таблицы. Столбец ***Register Time*** поддерживает фильтр, который можно использовать для выбора определенного промежутка времени. Для использования фильтра нажмите ▽.

Чтобы сбросить все фильтры, нажмите ***Reset***.

Таблица Hottest files in last hour

Поле	Описание
ID	Числовой идентификатор файла
File path	Путь к файлу
Access count	Количество запросов к файлу

Нажмите на заголовок столбца, чтобы сортировать строки таблицы. Столбец ***File path*** поддерживает фильтр, который можно использовать для поиска определенного пути. Для использования фильтра нажмите ▽.

Чтобы сбросить все фильтры, нажмите ***Reset***.

Таблица Files in cache

Поле	Описание
ID	Числовой идентификатор файла
File path	Путь к файлу
Cached Time	Время, когда файл был кеширован
Last Accessed Time	Время последнего обращения к файлу
Access count	Количество запросов файла

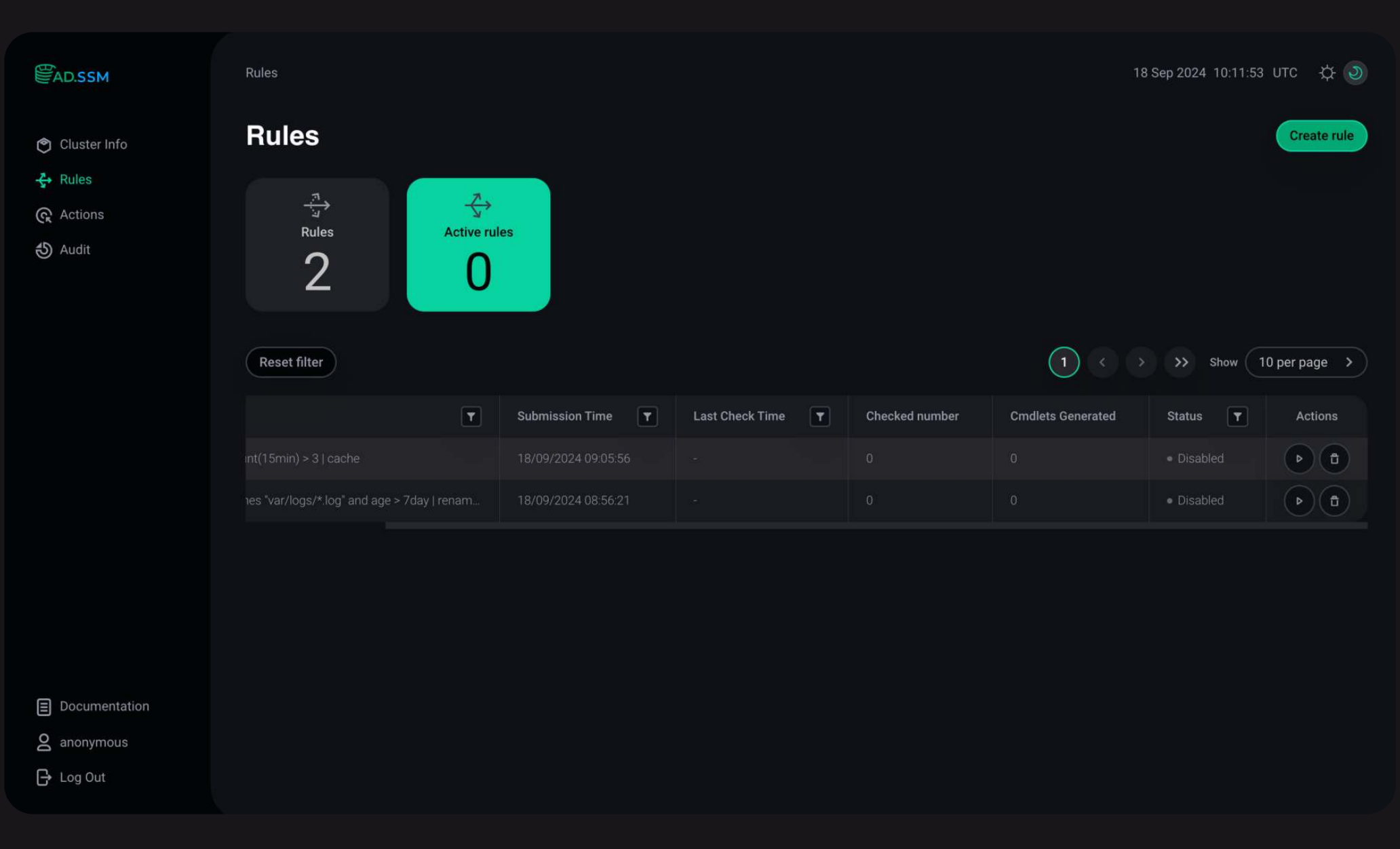
Нажмите на заголовок столбца, чтобы сортировать строки таблицы. Некоторые столбцы поддерживают фильтры, доступ к которым можно получить нажав на ▽:

- ***File path*** — предоставляет возможность поиска по определенному пути к файлу;
- ***Cached Time*** — позволяет фильтровать по времени кеширования файла;
- ***Last Accessed Time*** — позволяет выбрать дату последнего кеширования файла.

Чтобы сбросить все фильтры, нажмите ***Reset***.

Rules

На этой странице вы можете увидеть все добавленные правила, изменить их статус и просмотреть детали выполнения. Виджеты в верхней части страницы отображают общее количество существующих правил и количество активных правил. В таблице правил можно увидеть информацию о работе правил.



	Submission Time	Last Check Time	Checked number	Cmdlets Generated	Status	Actions
nt(15min) > 3 | cache	18/09/2024 09:05:56	-	0	0	Disabled	▷ 🗑
es "var/logs/*.log" and age > 7day | renam...	18/09/2024 08:56:21	-	0	0	Disabled	▷ 🗑

Страница Rules

Таблица Rules

Поле	Описание
ID	Числовой идентификатор правила
Rule Text	Код правила
Submission Time	Время создания правила в формате `DD/MM/YYYY HH:mm:ss`
Last Check Time	Последний раз, когда правило проверяло условие в формате `DD/MM/YYYY HH:mm:ss`
Checked number	Количество раз, когда правило проверяло условие
Cmdlets Generated	Количество сгенерированных командлетов (cmdlets)
Status	Индикатор активности правила. Может быть `Disabled` (отключено) или `Active` (активно)
Actions	Действия, доступные для правила: - ▷ — активировать правило; - ⏸ — приостановить правило; - 🗑 — удалить правило.

Нажмите на заголовок столбца, чтобы сортировать строки таблицы. Некоторые столбцы поддерживают фильтры, доступ к которым можно получить нажав на ▽:

- ***Rule Text*** — обеспечивает возможность поиска правил с определенными словами или символами;
- ***Submission Time*** — позволяет фильтровать по времени создания правила;
- ***Last Check Time*** — позволяет выбрать дату последней проверки условий правилом;
- ***Status*** — позволяет фильтровать по статусу правила.

Чтобы сбросить все фильтры, нажмите ***Reset***.

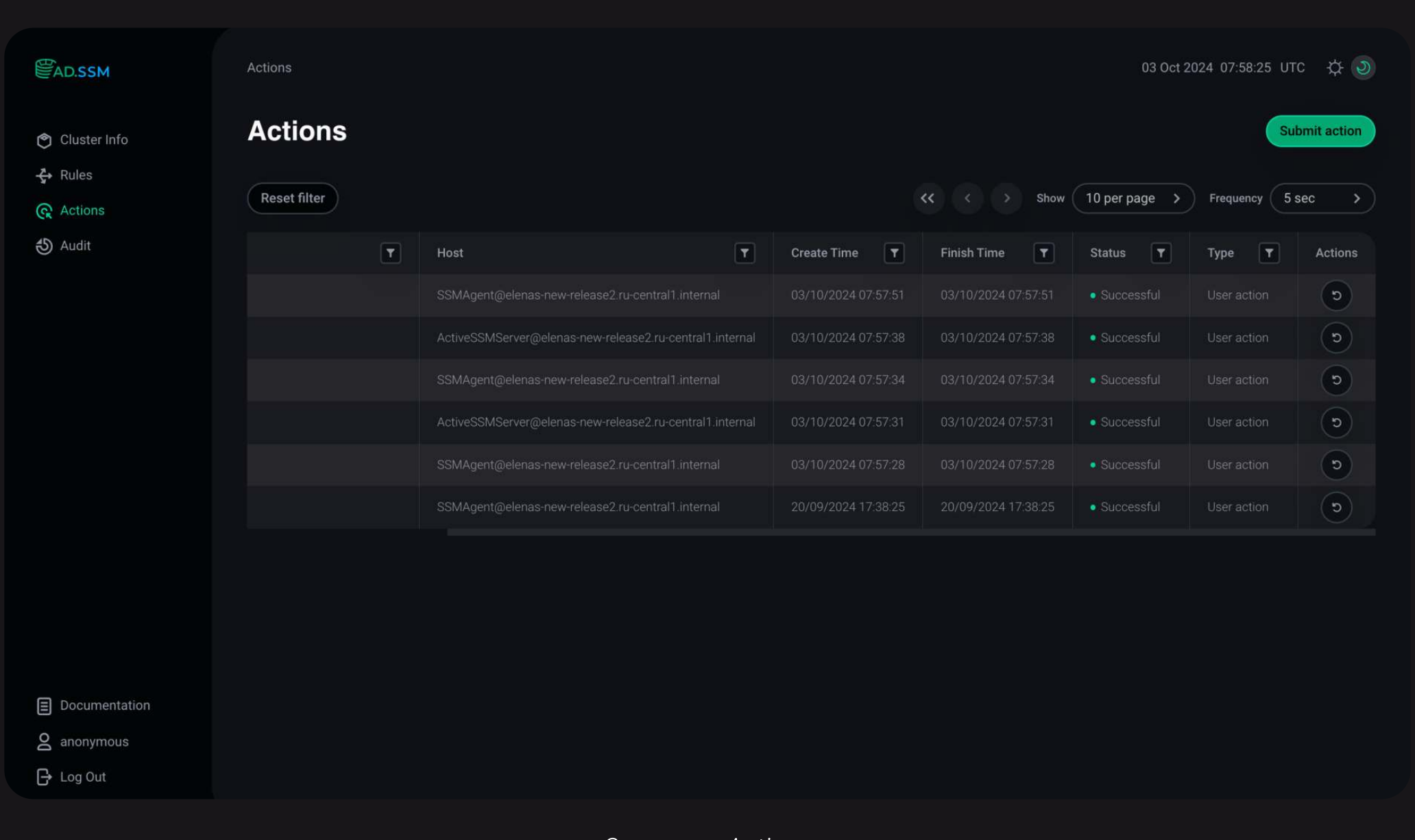
Чтобы создать правило:

1. В интерфейсе AD.SSM перейдите в раздел ***Rules***.
2. Выберите опцию ***Create rule***.
3. В появившемся окне опишите нужное правило и нажмите ***Create***.

Вновь добавленные правила отображаются на этой странице со статусом `DISABLED`. Это означает, что даже если условие правила выполнено, SSM не предпримет никаких действий по перемещению данных. Чтобы правило начало действовать, необходимо активировать его, кликнув ▷. И наоборот, для деактивации правила используйте ⏸. Чтобы удалить правило, выберите опцию 🗑.

Actions

На этой странице перечислены все действия, выполненные в результате срабатывания правил либо запущенные вручную.



	Host	Create Time	Finish Time	Status	Type	Actions
	SSMAgent@elenas-new-release2.ru-central1.internal	03/10/2024 07:57:51	03/10/2024 07:57:51	Successful	User action	↺
	ActiveSSMServer@elenas-new-release2.ru-central1.internal	03/10/2024 07:57:38	03/10/2024 07:57:38	Successful	User action	↺
	SSMAgent@elenas-new-release2.ru-central1.internal	03/10/2024 07:57:34	03/10/2024 07:57:34	Successful	User action	↺
	ActiveSSMServer@elenas-new-release2.ru-central1.internal	03/10/2024 07:57:31	03/10/2024 07:57:31	Successful	User action	↺
	SSMAgent@elenas-new-release2.ru-central1.internal	03/10/2024 07:57:28	03/10/2024 07:57:28	Successful	User action	↺
	SSMAgent@elenas-new-release2.ru-central1.internal	20/09/2024 17:38:25	20/09/2024 17:38:25	Successful	User action	↺

Страница Actions

Таблица Actions

Поле	Описание
ID	Числовой идентификатор действия
Action	Код действия
Host	Нода, запустившая действие
Create Time	Время создания действия в формате `DD/MM/YYYY HH:mm:ss`
Finish Time	Время завершения действия в формате `DD/MM/YYYY HH:mm:ss`
Status	Статус выполнения действия. Может быть одним из следующих: - `Running` — выполняется; - `Successful` — успешно; - `Failed` — завершилось с ошибкой.
Type	Тип действия в зависимости от его триггера. Действие может быть вызвано либо правилом (`Rule action`), либо пользователем (`User action`)
Actions	Возможности управления действиями. В настоящее время доступна только опция повторного выполнения действия ↺ ***Repeat action***

Нажмите на заголовок столбца, чтобы сортировать строки таблицы. Некоторые столбцы поддерживают фильтры, доступ к которым можно получить нажав на ▽:

- ***Action*** — обеспечивает возможность поиска действий с определенными словами или символами;
- ***Host*** — позволяет отображать действия, которые были выполнены определенной нодой;
- ***Create Time*** — позволяет фильтровать по времени создания действия;
- ***Finish Time*** — позволяет фильтровать по времени завершения действия;
- ***Status*** — отображает действия с определенным статусом выполнения;
- ***Type*** — отображает действия определенного типа.

Чтобы сбросить все фильтры, нажмите ***Reset***.

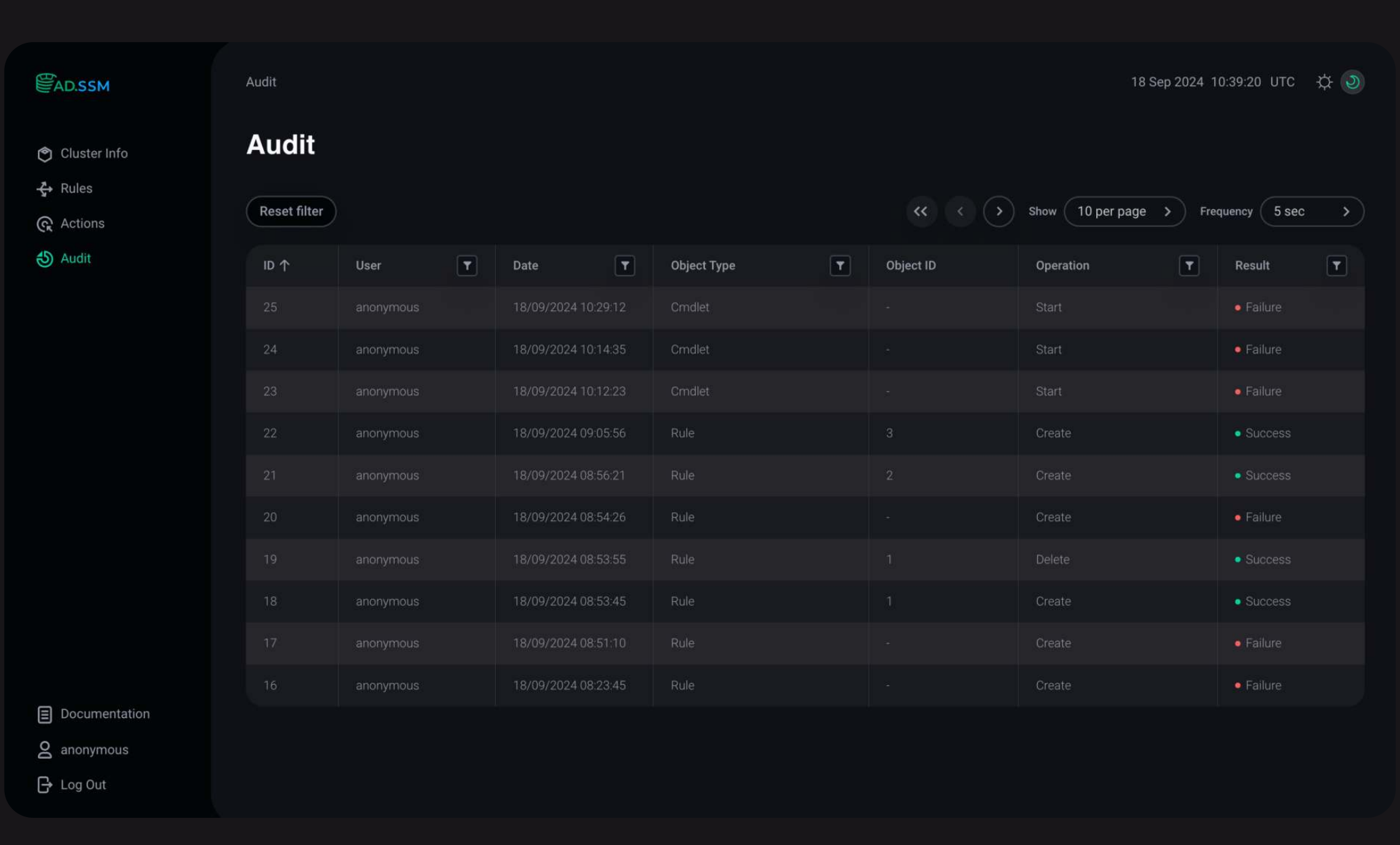
Чтобы запустить действие:

1. В интерфейсе AD.SSM перейдите в раздел ***Actions***.
2. Выберите опцию ***Submit action***.
3. В появившемся окне опишите желаемое действие и нажмите ***Run***.

Действие выполняется немедленно. Если вы хотите повторить действие, нажмите ↺.

Audit

На этой странице отображается таблица с информацией о действиях пользователя в AD.SSM.



ID	User	Date	Object Type	Object ID	Operation	Result
25	anonymous	18/09/2024 10:29:12	Cmdlet	-	Start	Failure
24	anonymous	18/09/2024 10:14:35	Cmdlet	-	Start	Failure
23	anonymous	18/09/2024 10:12:23	Cmdlet	-	Start	Failure
22	anonymous	18/09/2024 09:05:56	Rule	3	Create	Success
21	anonymous	18/09/2024 08:56:21	Rule	2	Create	Success
20	anonymous	18/09/2024 08:54:26	Rule	-	Create	Failure
19	anonymous	18/09/2024 08:53:55	Rule	1	Delete	Success
18	anonymous	18/09/2024 08:53:45	Rule	1	Create	Success
17	anonymous	18/09/2024 08:51:10	Rule	-	Create	Failure
16	anonymous	18/09/2024 08:23:45	Rule	-	Create	Failure

Страница Audit

Таблица Audit

Поле	Описание
ID	Числовой идентификатор события
User	Имя пользователя, выполнившего действие
Date	Дата и время события в формате `DD/MM/YYYY HH:mm:ss`
Object Type	Тип объекта, выполнившего действие: правило или командлет (cmdlets)
Object ID	Идентификатор объекта
Operation	Операция, выполненная пользователем
Result	Результат операции: `success` (успех) или `failure` (неудача)

Нажмите на заголовок столбца, чтобы сортировать строки таблицы. Некоторые столбцы поддерживают фильтры, доступ к которым можно получить нажав на ▽:

- ***User*** — предоставляет возможность поиска имен пользователей с определенными словами или символами;
- ***Date*** — позволяет фильтровать по времени события;
- ***Object Type*** — отображает события с определенным типом объекта;
- ***Operation*** — позволяет фильтровать по определенному типу операции: `Create`, `Delete`, `Start`, `Stop`;
- ***Result*** — отображает события с определенным результатом.

Чтобы сбросить все фильтры, нажмите ***Reset***.

Определение правил в SSM

Сергей Тихомиров

Содержание

Назад к содержанию

Обзор
Добавление правил

Обзор

Правилом в SSM называется конструкция, определяющая какое действие нужно выполнить с данными после выполнения определенного условия. Использование таких правил дает возможность гибко изменять кластер под определенные нужды. Так, с помощью правил SSM можно перемещать файлы логов, которым более 7 дней, в определенное место и переименовывать их, или помещать файл в кеш, если он запрашивается относительно часто.

Правило состоит из четырех частей:

- **Объекты**. Файлы, к которым применится команда в случае выполнения условия.
- **Триггер**. Опциональный параметр, позволяющий установить время, в которое произойдет проверка условия. Можно указать точное время или интервал обновления.
- **Условия**. Выражение, состоящее из значений свойств объектов и операторов, которое проверяется на выполнимость.
- **Команды**. Действия, запускаемые в случае успешной проверки условий.

Шаблон правила представлен ниже:

```
<объекты>: <триггер> | <условия> | <команды>
```

ПРИМЕЧАНИЕ

Символы `:` и `|` зарезервированы SSM в качестве разделителей и не могут использоваться внутри частей правил.

Примеры вышеупомянутых правил представлены ниже.

```
file: path matches "var/logs/*.log" and age > 7day | rename -dest "var/old_logs/"
file: accessCount(15min) > 3 | cache
```

Правило может находиться в четырех состояниях:

- **Active**. Когда правило активно, SSM проверяет условие в зависимости от триггера и выполняет команды при успешной проверке.
- **Disabled**. Состояние правила после создания. Активное правило можно поставить на паузу — тогда оно получит данный статус, а в последующем правило можно снова активировать. Если при установке правила на паузу в очереди остались невыполненные действия, то при реактивации правила пользователь может продолжить их выполнение либо начать выполнение заново. По умолчанию незавершенные операции отменятся.
- **Finished**. Если у правила был временной триггер, который уже выполнился и никогда не выполнится снова, то оно получит данный статус.
- **Deleted**. Состояние правила после того, как пользователь его удалил, но в очереди остались действия, связанные с этим правилом. После выполнения этих действий правило будет безвозвратно удалено.

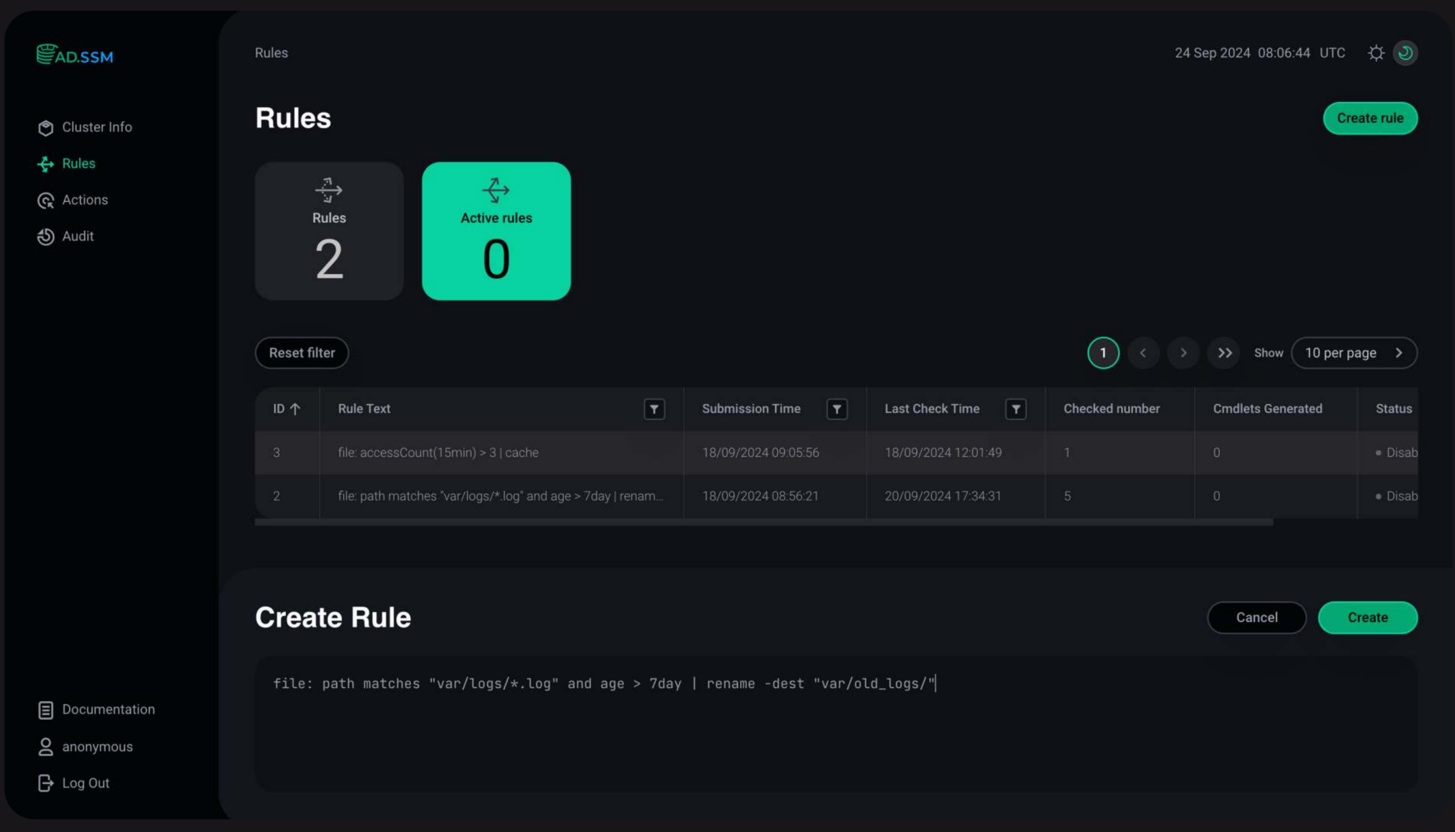
Более подробная информация о синтаксисе правил SSM доступна в статье Правила в SSM.

Добавление правил

Добавить правило можно в веб-интерфейсе AD.SSM. Чтобы перейти в него, следуйте шагам ниже.

1. На странице ***Clusters*** найдите ваш кластер ADH с установленным SSM и нажмите на него.
2. На вкладке ***Services*** нажмите ***SSM*** и перейдите на вкладку ***Info*** сервисного меню.
3. Перейдите по ссылке, ведущей в веб-интерфейс.

В AD.SSM UI правила можно добавить на странице ***Rules***.



ID	Rule Text	Submission Time	Last Check Time	Checked number	Cmdlets Generated	Status
3	file: accessCount(15min) > 3 | cache	18/09/2024 09:05:56	18/09/2024 12:01:49	1	0	• Disab
2	file: path matches "var/logs/*.log" and age > 7day | renam...	18/09/2024 08:56:21	20/09/2024 17:34:31	5	0	• Disab

```
file: path matches "var/logs/*.log" and age > 7day | rename -dest "var/old_logs/"
```

Страница Rules

Чтобы создать правило:

1. В интерфейсе AD.SSM перейдите в раздел ***Rules***.
2. Выберите опцию ***Create rule***.
3. В появившейся панели опишите нужное правило и нажмите ***Create***.

Созданное правило будет иметь статус `Disabled` (не активно). Чтобы активировать его, нажмите ▷. Статус изменится на `Active` (активно).

Определение действий в SSM

Сергей Тихомиров

Содержание

Назад к содержанию

Обзор

Запуск действия

Пример использования действия

Обзор

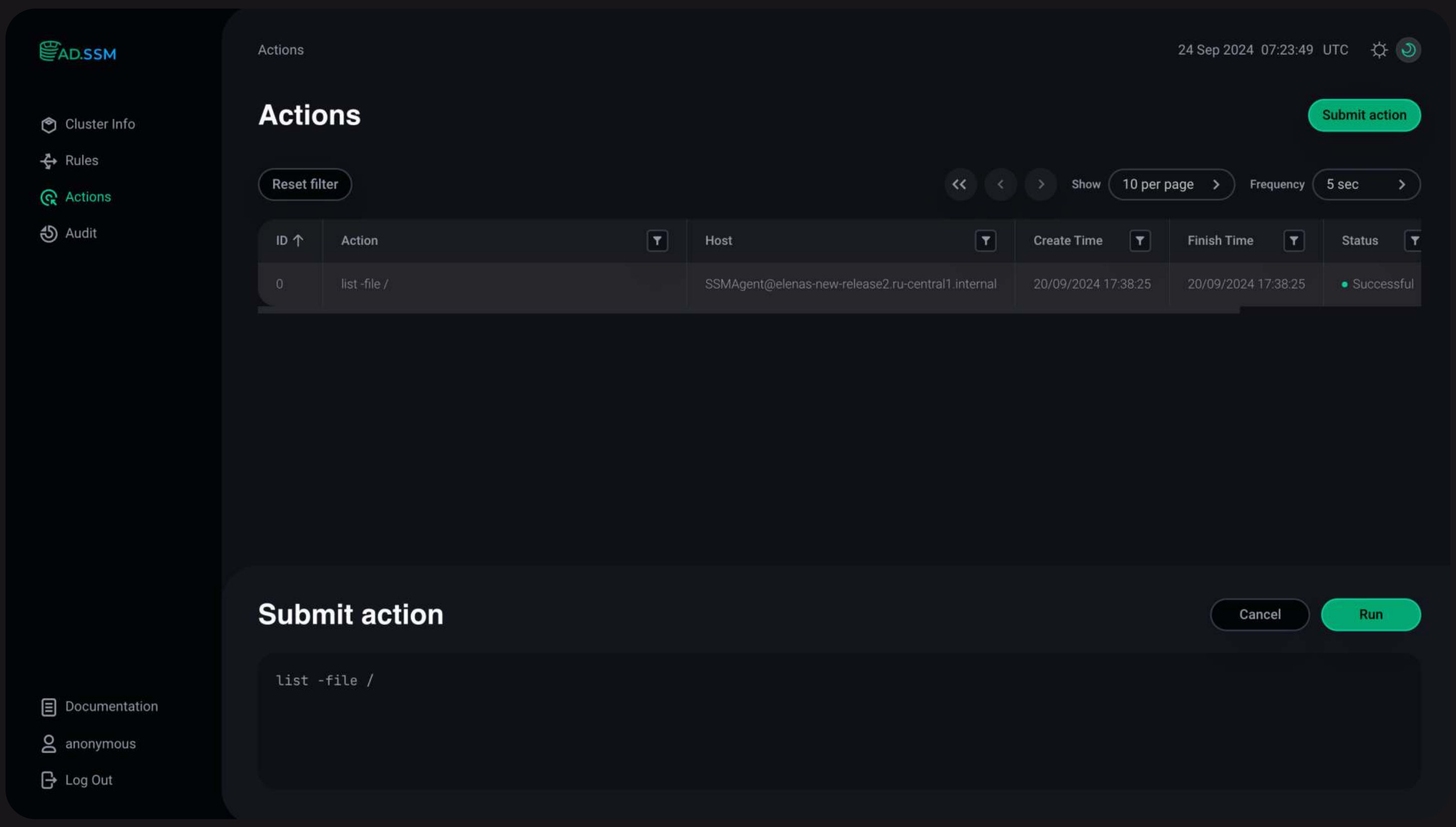
Действиями в SSM называются команды, выполняемые с данными при соблюдении условий, описанных в правилах. Вы можете запустить действие в любой момент с помощью специальной консоли в UI AD.SSM.

Полный список действий, доступных в SSM, можно найти в статье Действия в SSM.

Запуск действия

Чтобы запустить действие, следуйте шагам ниже:

1. В веб-интерфейсе AD.SSM перейдите на страницу ***Actions***.
2. Выберите опцию ***Submit action***.
3. Введите команду в панели ***Submit action*** и нажмите ***Run***.


AD.SSM
Cluster Info
Rules
Actions
Audit
Documentation
anonymous
Log Out
Actions
24 Sep 2024 07:23:49 UTC
Actions
Submit action
Reset filter
Show
10 per page
Frequency
5 sec
Submit action
Cancel
Run

ID	Action	Host	Create Time	Finish Time	Status
0	list -file /	SSMAgent@elenas-new-release2.ru-central1.internal	20/09/2024 17:38:25	20/09/2024 17:38:25	Successful

```
list -file /
```

Страница Actions с панелью Submit action

ПРИМЕЧАНИЕ

Для навигации используйте команду ниже, которая выводит содержимое HDFS-директории:

```
list -file /
```

Пример использования действия

Ниже вы можете увидеть, как определить действие, копирующее файлы в хранилище S3.

После настройки аутентификации для S3 вы можете копировать данные в хранилище S3, используя следующее действие:

```
copy2s3 -file /user/sergei/demoCopyS3/f1.txt -dest s3a://develop-test-bucket/f1.txt
```

Чтобы проверить, скопировался ли файл, выполните следующую команду:

```
$ hdfs dfs -cat s3a://develop-test-bucket/f1.txt
```

Если копирование прошло без ошибок, вы должны увидеть содержимое файла.

Примеры использования правил в SSM

Сергей Тихомиров

Содержание

Назад к содержанию

Обзор

Кеширование файлов

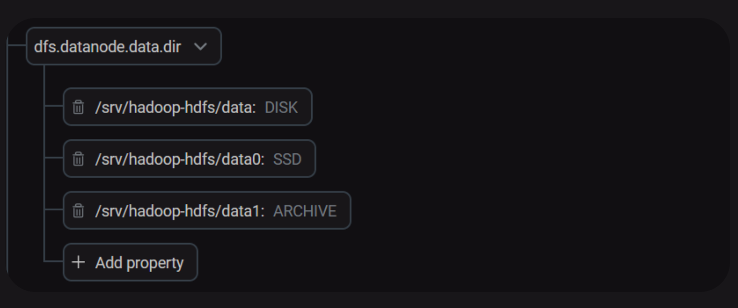
Перемещение данных между холодным хранилищем и горячим

Синхронизация данных

Обзор

В данной статье описаны примеры использования правил в SSM. Перед запуском примеров выполните следующие шаги:

1. Установите SSM.
2. Настройте типы хранения SSD, DISK и ARCHIVE в параметре ***hdfs-site.xml → dfs.datanode.data.dir*** в ADCM.



dfs.datanode.data.dir

/srv/hadoop-hdfs/data: DISK

/srv/hadoop-hdfs/data0: SSD

/srv/hadoop-hdfs/data1: ARCHIVE

Типы хранилищ

Кеширование файлов

Если вы хотите автоматически перемещать часто запрашиваемые файлы в кеш, используйте правило ниже:

```
file: accessCount(1min) > 0 and path matches "/demoCacheFile/*" | cache
```

С таким правилом SSM будет кешировать файлы из директории *demoCacheFile*, которые запрашивались хотя бы 1 раз за последнюю минуту.

Для тестирования заранее были созданы файлы в директории *user/sergei/demoCacheFile* в HDFS. Команда ниже печатает имена этих файлов:

```
list -file /user/sergei/demoCacheFile
```

```
Action starts at Wed Feb 14 14:05:59 UTC 2024 : List /user/sergei/demoCacheFile
-rw-r--r--     3 sergei sergei              10 2024-02-14 14:04 /user/sergei/demoCacheFile/f1.txt
-rw-r--r--     3 sergei sergei              13 2024-02-14 14:04 /user/sergei/demoCacheFile/f2.txt
-rw-r--r--     3 sergei sergei              12 2024-02-14 14:04 /user/sergei/demoCacheFile/f3.txt
```

Чтобы проверить работоспособность примера, запросите файл, например, выполнив действие `read`:

```
read -file /user/sergei/demoCacheFile/f1.txt
```

Если ошибок нет, то вы можете увидеть имя этого файла на странице ***Cluster Info → Files in cache***.

Перемещение данных между холодным хранилищем и горячим

> **ВАЖНО**
>
> Убедитесь, что в HDFS настроены типы хранения SSD и ARCHIVE, иначе пример может работать некорректно.

Чтобы переместить файлы из холодного хранилища в горячее, используйте правило ниже:

```
file: accessCount(1min) > 0 and path matches "/user/sergei/demoCold2Hot/*" | allssd
```

Чтобы проверить тип хранилища, выполните команду `checkstorage` в панели ***Submit action***:

```
checkstorage -file /user/sergei/demoCold2Hot/f1.txt
```

До применения правила вывод команды следующий:

```
File offset = 0, Block locations = {10.92.41.28:9866[DISK] 10.92.41.202:9866[DISK] 10.92.41.33:9866[DISK] }
```

После прочтения файла условие в правиле выполнилось, и действие `allssd` применилось. Вывод стал таким:

```
File offset = 0, Block locations = {10.92.41.33:9866[SSD] 10.92.41.28:9866[SSD] 10.92.41.202:9866[SSD] }
```

Также файл теперь можно найти на странице ***Cluster Info → Hottest files in last hour***.

Чтобы настроить перемещение файлов из горячего хранилища в холодное, используйте следующее правило:

```
file : age > 3min and path matches "/user/sergei/demoHot2Cold/*" | archive
```

Синхронизация данных

Для синхронизации файлов в директории с другим кластером с HDFS используйте следующее правило:

```
file: path matches "/user/sergei/demoSyncSrc/*" | sync -dest hdfs://stikhomirov-adh1.ru-central1.internal/demoSyncDest
```

Если на кластере с SSM настроено специальное пространство имен, вы можете подставить название пространства имен вместо имени хоста в команде выше.

> **ПРИМЕЧАНИЕ**
>
> Для корректной работы команды `sync` достаточно иметь установленный SSM хотя бы на одном кластере (источник или целевой). Также источник копирования не обязан быть локальным, а целевая директория — удаленной.

Чтобы проверить правильность синхронизации, можно сравнить контрольные суммы. Для источника можно получить контрольную сумму, выполнив соответствующую команду в панели ***Submit action***:

```
checksum -file /user/sergei/demoSyncSrc/f1.txt
```

Вывод для данного файла:

```
/user/sergei/demoSyncSrc/f1.txt MD5-of-0MD5-of-512CRC32C        000002000000000000000000049c19b14a1077d0a6fc95bee2db8914c
```

Для получения контрольной суммы на целевом сервере выполните на нем следующую команду:

```
$ hadoop fs -checksum /demoSyncDest/f1.txt
```

Контрольная сумма в выводе совпадает с приведенной выше:

```
/demoSyncDest/f1.txt    MD5-of-0MD5-of-512CRC32C        000002000000000000000000049c19b14a1077d0a6fc95bee2db8914c
```

Управление сервисом SSM через ADCM

Константин Алпашкин

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом SSM. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса SSM, перечислены в таблице ниже.

Действия с сервисом SSM

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты SSM
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются

Действия, доступные для компонентов сервиса SSM, перечислены в таблице ниже.

Действия с компонентами SSM

Действие	Описание
Add/Remove Agents	Добавляет/удаляет компонент(ы) SSM Agent. При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете распределить компоненты SSM Agent между хостами
Add/Remove Servers	Добавляет/удаляет компонент(ы) SSM Server. При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете распределить компоненты SSM Server между хостами
Check	Выполняет проверку работоспособности компонента
Restart	Перезапускает компонент. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт компонента выполняется с учетом всех параметров, указанных в настройках этого компонента в ADCM. В противном случае настройки компонента в ADCM игнорируются
Start	Запускает компонент. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск компонента выполняется с учетом всех параметров, указанных в настройках этого компонента в ADCM. В противном случае настройки компонента в ADCM игнорируются
Stop	Останавливает компонент

Подключение к Trino через CLI

Владимир Адаменков, Константин Алпашкин

Содержание

Назад к содержанию

- Компонент Trino Client
  - Использование
  - Выполнение выражений
  - Настройка
  - TLS/HTTPS
- Удаленное подключение

Компонент Trino Client

Сервис Trino содержит компонент Trino Client, который предоставляет интерфейс командной строки (CLI) для отправки SQL-запросов в Trino Coordinator. При установке компонента Trino Client он автоматически конфигурируется для работы с Trino Coordinator, доступным в кластере ADH.

Использование

Чтобы подключиться к Trino через CLI, на хосте ADH с установленным компонентом Trino Client выполните команду:

```
$ trino
```

Если подключение выполнено успешно — появится приглашение командной строки Trino:

```
trino>
```

Для просмотра доступных команд выполните команду `help`. Пример:

```
help
```

```
Supported commands:
QUIT
EXIT
CLEAR
EXPLAIN [ ( option [, ...] ) ] <query>
    options: FORMAT { TEXT | GRAPHVIZ | JSON }
            TYPE { LOGICAL | DISTRIBUTED | VALIDATE | IO }
DESCRIBE <table>
SHOW COLUMNS FROM <table>
SHOW FUNCTIONS
SHOW CATALOGS [LIKE <pattern>]
SHOW SCHEMAS [FROM <catalog>] [LIKE <pattern>]
SHOW TABLES [FROM <schema>] [LIKE <pattern>]
USE [<catalog>.]<schema>
```

Для очистки экрана терминала используйте команду `clear`. Для выхода из командной строки Trino и возврата в терминал ОС используйте команды `exit` или `quit`.

Выполнение выражений

После запуска командной строки Trino вы можете выполнять в ней SQL-выражения. После обработки выражения на экран будет выведен его результат и статистика. Пример:

```
select * from "hive-adh"."default"."solar_objects";
```

```
  name   |    mass    | diameter
---------+------------+----------
 Sun     | 1989100000 |  1392000
 Mercury |        330 |     4879
 Venus   |       4867 |    12104
 Earth   |       5972 |    12742
 Mars    |        642 |     6780
 Jupiter |    1898187 |   139822
 Saturn  |     568317 |   116464
 Uranus  |      86813 |    50724
 Neptune |     102413 |    49244
(9 rows)

Query 20250115_092203_00003_7thb6, FINISHED, 1 node
Splits: 1 total, 1 done (100,00%)
1,51 [9 rows, 172B] [5 rows/s, 114B/s]
```

В этом примере `hive-adh` — название каталога, `default` — имя схемы, а `solar_objects` — имя таблицы.

РЕКОМЕНДАЦИЯ

Используя настройки Trino Client в ADCM, можно указать каталог и схему Trino по умолчанию.

Настройка

По умолчанию компонент Trino Client сконфигурирован для работы с Trino Coordinator, доступным в кластере ADH, и не требует ручной настройки. Для тонкой настройки компонента Trino Client используйте настройки Trino Client в ADCM.

TLS/HTTPS

Компонент Trino Client поддерживает безопасное соединение с Trino Coordinator с использованием TLS. Для конфигурации безопасного соединения используйте раздел ***SSL Configuration*** в настройках компонента Trino Client в ADCM.

Удаленное подключение

Вы также можете подключиться к сервису Trino с удаленного хоста за пределами ADH-кластера. Для этого используйте клиент trino-cli. Он реализован с помощью JAR-файла, который может быть запущен в любой ОС при условии, что в ней установлена среда исполнения Java (Java Runtime Environment, JRE).

Для использования командной строки Trino нужно загрузить исполняемый JAR-файл. Ссылку на наиболее актуальный файл можно найти на сайте документации Trino. Если вы используете ОС Linux, переименуйте файл как *trino* и назначьте для него права на исполнение с помощью команды `chmod +x`. Если вы используете ОС Windows, запустите командную строку Windows от имени администратора перед запуском JAR-файла в ней.

Для запуска командной строки Trino откройте терминал в вашей ОС и используйте следующий синтаксис:

Windows

```
java -jar trino-cli-468-executable.jar http://coordinator.trino.example:18188
```

Linux

```
$ ./trino http://coordinator.trino.example:18188
```

При запуске командной строки Trino также можно использовать различные опции. Их список представлен на странице Command line interface документации Trino.

Подключение к Trino через JDBC

Владимир Адаменков

Содержание

Назад к содержанию

- Обзор
- Установка
- Использование
  - Подключение
  - Параметры подключения

Обзор

Сервис Trino поддерживает пользовательский доступ из Java-приложений с использованием JDBC-драйвера. Он использует клиентский протокол через HTTP/HTTPS для подключения к компоненту Trino Coordinator.

Установка

Перед установкой JDBC-драйвера убедитесь, что выполнены следующие условия:

1. Установлена среда выполнения Java (JRE) версии 8 или выше. Рекомендуется использовать версию 22 или выше.
2. Пользователи, от имени которых предполагается подключение к таблицам с помощью Trino, должны иметь права доступа, определенные в схеме `system.jdbc`.
3. Компонент Trino Coordinator должен быть доступен в сети.

Некоторые клиентские приложения уже имеют встроенный JDBC-драйвер Trino. Если в вашем приложении его нет, загрузите файл драйвера самостоятельно. Ссылку на наиболее актуальный файл можно найти на сайте документации Trino или в центральном репозитории Maven. Добавьте загруженный файл в директорию, записанную в classpath пользователей на тех машинах, с которых предполагается осуществлять доступ к Trino, и перезапустите клиентское приложение. Возможно, потребуется вручную зарегистрировать JDBC-драйвер Trino в клиентском приложении: некоторые из них делают это автоматически, некоторые — нет. За инструкциями по регистрации драйвера обратитесь к документации вашего клиентского приложения.

Использование

Подключение

После регистрации и настройки JDBC-драйвера Trino в клиентском приложении вы можете подключаться к Trino. Используйте следующие форматы JDBC URL:

```
jdbc:trino://<host>:<port>
jdbc:trino://<host>:<port>/<catalog>
jdbc:trino://<host>:<port>/<catalog>/<schema>
```

где:

- `<host>` — сетевой адрес узла компонента Trino Coordinator;
- `<port>` — порт прослушивания компонента Trino Coordinator (значение по умолчанию: 18188);
- `<catalog>` — название каталога Trino;
- `<schema>` — имя БД или схемы для подключения.

Параметры подключения

Можно задать дополнительные параметры подключения как параметры URL или как свойства, передаваемые классу `DriverManager`. Например, вам нужно подключиться к Trino с использованием следующих параметров:

- сетевой адрес и порт — `myorganization.org:18188`;
- каталог и схема — `hive/tech`;
- пользователь — `admin`;
- пароль — `MySecretPassword1984`.

В этом случае необходимо использовать следующие параметры URL:

```
String mycustomurl = "jdbc:trino://myorganization.org:18188/hive/tech?user=admin&password=MySecretPassword1984";
Connection connection = DriverManager.getConnection(mycustomurl);
```

Или если вы хотите передать параметры подключения классу `DriverManager`, задайте следующие свойства:

```
String url = "jdbc:trino://myorganization.org:18188/hive/tech";
Properties mycustomproperties = new Properties();
mycustomproperties.setProperty("user", "admin");
mycustomproperties.setProperty("password", "MySecretPassword1984");
Connection connection = DriverManager.getConnection(url, mycustomproperties);
```

Можно использовать комбинацию этих подходов, но дублирование параметров запрещено: параметр, заданный одним способом, нельзя задавать другим.

Список доступных параметров приведен на странице JDBC driver документации Trino.

Обзор Trino UI

Владимир Адаменков

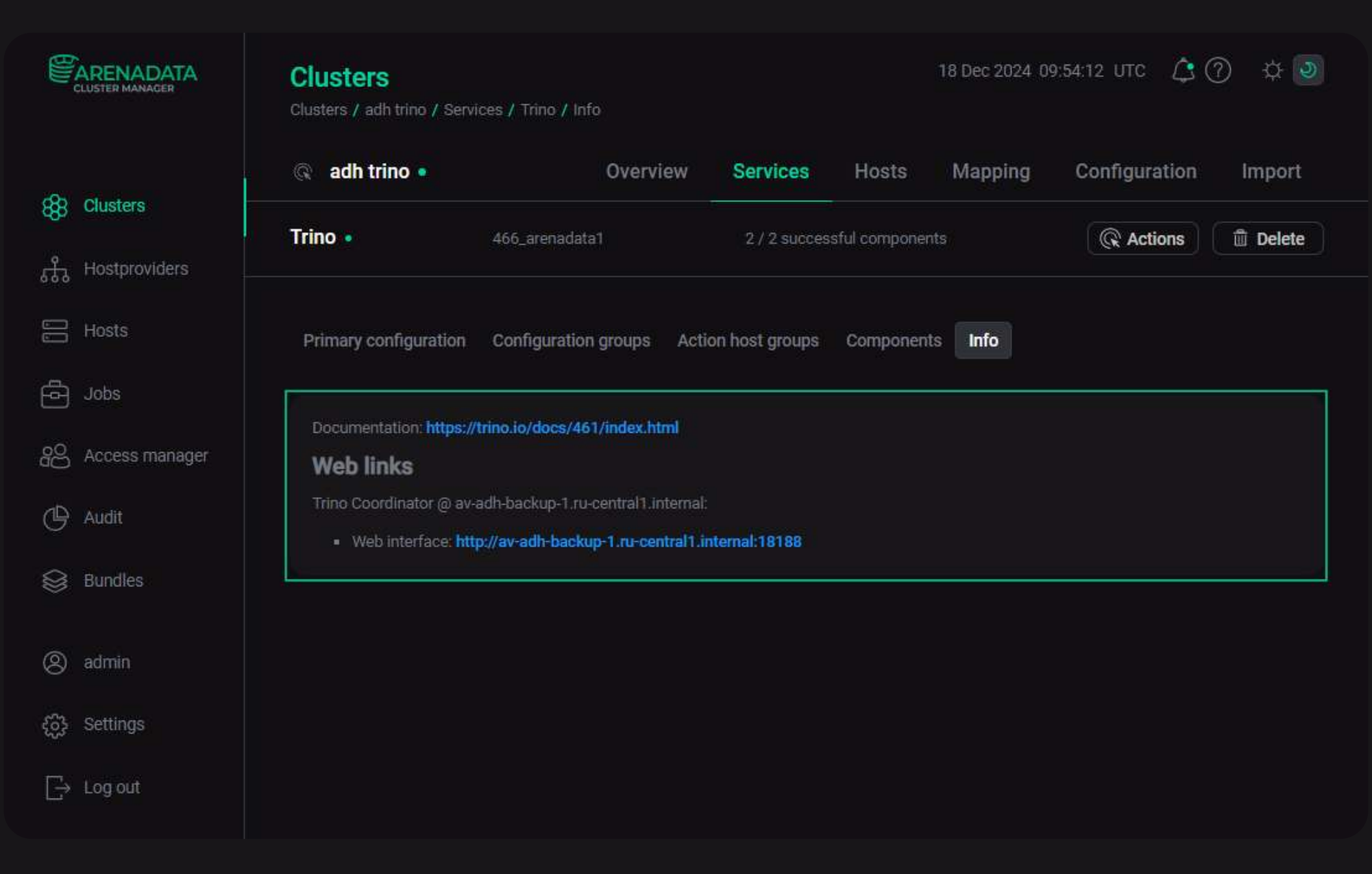
Содержание

Назад к содержанию

- Cluster overview
- Query details
  - Overview
  - Live plan
  - Stage performance
  - Splits
  - JSON
  - References

Сервис Trino имеет веб-интерфейс, позволяющий просматривать запросы к базам данных, расположенным в кластере Trino, и управлять их исполнением.

Ссылку на веб-интерфейс Trino можно найти на вкладке ***Info*** сервиса Trino в веб-интерфейсе ADCM.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / adh trino / Services / Trino / Info
18 Dec 2024 09:54:12 UTC
adh trino
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Trino
466_arenadata1
2 / 2 successful components
Actions
Delete
Primary configuration
Configuration groups
Action host groups
Components
Info

Documentation: https://trino.io/docs/461/index.html

Web links

Trino Coordinator @ av-adh-backup-1.ru-central1.internal:

- Web interface: http://av-adh-backup-1.ru-central1.internal:18188

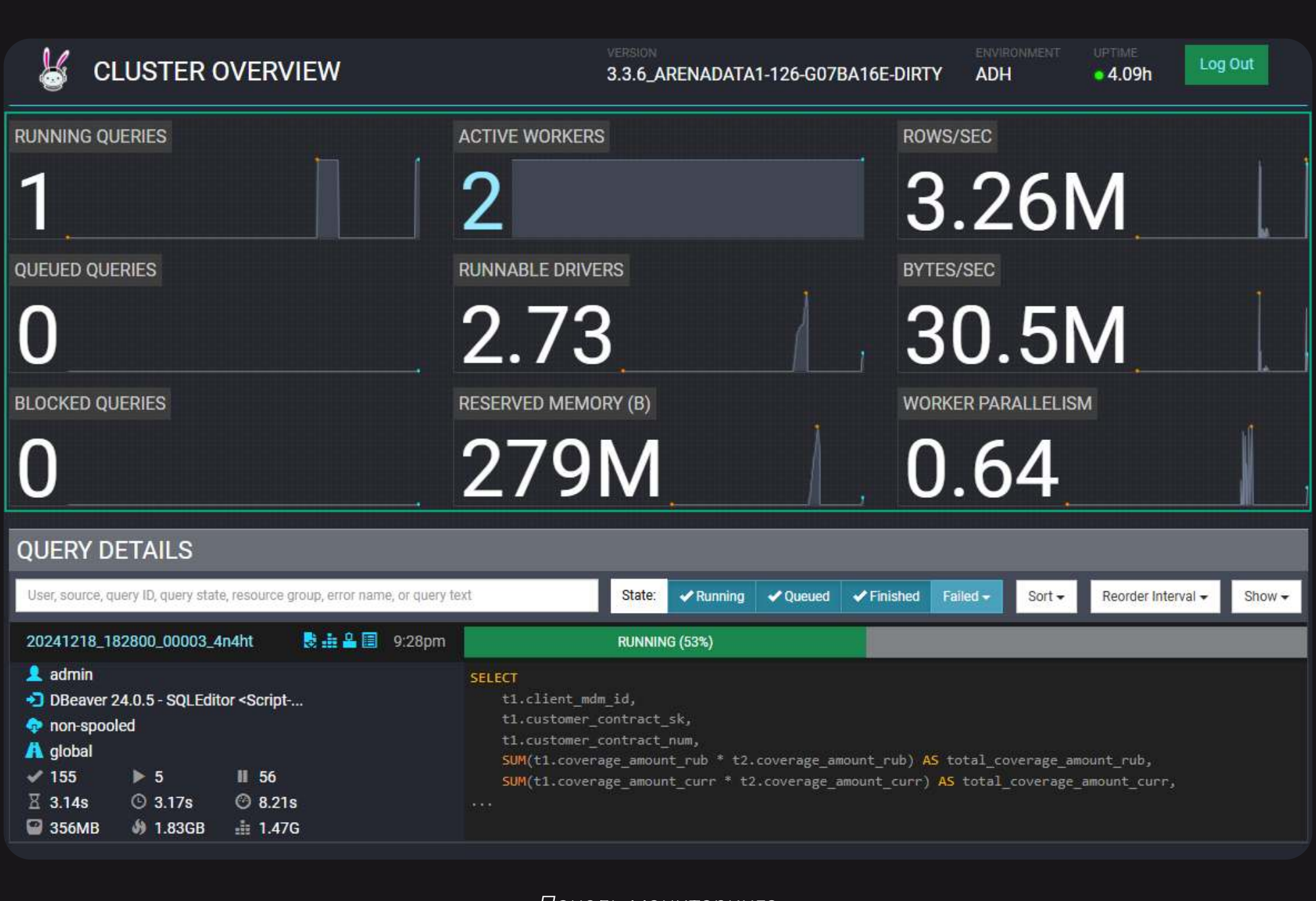
Ссылка на веб-интерфейс Trino

Cluster overview

Начальной страницей веб-интерфейса Trino является страница ***Cluster overview***. В её верхней части расположена следующая информация:

- ***Version*** — название версии установленного дистрибутива Trino.
- ***Environment*** — название окружения, в котором работает данный кластер Trino.
- ***Uptime*** — время непрерывной работы компонента Trino Coordinator.
- ***Log out*** — кнопка, позволяющая выйти из веб-интерфейса.

Под этой информацией расположена панель мониторинга.


CLUSTER OVERVIEW
VERSION
3.3.6_ARENADATA1-126-G07BA16E-DIRTY
ENVIRONMENT
ADH
UPTIME
4.09h
Log Out
RUNNING QUERIES
1
ACTIVE WORKERS
2
ROWS/SEC
3.26M
QUEUED QUERIES
0
RUNNABLE DRIVERS
2.73
BYTES/SEC
30.5M
BLOCKED QUERIES
0
RESERVED MEMORY (B)
279M
WORKER PARALLELISM
0.64
QUERY DETAILS
User, source, query ID, query state, resource group, error name, or query text
State:
Running
Queued
Finished
Failed
Sort
Reorder Interval
Show
20241218_182800_00003_4n4ht
9:28pm
RUNNING (53%)
admin
DBeaver 24.0.5 - SQLEditor <Script-...
non-spooled
global
155
5
56
3.14s
3.17s
8.21s
356MB
1.83GB
1.47G

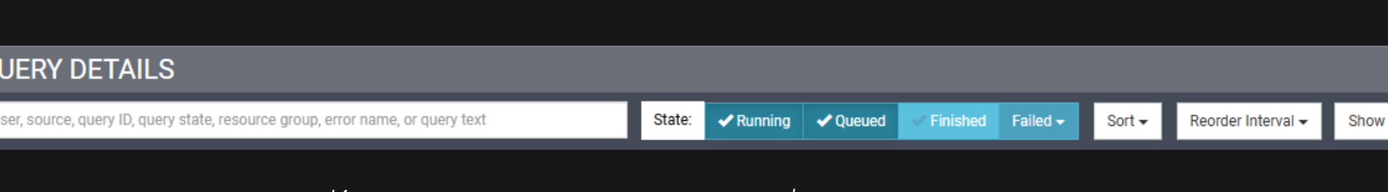
```
SELECT
    t1.client_mdm_id,
    t1.customer_contract_sk,
    t1.customer_contract_num,
    SUM(t1.coverage_amount_rub * t2.coverage_amount_rub) AS total_coverage_amount_rub,
    SUM(t1.coverage_amount_curr * t2.coverage_amount_curr) AS total_coverage_amount_curr,
...
```

Панель мониторинга

Назначение элементов панели мониторинга:

- ***Running queries*** — количество запросов, находящихся в работе в текущий момент.
- ***Active workers*** — количество активных компонентов Trino Worker.
- ***Rows/sec*** — количество строк таблиц, обрабатываемых в секунду на текущий момент.
- ***Queued queries*** — количество запросов, находящихся в очереди.
- ***Runnable drivers*** — количество работоспособных драйверов на текущий момент.
- ***Bytes/sec*** — размер данных, обрабатываемых в секунду на текущий момент.
- ***Blocked queries*** — количество заблокированных запросов.
- ***Reserved memory*** — размер памяти виртуальной машины Java, используемой в Trino, зарезервированной для исполняемых в настоящий момент запросов.
- ***Worker parallelism*** — мера эффективности использования нескольких компонентов Trino Worker на текущий момент.

Под панелью мониторинга расположена секция ***Query details***. В ней отображаются статистические данные всех типов запросов. Также здесь можно искать запросы, сортировать и фильтровать список запросов по различным критериям.


QUERY DETAILS
User, source, query ID, query state, resource group, error name, or query text
State:
Running
Queued
Finished
Failed
Sort
Reorder Interval
Show

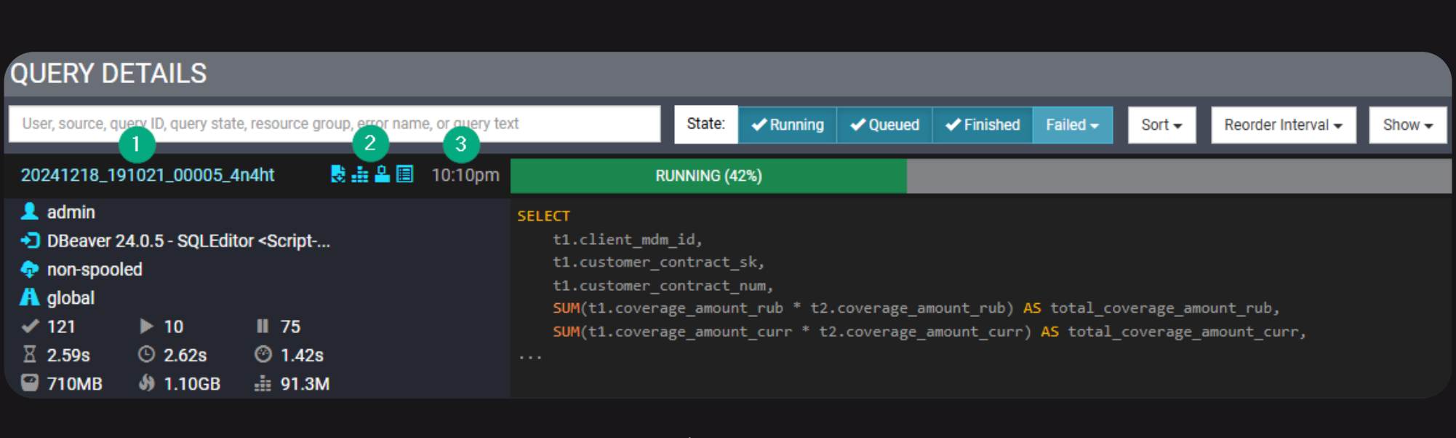
Инструменты поиска, сортировки и фильтрации запросов

В поле ввода можно искать запросы по следующим параметрам:

- имя пользователя, запустившего запрос;
- источник запроса;
- идентификатор запроса;
- состояние запроса;
- группа ресурсов запроса;
- название ошибки, с которой завершился запрос;
- текст запроса.

Справа от поля поиска расположены инструменты фильтрации и сортировки. Для просмотра выполняемых, находящихся в очереди или успешно выполненных запросов нажмите кнопки-переключатели ***Running***, ***Queued*** или ***Finished*** соответственно. Выпадающий список ***Failed*** объединяет переключатели для отображения запросов, которые завершились неудачей по разным причинам: внутренняя ошибка (***Internal error***), внешняя ошибка (***External error***), ошибка ресурсов (***Resources error***) или ошибка пользователя (***User error***). С помощью выпадающего списка ***Show*** можно задать количество запросов для отображения. Значения из выпадающего списка ***Sort*** позволяют сортировать список запросов по различным параметрам. С помощью списка ***Reorder interval*** можно установить частоту обновления списка или отключить обновление.

Под вышеописанной частью начинается список запросов. Каждая запись о запросе показывает его основную информацию.


QUERY DETAILS
User, source, query ID, query state, resource group, error name, or query text
State:
Running
Queued
Finished
Failed
Sort
Reorder Interval
Show
1
2
3
20241218_191021_00005_4n4ht
10:10pm
RUNNING (42%)
admin
DBeaver 24.0.5 - SQLEditor <Script-...
non-spooled
global
121
10
75
2.59s
2.62s
1.42s
710MB
1.10GB
91.3M

```
SELECT
    t1.client_mdm_id,
    t1.customer_contract_sk,
    t1.customer_contract_num,
    SUM(t1.coverage_amount_rub * t2.coverage_amount_rub) AS total_coverage_amount_rub,
    SUM(t1.coverage_amount_curr * t2.coverage_amount_curr) AS total_coverage_amount_curr,
...
```

Основная информация о запросе

Заголовок в верхней левой части содержит следующую информацию:

① Query ID — идентификатор запроса. Нажатие на него открывает вкладку ***Overview*** страницы Query details.

② Иконки Query JSON, Stage performance, Query plan, и References — нажатие на них равнозначно нажатию на вкладки с такими же названиями на странице ***Query details***.

③ Submit time — время, когда запрос был отправлен на исполнение.

Под заголовком собрана следующая информация:

- имя пользователя, запустившего запрос;
- источник запроса;
- кодировка протокола;
- группа ресурсов запроса;
- количество выполненных, выполняемых и находящихся в очереди запросов;
- реальное время исполнения запроса за вычетом времени нахождения в очереди;
- полное реальное время исполнения запроса;
- процессорное время исполнения запроса;
- полный объем зарезервированной памяти на текущий момент;
- пиковый полный объем зарезервированной памяти;
- кумулятивная пользовательская память.

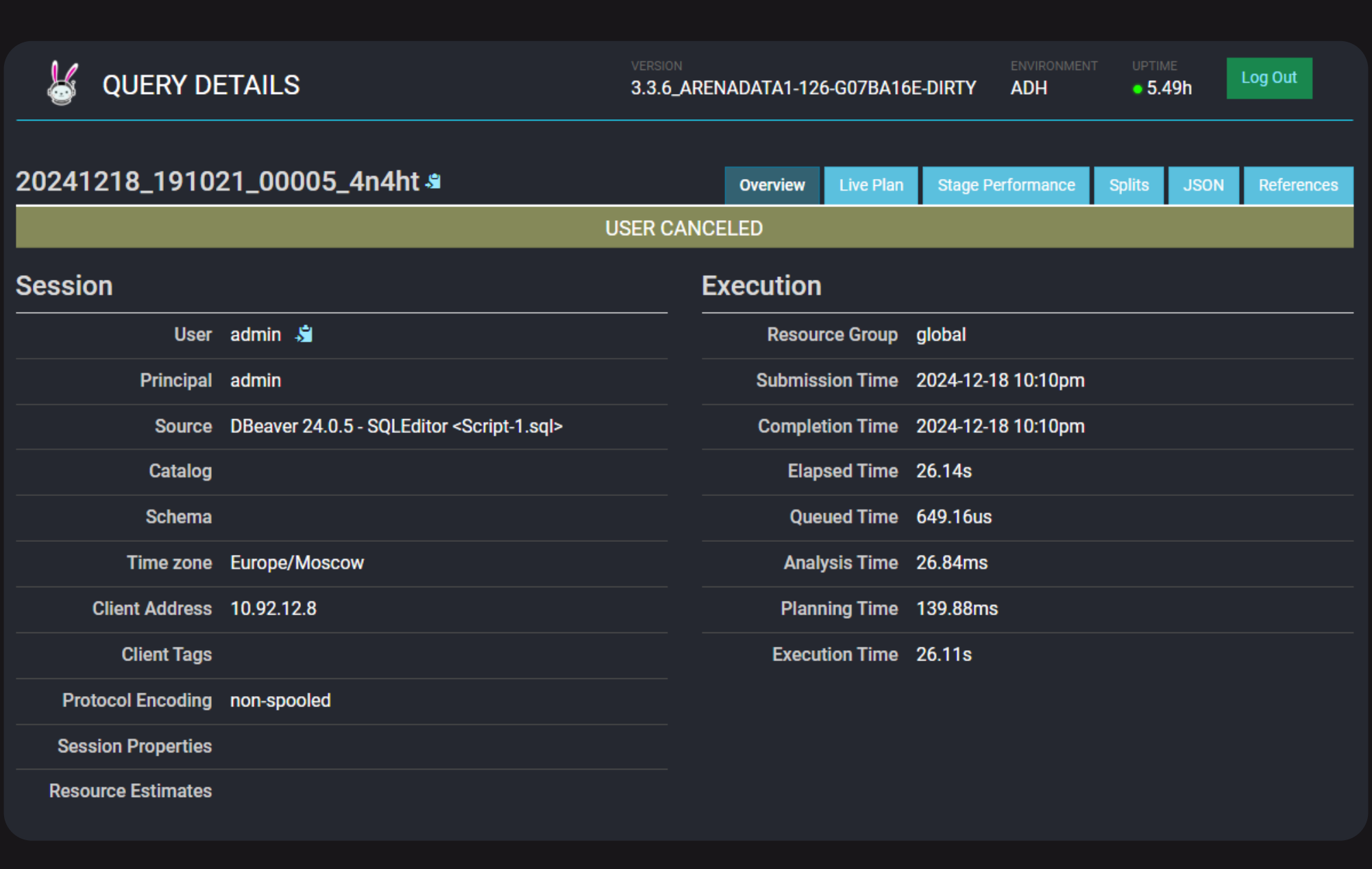
В правой части записи о запросе можно увидеть статус хода выполнения запроса или его итог в зависимости от того, исполняется ли этот запрос или уже завершен. Также здесь приведен текст запроса (или его начало).

Query details

Страницу ***Query details*** можно открыть, нажав на идентификатор запроса на странице Cluster overview. Она содержит несколько вкладок, описанных ниже.

Overview

Вкладка ***Overview*** содержит общую информацию о запросе.


QUERY DETAILS
VERSION
3.3.6_ARENADATA1-126-G07BA16E-DIRTY
ENVIRONMENT
ADH
UPTIME
5.49h
Log Out
20241218_191021_00005_4n4ht
Overview
Live Plan
Stage Performance
Splits
JSON
References
USER CANCELED
Session
Execution

Session		Execution	
User	admin	Resource Group	global
Principal	admin	Submission Time	2024-12-18 10:10pm
Source	DBeaver 24.0.5 - SQLEditor <Script-1.sql>	Completion Time	2024-12-18 10:10pm
Catalog		Elapsed Time	26.14s
Schema		Queued Time	649.16us
Time zone	Europe/Moscow	Analysis Time	26.84ms
Client Address	10.92.12.8	Planning Time	139.88ms
Client Tags		Execution Time	26.11s
Protocol Encoding	non-spooled		
Session Properties			
Resource Estimates			

Вкладка Overview страницы Query details

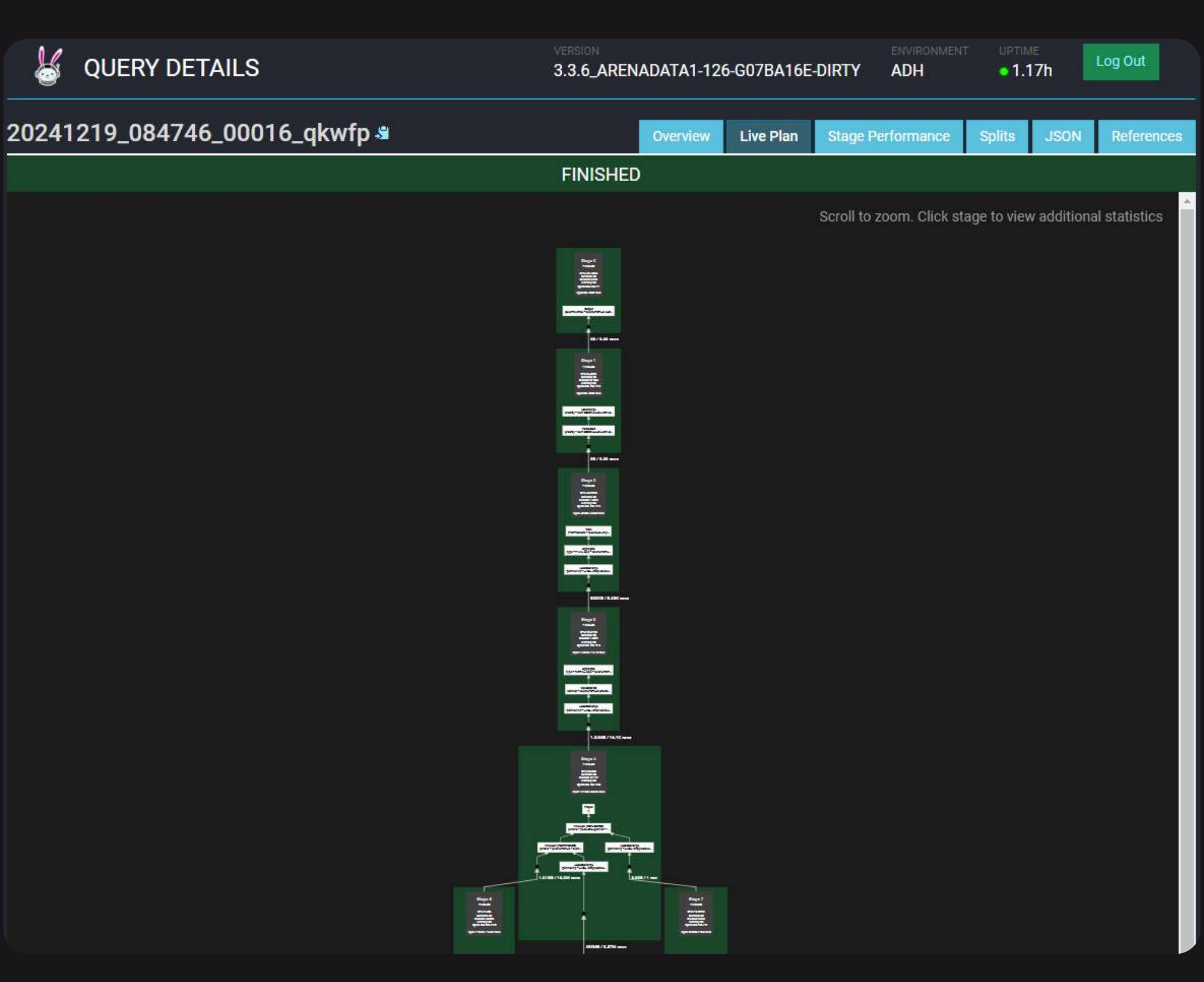
Здесь можно прервать или завершить исполнение запроса с помощью кнопок ***Preempt*** и ***Kill*** соответственно, если он ещё исполняется. Также здесь собрана подробная информация о следующих характеристиках запроса:

- ***Session*** — пользователь, источник, расположение и другое.
- ***Execution*** — различные тайминги запроса.
- ***Resource utilization summary*** — статистика использования ресурсов при исполнении запроса.
- ***Timeline*** — статистика метрик исполнения запроса с точки зрения таймлайна.
- ***Error information*** — тип и код ошибки, а также листинг трассировки стека.

Внизу этой вкладки также расположен полный текст запроса и список стадий выполнения запроса с подробностями.

Live plan

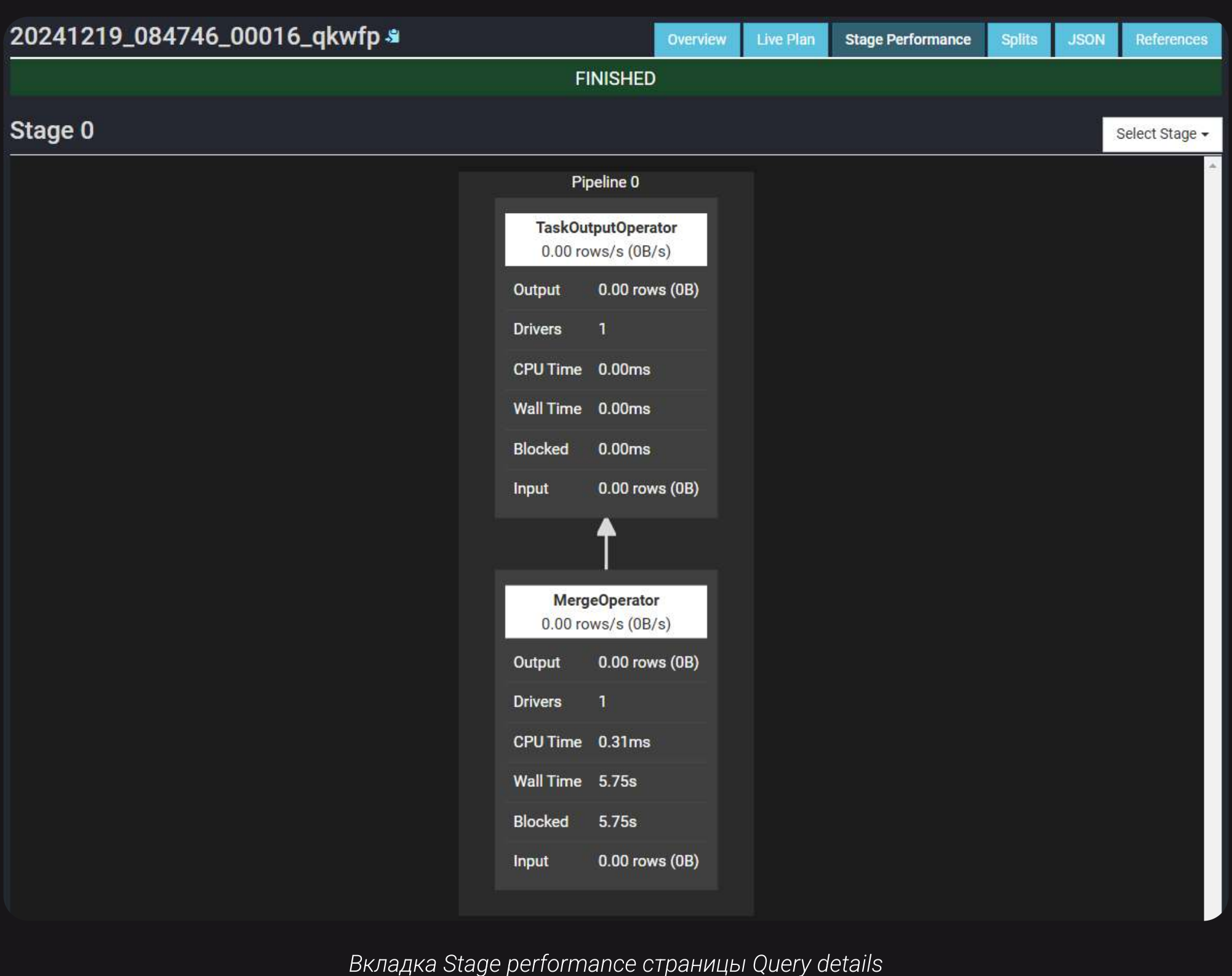
На вкладке ***Live plan*** расположена схема исполнения запроса по стадиям с подробной информацией о каждой стадии: статус выполнения, использование ресурсов, обработанные выражения и направление дальнейшей обработки запроса. Схему можно приближать и отдалять с помощью колеса мыши, а также двигать с помощью зажатой левой кнопки. Нажатие на одну из стадий в схеме откроет вкладку Stage performance, где будет выбрана эта стадия.


QUERY DETAILS
VERSION
3.3.6_ARENADATA1-126-G07BA16E-DIRTY
ENVIRONMENT
ADH
UPTIME
1.17h
Log Out
20241219_084746_00016_qkwfp
Overview
Live Plan
Stage Performance
Splits
JSON
References
FINISHED
Scroll to zoom. Click stage to view additional statistics

Вкладка Live plan страницы Query details

Stage performance

На вкладке ***Stage performance*** показана дальнейшая детализация выбранной стадии исполнения запроса. Стадию для просмотра можно выбрать в правом верхнем углу экрана (список ***Select stage***).


20241219_084746_00016_qkwfp
Overview
Live Plan
Stage Performance
Splits
JSON
References
FINISHED
Stage 0
Select Stage
Pipeline 0
TaskOutputOperator
0.00 rows/s (0B/s)
MergeOperator
0.00 rows/s (0B/s)

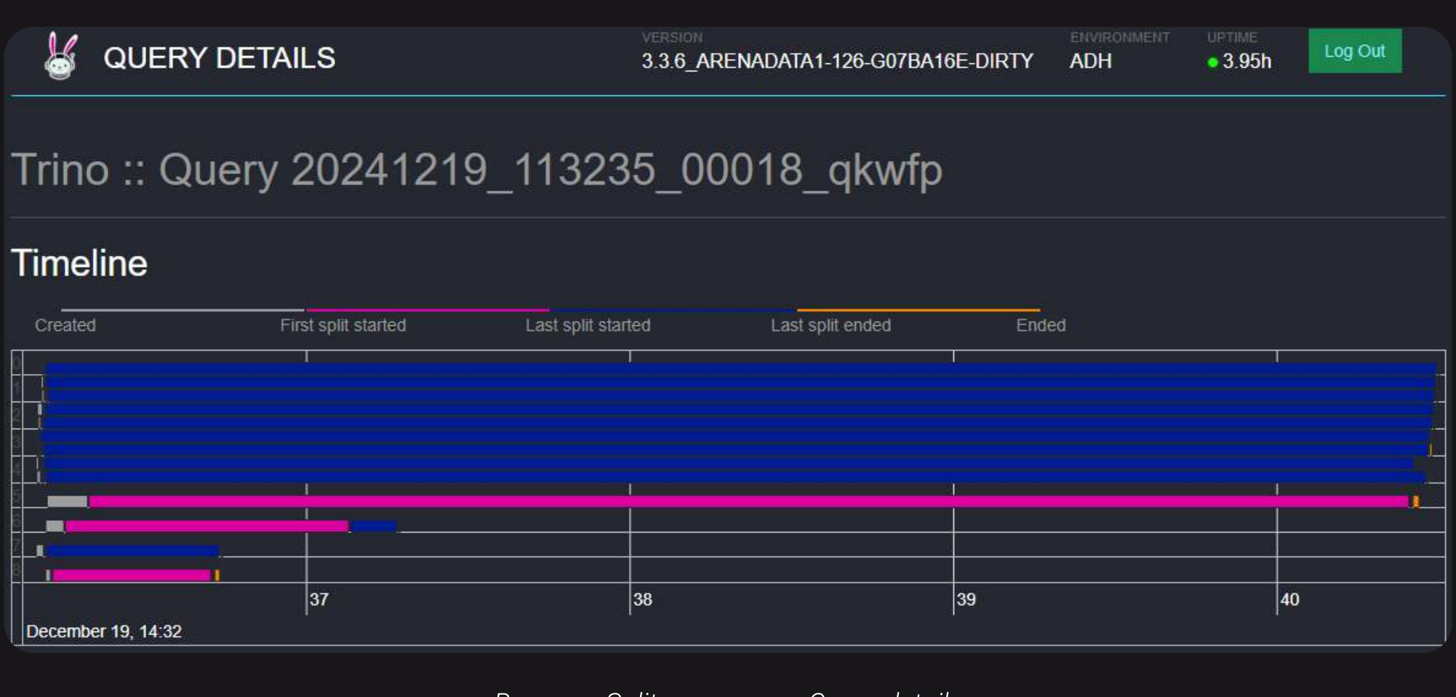
TaskOutputOperator	
Output	0.00 rows (0B)
Drivers	1
CPU Time	0.00ms
Wall Time	0.00ms
Blocked	0.00ms
Input	0.00 rows (0B)

MergeOperator	
Output	0.00 rows (0B)
Drivers	1
CPU Time	0.31ms
Wall Time	5.75s
Blocked	5.75s
Input	0.00 rows (0B)

Вкладка Stage performance страницы Query details

Splits

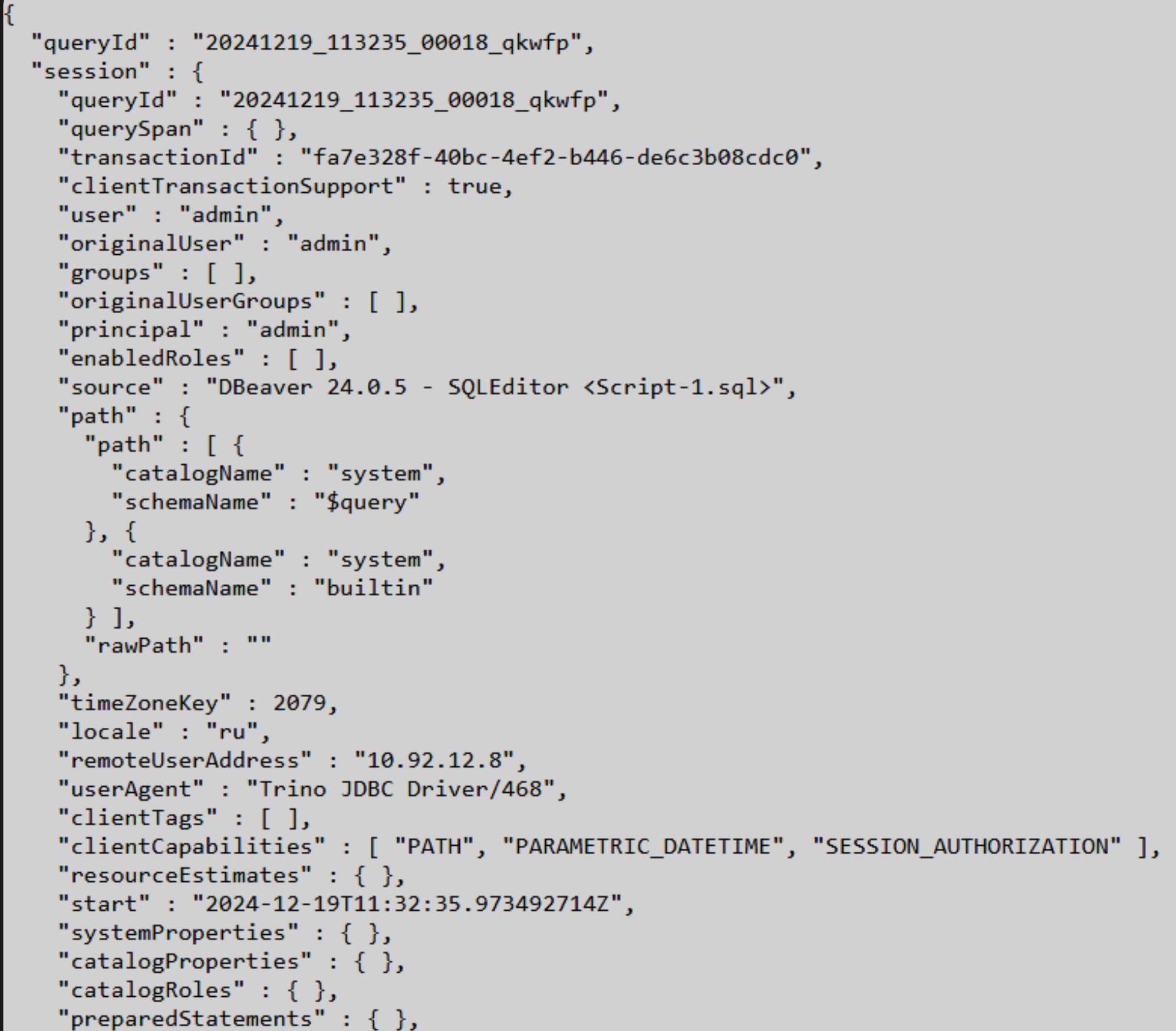
На вкладке ***Splits*** изображен график таймлайнов сплитов для каждой стадии исполнения запроса. Каждый таймлайн показывает, сколько времени прошло от создания стадии до запуска первого сплита, затем до запуска последнего сплита, затем до завершения последнего сплита, и до завершения стадии. Эти интервалы размечены цветами на таймлайне.


QUERY DETAILS
VERSION
3.3.6_ARENADATA1-126-G07BA16E-DIRTY
ENVIRONMENT
ADH
UPTIME
3.95h
Log Out
Trino :: Query 20241219_113235_00018_qkwfp
Timeline
Created
First split started
Last split started
Last split ended
Ended
37
38
39
40
December 19, 14:32

Вкладка Splits страницы Query details

JSON

Нажатие на вкладку ***JSON*** откроет вкладку бразузера с полным JSON-представлением запроса.

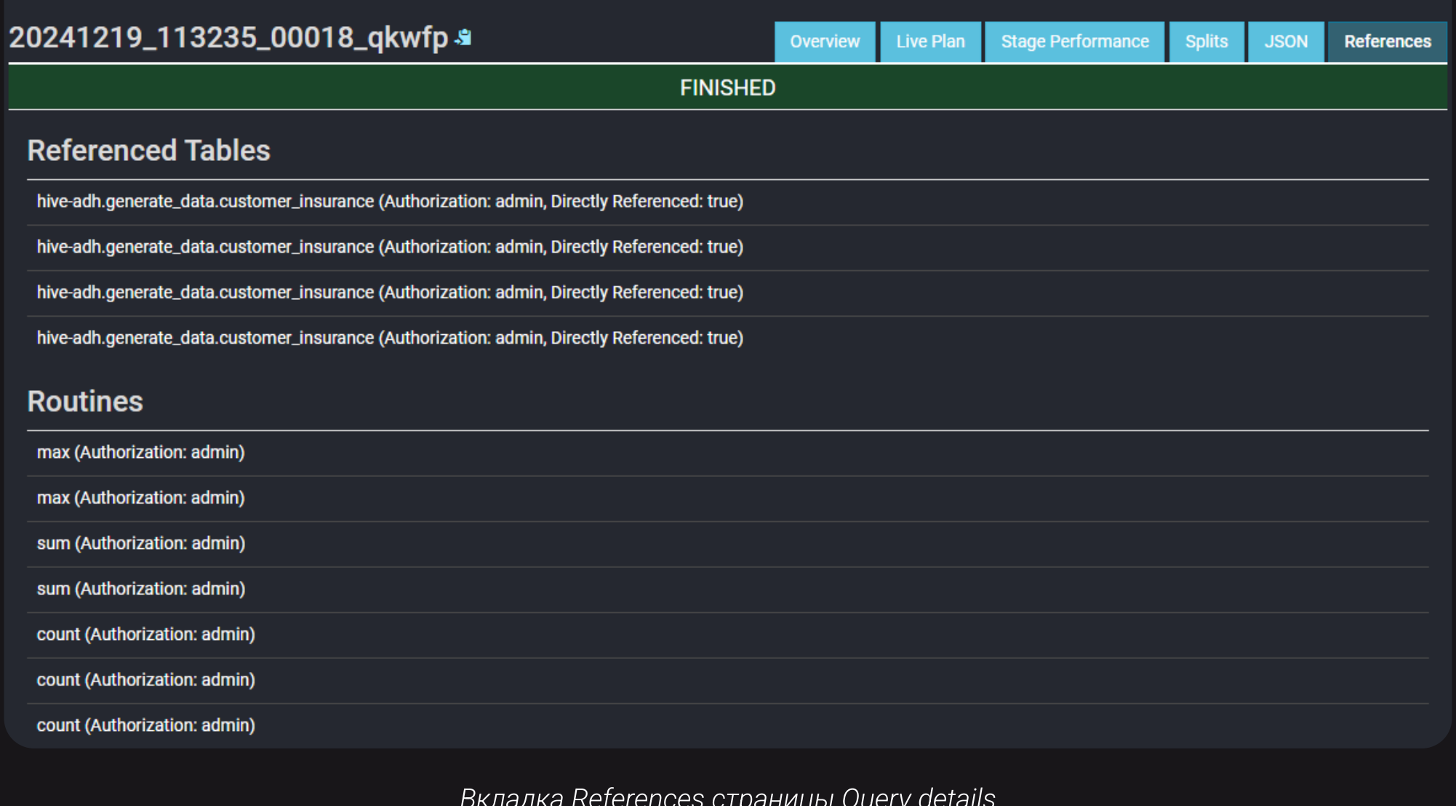


```
{
  "queryId" : "20241219_113235_00018_qkwfp",
  "session" : {
    "queryId" : "20241219_113235_00018_qkwfp",
    "querySpan" : { },
    "transactionId" : "fa7e328f-40bc-4ef2-b446-de6c3b08cdc0",
    "clientTransactionSupport" : true,
    "user" : "admin",
    "originalUser" : "admin",
    "groups" : [ ],
    "originalUserGroups" : [ ],
    "principal" : "admin",
    "enabledRoles" : [ ],
    "source" : "DBeaver 24.0.5 - SQLEditor <Script-1.sql>",
    "path" : {
      "path" : [ {
        "catalogName" : "system",
        "schemaName" : "$query"
      }, {
        "catalogName" : "system",
        "schemaName" : "builtin"
      } ],
      "rawPath" : ""
    },
    "timeZoneKey" : 2079,
    "locale" : "ru",
    "remoteUserAddress" : "10.92.12.8",
    "userAgent" : "Trino JDBC Driver/468",
    "clientTags" : [ ],
    "clientCapabilities" : [ "PATH", "PARAMETRIC_DATETIME", "SESSION_AUTHORIZATION" ],
    "resourceEstimates" : { },
    "start" : "2024-12-19T11:32:35.973492714Z",
    "systemProperties" : { },
    "catalogProperties" : { },
    "catalogRoles" : { },
    "preparedStatements" : { },
    "protocolName" : "Trino",
    "timeZone" : "Europe/Moscow"
  },
```

Фрагмент JSON-представления запроса

References

На вкладке ***References*** перечислены таблицы, на которые сослался запрос прямо или косвенно, а также функции, вызванные в процессе исполнения запроса.


REFERENCES
VERSION
3.3.6_ARENADATA1-126-G07BA16E-DIRTY
ENVIRONMENT
ADH
UPTIME
4.39h
Log Out
20241219_113235_00018_qkwfp
Overview
Live Plan
Stage Performance
Splits
JSON
References
FINISHED

Referenced Tables

- hive-adh.generate_data.customer_insurance (Authorization: admin, Directly Referenced: true)
- hive-adh.generate_data.customer_insurance (Authorization: admin, Directly Referenced: true)
- hive-adh.generate_data.customer_insurance (Authorization: admin, Directly Referenced: true)
- hive-adh.generate_data.customer_insurance (Authorization: admin, Directly Referenced: true)

Routines

- max (Authorization: admin)
- max (Authorization: admin)
- sum (Authorization: admin)
- sum (Authorization: admin)
- count (Authorization: admin)
- count (Authorization: admin)
- count (Authorization: admin)

Вкладка References страницы Query details

Управление каталогами в Trino

Константин Алпашкин

Содержание

Назад к содержанию

Настройка каталогов

Просмотр доступных каталогов

Преднастроенные каталоги

Управление каталогами с помощью ADCM

Режимы управления каталогом

Параметры управления каталогами

Каталог в Trino — это набор конфигурационных свойств, которые представляют один источник данных (например, базу данных) и обеспечивают доступ к нему для движка Trino. В каждом каталоге указан коннектор, а также другие параметры, необходимые для взаимодействия с источником данных.

С помощью каталогов Trino можно обращаться к источнику данных непосредственно в SQL-запросах, указав имя каталога. Например:

```sql
SELECT * FROM "hive-adh"."default"."transactions";
```

Где:

- `hive-adh` — имя встроенного Hive-каталога.
- `default` — имя схемы, которая служит для группировки таблиц, представлений и так далее. Например, схема в Trino эквивалентна базе данных в Hive. Для других источников данных схема может сопоставляться с другой сущностью.
- `transactions` — имя таблицы для получения данных.

В приведенном выше примере показан каталог для работы с Hive, однако каталоги могут использовать множество коннекторов для работы с различными системами хранения данных. Таким образом, каталоги являются абстракцией для обращения к различным источникам данных.

Вы можете использовать несколько каталогов в одном SQL-выражении, чтобы получить информацию из разных источников данных одним запросом. Например, при выполнении следующего SQL-запроса Trino получает данные из двух независимых источников: базы данных MySQL и хранилища Hive. Получив данные с помощью соответствующих коннекторов, Trino применяет операцию `JOIN` и возвращает окончательный результат клиенту.

```sql
SELECT txns.*, ords.*
FROM "mysql_test_catalog"."demo_schema"."transactions" txns
JOIN "hive-adh"."demo_schema"."orders" ords
ON txns.acc_id = ords.account_id;
```

Настройка каталогов

Каталоги настраиваются с помощью файлов *<catalog_name>.properties*, которые хранятся в директории каталогов Trino. По умолчанию данная директория расположена в */etc/trino/conf/catalog* на хостах с компонентами Trino. Для изменения директории каталогов по умолчанию используйте параметр ***catalog.config-dir*** в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Components*** → ***Trino Coordinator*** → ***node.properties***). Пример содержимого директории */etc/trino/conf/catalog/* приведен ниже:

```
$ ls -l /etc/trino/conf/catalog
```

```
-rw-r--r--. 1 trino hadoop 433 Dec 19 14:34 hive-adh.properties
-rw-r--r--. 1 trino hadoop 280 Dec 20 10:21 iceberg-adh.properties
```

Файлы *.properties* конфигурации каталога содержат следующую информацию:

- Коннектор, используемый каталогом (свойство `connector.name`). Данное свойство является обязательным в файле *<catalog_name>.properties*.
- Детали подключения для доступа к источнику данных, такие как URL-адреса подключения, учетные данные и так далее.
- Настройки безопасности.
- Другие параметры, необходимые для корректного взаимодействия с источником данных. Например, формат данных, кодеки сжатия и так далее.

Ниже приведен пример встроенного каталога Hive (*hive-adh.properties*), который используется сервисом Trino для доступа к Hive:

```
connector.name=hive
fs.hadoop.enabled=True
hive.compression-codec=SNAPPY
hive.config.resources=/etc/hadoop/conf/core-site.xml,/etc/hadoop/conf/hdfs-site.xml
hive.hdfs.impersonation.enabled=False
hive.metastore.thrift.client.ssl.enabled=False
hive.metastore.thrift.impersonation.enabled=True
hive.metastore.uri=thrift://ka-adh-3.ru-central1.internal:9083
hive.non-managed-table-writes-enabled=True
hive.storage-format=PARQUET
```

Имя файла *<catalog_name>.properties* определяет имя каталога. Например, *hive-adh.properties* хранит настройки каталога с именем `hive-adh`.

Просмотр доступных каталогов

Для получения списка каталогов, доступных в сервисе Trino, используйте команду:

```sql
SHOW CATALOGS;
```

Пример вывода:

```
Catalog
---------
hive-adh
iceberg-adh
test-postgresql-catalog
system
(4 rows)
```

Чтобы увидеть все каталоги, использующие определенный коннектор, отправьте следующий запрос:

```sql
SELECT *
FROM system.metadata.catalogs
WHERE connector_name='hive';
```

Пример вывода:

```
catalog_name|connector_id|connector_name|
------------+------------+--------------+
hive-adh    |hive-adh    |hive          |
```

Преднастроенные каталоги

Сервис Trino содержит следующие преднастроенные каталоги:

- Каталог Hive с именем `hive-adh`. Используется для взаимодействия с сервисом Hive, развернутым в ADH-кластере.
- Каталог Iceberg с именем `iceberg-adh`. Используется для работы с таблицами Iceberg.

Если в кластере ADH установлен сервис Hive, конфигурации этих каталогов генерируются автоматически с использованием актуальных значений для доступа к сервису Hive. Параметры преднастроенных каталогов можно редактировать в ADCM UI. Для этого используйте разделы настроек ***Hive configuration*** и ***Iceberg configuration*** (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Primary configuration***).

РЕКОМЕНДАЦИЯ

Чтобы переопределить параметры преднастроенных каталогов, используйте ADCM-настройки ***Custom Hive configuration*** и ***Custom Iceberg configuration*** соответственно.

Управление каталогами с помощью ADCM

Добавление новых каталогов в Trino предполагает создание файлов *<catalog_name>.properties* в директории каталогов (*/etc/trino/conf/catalog/*) на всех хостах кластера Trino. ADCM предоставляет пользовательский интерфейс для добавления каталогов вместо ручных манипуляций с файлами *.properties*.

Ниже показан пример создания каталога Trino, который взаимодействует с Arenadata Postgres (ADPG) через коннектор PostgreSQL:

1. В ADCM перейдите на страницу ***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Primary configuration*** и активируйте опцию ***Custom catalogs***.
2. Разверните секцию ***Custom catalogs*** и укажите свойства, перечисленные в таблице ниже.

catalog_name	key	value
test_catalog_adpg	connector.name	postgresql
	connection-url	jdbc:postgresql://<ADPG_master_host>:5432
	connection-user	postgres
	connection-password	postgres

3. Сохраните конфигурацию и перезапустите сервис Trino. При перезапуске ADCM создает файлы *test_catalog_adpg.properties* на всех хостах ADH, где установлены компоненты Trino. Проверьте, что файлы *test_catalog_adpg.properties* были созданы:

   ```
   $ ls /etc/trino/conf/catalog/
   ```

   Результат:

   ```
   hive-adh.properties  iceberg-adh.properties  test_catalog_adpg.properties
   ```

4. Подключитесь к Trino (например, через Trino CLI или SQL-клиент типа DBeaver) и запросите список доступных каталогов:

   ```sql
   SHOW CATALOGS;
   ```

   Новый каталог доступен в списке:

   ```
   Catalog          |
   -----------------+
   hive-adh         |
   iceberg-adh      |
   system           |
   test_catalog_adpg|
   ```

Теперь можно обращаться к серверу ADPG, используя каталог `test_catalog_adpg`. Например:

```sql
SELECT * FROM "test_catalog_adpg"."public"."book";
```

Результат:

```
book_id|title                |author_id|in_stock|
-------+---------------------+---------+--------+
      1|Mrs. Dalloway        |        1|      25|
      2|To the Lighthouse    |        1|      13|
      3|To Kill a Mockingbird|        2|       0|
      4|The Great Gatsby     |        3|       3|
      5|The Lord of the Rings|        4|      45|
```

Режимы управления каталогом

Сервис Trino предоставляет несколько режимов для управления каталогами:

- `static`. Считывает конфигурационные файлы и создает объекты каталогов в момент запуска сервера Trino. Добавление нового каталога требует перезапуска сервиса Trino.
- `dynamic`. Позволяет управлять каталогами динамически, используя команды `CREATE CATALOG` и `DROP CATALOG`.
- `DBCatalogManager`. Позволяет управлять каталогами динамически и хранит метаданные каталога в базе данных ADPG/PostgreSQL. Позволяет обновлять конфигурации каталогов без перезапуска сервиса Trino.

Для изменения режима используйте параметр ***catalog.management*** в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Components*** → ***Trino Coordinator*** → ***Trino catalog management***).

Параметры управления каталогами

Следующие конфигурационные параметры используются для управления жизненным циклом каталогов в Trino. По умолчанию сервис Trino запускается с дефолтными значениями этих параметров. Чтобы переопределить конфигурационный параметр, используйте поле ***Custom config.properties*** в ADCM-настройках компонента ***Trino Coordinator***.

Свойство	Описание	Значение по умолчанию
catalog.prune.update-interval	Требует установки `catalog.management=dynamic`. Указывает интервал для физического удаления файлов каталога. Удаление каталога не прерывает выполняющиеся запросы (если таковые имеются), однако делает каталог недоступным для новых запросов	5s
catalog.store	Допустимые значения: `file`, `memory`. Требует установки `catalog.management=dynamic`. Если установлено значение `file`, создание и удаление каталогов с помощью команд SQL также добавляет/удаляет файлы *<catalog_name>.properties*. Существующие файлы каталогов также считываются при запуске сервиса Trino. Если установлено значение `memory`, объекты каталогов хранятся только в памяти, а все файлы *.properties* игнорируются при старте сервиса	file
catalog.config-dir	Требует установки `catalog.management=static` или `catalog.store=file`. Определяет директорию с файлами *<catalog_name>.properties*	/etc/trino/conf/catalog/
catalog.disabled-catalogs	Требует установки `catalog.management=static` или `catalog.store=file`. Разделенный запятыми список каталогов, которые следует игнорировать при чтении файлов *<catalog_name>.properties* во время запуска сервиса	—
catalog.read-only	Требует установки `catalog.store=file`. Если установлено значение `true`, существующие файлы *<catalog_name>.properties* не могут быть удалены с помощью команды `DROP CATALOG`, а также не могут быть созданы новые файлы с дублирующимися именами при выполнении команды `CREATE CATALOG`. В итоге при перезапуске сервиса Trino остаются лишь каталоги, определенные в файлах *<catalog_name.properties>*	false

Использование DBCatalogManager

Константин Алпашкин

Содержание

Назад к содержанию

Обзор

Активация DBCatalogManager

Настройка

- Версионирование и удаление каталогов
- Шифрование
- Миграция

Пример подключения к ADPG

Обзор

DBCatalogManager — это одна из реализаций управления каталогами Trino, которая хранит конфигурации каталогов в базе данных ADPG/PostgreSQL, а не в файлах *.properties*. Хранение метаданных каталога в БД — полезная практика с точки зрения совместного доступа, версионирования, синхронизации настроек, а также безопасности.

Данная реализация позволяет динамически управлять каталогами Trino с помощью команд `CREATE CATALOG`/`DROP CATALOG`. DBCatalogManager позволяет обновлять свойства каталогов без необходимости перезапускать сервис Trino. DBCatalogManager поддерживает шифрование для хранения конфиденциальных параметров конфигурации в зашифрованном виде.

По умолчанию DBCatalogManager настроен для работы с ADH-сервисом ADPG. DBCatalogManager может подключаться к ADPG-сервису "из коробки", не требуя дополнительных настроек, а также может инициализировать все необходимые сущности на стороне ADPG (БД, пользователь, роль). Больше информации об использовании DBCatalogManager с сервисом ADPG можно найти в разделе Пример подключения к ADPG.

Активация DBCatalogManager

DBCatalogManager можно активировать с помощью ADCM. Для этого укажите значение `DB` в параметре ***catalog.management*** (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Components*** → ***Trino Coordinator*** → ***Trino catalog management***) и перезапустите сервис Trino.

> **РЕКОМЕНДАЦИЯ**
>
> При выборе DBCatalogManager и последующем рестарте сервиса Trino ADCM предлагает автоматически создать необходимые сущности в ADPG (БД, пользователь, роли).

После активации DBCatalogManager новые каталоги можно создавать с помощью команды `CREATE CATALOG`, а сервис Trino будет сохранять конфигурацию каталогов в БД, указанной в свойстве ***Postgres JDBC URL*** (БД по умолчанию — `catalog`).

DBCatalogManager сохраняет конфигурации каталогов в таблицах в схеме `trino`. Например, чтобы увидеть эти таблицы с помощью psql, используйте команду:

```
\c catalog
\dt trino.*
```

Вывод:

```
                   List of relations
 Schema |              Name              | Type  |  Owner
--------+--------------------------------+-------+---------
 trino  | catalog                        | table | catalog
 trino  | catalog_property               | table | catalog
 trino  | catalog_scheduler_job_launch | table | catalog
```

Настройка

ADCM предоставляет список конфигурационных параметров для настройки DBCatalogManager. Эти параметры приведены ниже.

Trino catalog management

Параметр	Описание	Значение по умолчанию
catalog.management	Тип менеджера каталогов	static
Postgres JDBC URL	URL для JDBC-подключения к хранилищу каталогов Postgres	jdbc:postgresql://{{ groups['adpg.adpg'][0] | d(omit) }}:5432/catalog
Postgres username	Имя пользователя хранилища каталогов Postgres для JDBC-подключения	catalog
Postgres password	Пароль для JDBC-подключения к хранилищу каталогов Postgres	—
Enable cleanup	Определяет, включена ли очистка каталогов старых версий	true
Encryption password	Пароль для шифрования свойств	—
RegEx pattern	Шаблон регулярного выражения для определения свойств, подлежащих шифрованию	—
Cleanup old version catalogs	Интервал очистки каталогов старых версий. Минимальное значение - - `10d`	30d
Time to live of old version catalogs	Время жизни (TTL) каталогов старых версий. Минимальное значение — `1d`	365d

Версионирование и удаление каталогов

Выполнение команды `DROP CATALOG` для DBCatalogManager не удаляет данные каталога мгновенно, а помечает каталог как неактивный. Такой каталог не будет отображаться при последующих операциях `SHOW CATALOGS`.

Неактивные каталоги могут храниться в базе данных в течение определенного периода времени, который можно указать с помощью параметра ***Time to live of old version catalogs*** в настройках ADCM. После этого каталоги старше указанного time-to-live значения удаляются планировщиком, который регулярно запускает процесс очистки через определенный интервал (свойство ***Cleanup old version catalogs***).

Шифрование

DBCatalogManager позволяет шифровать определенные поля конфигурации перед записью в базу данных. Чтобы указать поле, которое необходимо зашифровать, используйте параметр ***RegEx pattern*** в ADCM. В значении параметра необходимо указать регулярное выражение, соответствующее одному или нескольким ключам свойств конфигурации каталога, которые необходимо зашифровать. Например, для шифрования паролей можно использовать следующее выражение:

```
.*password
```

Миграция

Существующие конфигурации каталогов, хранящиеся в файлах *.properties*, можно перенести в базу данных ADPG. Для этого используйте команду:

```
MIGRATE CATALOG [IF NOT EXISTS] ALL | <catalog name>
```

Ниже показаны примеры миграции одного каталога, а также всех доступных каталогов:

```
MIGRATE CATALOG IF NOT EXISTS test_trino_catalog
MIGRATE CATALOG IF NOT EXISTS ALL
```

Пример подключения к ADPG

В следующем примере показано использование DBCatalogManager с сервисом ADPG, работающим в том же кластере ADH. Сценарий предполагает, что оба сервиса Trino и ADPG используют настройки по умолчанию, что позволяет установить соединение без каких-либо манипуляций на стороне ADPG. Если для подключения к ADPG необходимо использовать другого пользователя, пароль или базу данных, укажите их с помощью соответствующих настроек.

1. Активируйте DBCatalogManager, используя параметр Trino ***catalog.management*** в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Components*** → ***Trino Coordinator*** → ***Trino catalog management***) и перезапустите сервис Trino. При выполнении действия ***Restart*** убедитесь, что выбрана опция ***Init DB credentials***.

2. Создайте тестовый каталог Trino. Для этого подключитесь к Trino Coordinator и выполните команду:

   ```
   CREATE CATALOG test_trino_catalog USING memory
   WITH ("memory.max-data-per-node"='128MB');
   ```

3. Убедитесь, что каталог создан:

   ```
   SHOW CATALOGS;
   ```

   Вывод:

   ```
   information_schema
   test_trino_catalog
   ```

4. Проверьте базу данных ADPG на наличие новых таблиц. Например, с помощью psql запросите таблицы, принадлежащие схеме `trino`:

   ```
   \c catalog
   SELECT * FROM trino.catalog;
   ```

   Пример вывода:

   ```
    id |        name        | connector_name |         date_from          | date_to | is_active
   ----+--------------------+----------------+----------------------------+---------+-----------
     1 | test_trino_catalog | memory         | 2025-06-03 21:12:11.193509 |         | t
   ```

5. Чтобы перенести в базу данных существующий каталог, хранящийся в файле *.properties* (например, */etc/trino/catalogs/hive-adh.properties*), выполните команду:

   ```
   MIGRATE CATALOG "hive-adh"
   ```

   Убедитесь, что каталог появился в списке доступных:

   ```
   SHOW CATALOGS;
   ```

   Вывод:

   ```
   Catalog           |
   ------------------+
   hive-adh          |
   system            |
   test_trino_catalog|
   ```

   Проверьте информацию о каталоге в ADPG:

   ```
   \c catalog
   SELECT * FROM trino.catalog;
   ```

   Вывод:

   ```
    id |        name        | connector_name |         date_from          | date_to | is_active
   ----+--------------------+----------------+----------------------------+---------+-----------
     1 | test_trino_catalog | memory         | 2025-06-03 21:12:11.193509 |         | t
     2 | hive-adh           | hive           | 2025-06-03 22:10:43.222419 |         | t
   ```

Управление отказоустойчивостью в Trino

Константин Алпашкин

Содержание

Назад к содержанию

- Активация FTE
- Exchange manager и данные обмена
- Настройки FTE
  - Буферизация в HDFS, Ozone и в локальной FS

По умолчанию, если на хосте Trino заканчиваются ресурсы при выполнении запроса или Trino worker выходит из строя по какой-то другой причине, текущий запрос завершается ошибкой и его необходимо перезапустить вручную. Как правило, чем дольше выполняется запрос, тем больше вероятность, что такой запрос приведет к ошибке.

Сервис Trino поддерживает отказоустойчивый режим работы (Fault-tolerant execution, FTE), который позволяет автоматически перезапускать запросы с ошибками или перезапускать отдельные их подзадачи (tasks). Если режим FTE включен и запрос падает с ошибкой, Trino сохраняет промежуточные данные в файловой системе. Используя эти промежуточные данные, другие компоненты Trino worker могут повторно запустить выполнение запроса.

NOTE

- Поддержка режима FTE зависит от коннектора. Список коннекторов Trino, поддерживающих FTE, доступен в документации Trino.
- Механизм FTE не применяется к некорректно составленным запросам или другим ошибкам пользователя. Например, Trino не будет повторно выполнять запрос, завершившийся с ошибкой из-за некорректного SQL-синтаксиса или несовместимых типов столбцов.

Активация FTE

По умолчанию режим FTE отключен для сервиса Trino. Активировать режим FTE можно с помощью ADCM (***Clusters → <clusterName> → Services → Trino → Primary Configuration → Fault-tolerant execution***). А именно, следует указать конфигурационный параметр retry-policy. Данный параметр принимает следующие значения:

- `NONE` — отключает режим FTE (значение по умолчанию).
- `QUERY` — если Trino worker завершает выполнение запроса с ошибкой, Trino повторно выполняет весь запрос (query). Данная политика рекомендуется, когда большинство запросов Trino небольшие.
- `TASK` — если Trino worker завершает выполнение запроса с ошибкой, Trino повторно выполняет отдельные задачи (tasks). Данный режим оптимален для больших пакетных запросов, однако может привести к повышенной задержке при большом количестве маленьких запросов.

ПРИМЕЧАНИЕ

Выбор политики `TASK` автоматически обновляет некоторые релевантные параметры конфигурации для обеспечения наилучшей отказоустойчивости кластера Trino. При выборе политики `TASK` настоятельно рекомендуется установить параметр `task.low-memory-killer.policy=total-reservation-on-blocked-nodes`. В противном случае при дефиците памяти на хостах с Trino worker может потребоваться ручное завершение запросов.

Exchange manager и данные обмена

Trino Exchange manager — это модуль, отвечающий за передачу данных обмена (exchange data) между хостами Trino worker при выполнении распределенного запроса. Если FTE включен, этот компонент также отвечает за буферизацию (spooling) данных обмена в файловой системе, например HDFS, Ozone или в локальной файловой системе. При активации FTE необходимо выбрать подходящую реализацию Exchange manager для записи промежуточных данных в нужную файловую систему.

Настройки FTE

Сервис Trino предоставляет несколько групп настроек в ADCM, которые используются для настройки поведения FTE. К ним относятся:

- Fault-tolerant execution. Используя эти настройки, можно определить поведение для повторных запусков запросов, включая политики, интервалы, параметры шифрования и так далее.
- Exchange manager. В этих настройках регулируются параметры промежуточных данных, которые генерирует Exchange manager (тип хранилища, пути, разрешения, размер буфера и так далее).

Буферизация в HDFS, Ozone и в локальной FS

По умолчанию сервис Trino предоставляет две реализации Exchange manager, которые могут сохранять промежуточные данные обмена в HDFS, Ozone, а также в локальной файловой системе.

При включении режима FTE сервис Trino автоматически определяет, какой тип хранилища используется в ADH-кластере (HDFS или Ozone) и обновляет все необходимые настройки для работы с данным типом хранилища.

Для записи данных обмена в HDFS установите параметр `exchange-manager.name=hdfs` и укажите путь для хранения данных с помощью `exchange.base-directories`, используя схему `hdfs:///`.

Чтобы сохранять данные обмена в Ozone, установите параметр `exchange-manager.name=hdfs` и укажите путь к хранилищу, используя схему `ofs:///`.

ПРИМЕЧАНИЕ

Одновременное использование путей HDFS и Ozone в поле `exchange.base-directories` недопустимо.

Чтобы сохранять данные обмена в локальной файловой системе, установите параметр `exchange-manager.name=local` и укажите путь для хранения данных с помощью параметра `exchange.base-directories` (используйте схему `file:///`).

Сервис Trino может автоматически создавать директории для хранения промежуточных данных. Используя настройки FTE, можно указать расположение, разрешения для директорий и так далее.

Передача учетных данных в Trino

Елена Костюченко

Содержание

Назад к содержанию

Обзор

Включение credentials propagation

Пример

Обзор

Передача учетных данных (credentials propagation) в Trino позволяет аутентифицироваться от имени текущего пользователя через коннекторы, такие как ADB, PostgreSQL, ClickHouse и другие. Это обеспечивает большую безопасность при доступе к данным.

В этой статье описывается, как включить и настроить передачу учетных данных для Trino в кластерах Arenadata.

> **ПРИМЕЧАНИЕ**
>
> Эта функция работает для JDBC-коннекторов и при использовании HTTP-аутентификации с паролем (свойство `http-server.authentication.type=PASSWORD`).

Включение credentials propagation

Когда значение параметра `arenadata.http-server.authentication.password.populate-extra-credentials` установлено в `true`, Trino автоматически добавляет имя пользователя и пароль из текущей сессии в extra credentials. Используемый коннектор считывает эти учетные данные и аутентифицирует запрос от имени пользователя.

Чтобы включить передачу учетных данных с помощью ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите в ***Services*** → ***Trino*** → ***Components*** → ***Trino Coordinator/Trino Worker*** и включите опцию ***Show advanced***.
3. Откройте раздел ***Custom node.properties*** и нажмите ***Add property***.
4. Введите ***arenadata.http-server.authentication.password.populate-extra-credentials*** в качестве имени поля и `true` — в качестве значения.
5. Сохраните конфигурацию, нажав ***Save***, и перезапустите компонент через ***Actions*** → ***Restart***.

Для каждого коннектора, который должен использовать передаваемые учетные данные, добавьте следующие свойства в конфигурационный файл коннектора:

```
user-credential-name=arenadata.username
password-credential-name=arenadata.password
```

Дополнительную информацию о настройке коннекторов можно получить в статье Обзор коннекторов в Trino.

Пример

Чтобы проверить, что передача учетных данных работает корректно:

1. Подключитесь к Trino, используя CLI. Пример команды для подключения:

```
$ ./trino https://<coordinator_host>:18188 --user=<username> --password
```

2. Введите пароль.
3. Выполните запрос к коннектору, который использует передаваемые учетные данные:

```
SHOW TABLES FROM adb.default;
```

Коннектор выполнит аутентификацию, используя предоставленные учетные данные пользователя.

Высокая доступность Trino Coordinator

Константин Алпашкин

Содержание

Назад к содержанию

Компонент Haproxy Trino

Настройка Haproxy Trino

Логирование

Статистика Haproxy Trino

Особенности Kerberos

Сервис Trino поддерживает режим высокой доступности (High Availability, HA), при котором возможно использование нескольких компонентов Trino Coordinator. Если один Trino Coordinator становится недоступным, резервные Trino Coordinator включаются в работу, не прерывая работу Trino-кластера.

Компонент Haproxy Trino

Переключение между несколькими Trino Coordinator осуществляется с помощью компонента Haproxy Trino, который выполняет роль обратного прокси (reverse proxy) между клиентами Trino и серверами Trino Coordinator в ADH. Данный компонент основан на проекте HAProxy и используется для обеспечения высокой доступности, балансировки нагрузок и маршрутизации HTTP/TCP-трафика между серверами.

ВАЖНО

- При подключении к Trino-сервису в режиме HA клиент Trino (Trino CLI, JDBC или браузер) должен подключаться к компоненту Haproxy Trino, а не напрямую к Trino Coordinator. Актуальная ссылка на веб-интерфейс и строка подключения JDBC доступны в ADCM (***Clusters*** → ***<ADH_cluster>*** → ***Services*** → ***Trino*** → ***Info***).
- Компоненты Haproxy Trino и Trino Coordinator необходимо устанавливать на разные хосты во избежание конфликта портов.

Кластер ADH может содержать один или несколько компонентов Haproxy Trino. Установка нескольких компонентов Haproxy Trino может использоваться для реализации полноценного режима высокой доступности на сетевом уровне.

Настройка Haproxy Trino

При запуске сервиса Trino в режиме HA ADCM автоматически настраивает компонент Haproxy Trino для маршрутизации входящих запросов к компонентам Trino Coordinator, не требуя каких-либо ручных настроек. Дополнительно вы можете настроить поведение Haproxy Trino, как описано далее.

Основным файлом настроек для Haproxy Trino является */etc/adh-haproxy/conf/haproxy-trino.cfg* — файл конфигурации HAProxy, расположенный на хостах ADH, где установлен компонент Haproxy Trino.

Пример haproxy-trino.cfg

```
#---------------------------------------------------------------------
# Global settings
#---------------------------------------------------------------------
global
    log /dev/log    local0
    log 127.0.0.1:514  local0

    chroot      /var/lib/adh-haproxy/trino
    maxconn     1024
    user        haproxy
    group       haproxy
    daemon

    ssl-server-verify none

#---------------------------------------------------------------------
# common defaults that all the 'listen' and 'backend' sections will
# use if not designated in their block
#---------------------------------------------------------------------
defaults
    mode                    http
    log                     global
    option                  httplog
    option                  dontlognull
    option http-server-close
    option forwardfor       except 127.0.0.0/8
    option                  redispatch
    retries                 3
    timeout http-request    10s
    timeout queue           1m
    timeout connect         10s
    timeout client          1m
    timeout server          1m
    timeout http-keep-alive 10s
    timeout check           10s

listen stats
    bind *:7001
    stats enable
    stats uri /

frontend trino_http_in
    bind *:18188
    default_backend trino_http_servers

backend trino_http_servers
    balance first

    server trino-coordinator0 ka-adh-1.ru-central1.internal:18188 check inter 1s fall 2 rise 5
    server trino-coordinator1 ka-adh-3.ru-central1.internal:18188 check inter 1s fall 2 rise 5 backup
```

Данный файл не следует редактировать вручную — вместо этого для настройки поведения Haproxy Trino используйте Jinja-шаблон в ADCM (***Clusters*** → ***<ADHclusterName>*** → ***Services*** → ***Trino*** → ***Components*** → ***Haproxy Trino*** → ***Configuration***). При каждом старте сервиса Trino файл *haproxy-trino.cfg* генерируется на основе этого шаблона.

Ниже представлен дефолтный шаблон *haproxy-trino.cfg* с комментариями к основным блокам настроек.

```
{% set ssl_sert = '' %}
{%- if trino_ssl_enable -%}
    {% set ssl_sert = haproxy_conf_ssl_vars %}
{% endif %}
#---------------------------------------------------------------------
# Global settings
#---------------------------------------------------------------------
global ①
    log /dev/log    local0
    log 127.0.0.1:514  local0

    chroot      /var/lib/adh-haproxy/trino
    maxconn     1024
    user        haproxy
    group       haproxy
    daemon

    ssl-server-verify none

#---------------------------------------------------------------------
# common defaults that all the 'listen' and 'backend' sections will
# use if not designated in their block
#---------------------------------------------------------------------
defaults ②
    mode                    http
    log                     global
    option                  httplog
    option                  dontlognull
    option http-server-close
    option forwardfor       except 127.0.0.0/8
    option                  redispatch
    retries                 3
    timeout http-request    10s
    timeout queue           1m
    timeout connect         10s
    timeout client          1m
    timeout server          1m
    timeout http-keep-alive 10s
    timeout check           10s

listen stats ③
    bind *:7001 {{ ssl_sert }}
    stats enable
    stats uri /

frontend trino_http_in ④
    bind *:{{ roles_trino_vars_component_ports['haproxy']['web'] }} {{ ssl_sert }}
    default_backend trino_http_servers

backend trino_http_servers ⑤
    balance first

{% for host in haproxy_conf_hosts %}
    {{ host }}
{% endfor %}
```

① Глобальные параметры HAProxy, например, параметры логирования, изменения корневой директории, проверка сертификатов и прочие.

② Параметры HAProxy по умолчанию, такие как режим работы, логирование и различные тайм-ауты.

③ Активирует сбор статистики и мониторинг ядра HAProxy в реальном времени. Определяет порт и URI эндпойнта статистики (*http://<Haproxy_Trino_host>:7001/* по умолчанию).

④ Список прослушиваемых сокетов для входящих соединений.

⑤ Список серверов Trino Coordinator, куда компонент Haproxy Trino будет направлять входящие запросы. В данном шаблоне цикл `{% for host in … %}` генерирует список серверов Trino Coordinator, доступных в ADH-кластере. Параметр `balance first` указывает, что первый сервер используется в качестве активного (основного) Trino Coordinator, а другие серверы используются в качестве резервных, если первый становится недоступным.

Подробное описание настроек HAProxy доступно в справочной документации HAProxy.

Логирование

По умолчанию компонент Haproxy Trino логирует собственную активность и сохраняет логи в директории */var/log/adh-haproxy/*. Настройки логирования указываются в *haproxy-trino.cfg*. Конфигурация по умолчанию обеспечивает двойное логирование, а именно:

```
global
    log /dev/log    local0 ①
    log 127.0.0.1:514  local0 ②
```

① Направляет логи в локальный сокет syslog (*/dev/log*) для обработки системным syslog-демоном.

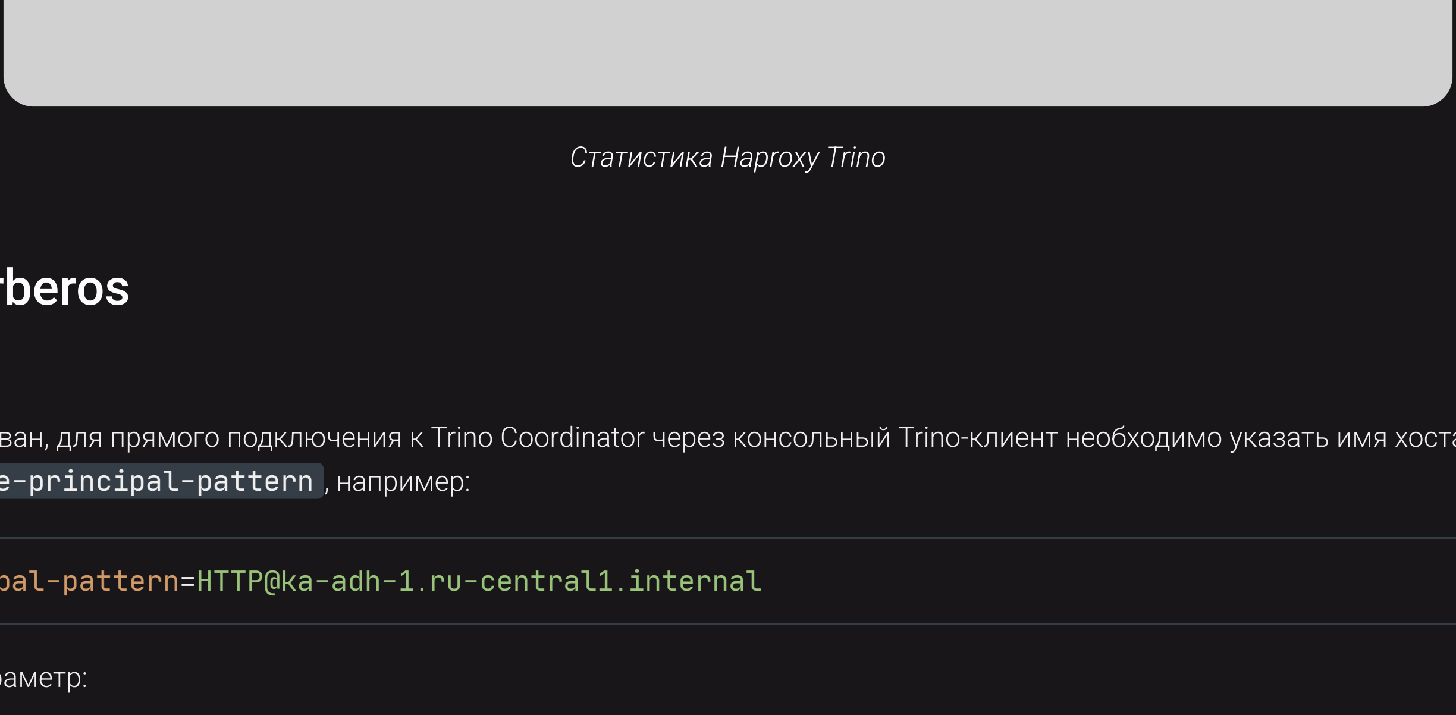
② Направляет логи syslog-демону, слушающему порт `514` на localhost (например, rsyslog или syslog-ng). По умолчанию rsyslog/syslog-ng сохраняют лог-файлы в */var/log/adh-haproxy/*.

РЕКОМЕНДАЦИЯ

Компонент Haproxy Trino имеет соответствующие настройки для rsyslog/syslog-ng в ADCM (***/etc/syslog-ng/conf.d/haproxy-trino.conf***, ***/etc/rsyslog.d/haproxy-trino.conf***). С их помощью можно изменить шаблоны лог-сообщений, фильтры, расположение лог-файлов и так далее.

Статистика Haproxy Trino

Компонент Haproxy Trino предоставляет статистику о своей работе. Сбор статистики включен по умолчанию (блок настроек `listen stats` в *haproxy-trino.cfg*) и данные доступны по следующему эндпойнту: *http://<Haproxy_Trino_host>:7001/*.



HAProxy version 3.2.0, released 2025/05/28

Statistics Report for pid 2081

> General process information

Статистика Haproxy Trino

Особенности Kerberos

- Если кластер керберизирован, для прямого подключения к Trino Coordinator через консольный Trino-клиент необходимо указать имя хоста Trino Coordinator в параметре `krb5-service-principal-pattern`, например:

  ```
  krb5-service-principal-pattern=HTTP@ka-adh-1.ru-central1.internal
  ```

  Для JDBC используйте параметр:

  ```
  KerberosServicePrincipalPattern=HTTP@ka-adh-1.ru-central1.internal
  ```

- Прямое подключение к веб-интерфейсу Trino Coordinator через браузер (Chrome, Firefox) или cURL не поддерживается.

Обзор коннекторов в Trino

Константин Алпашкин

Содержание

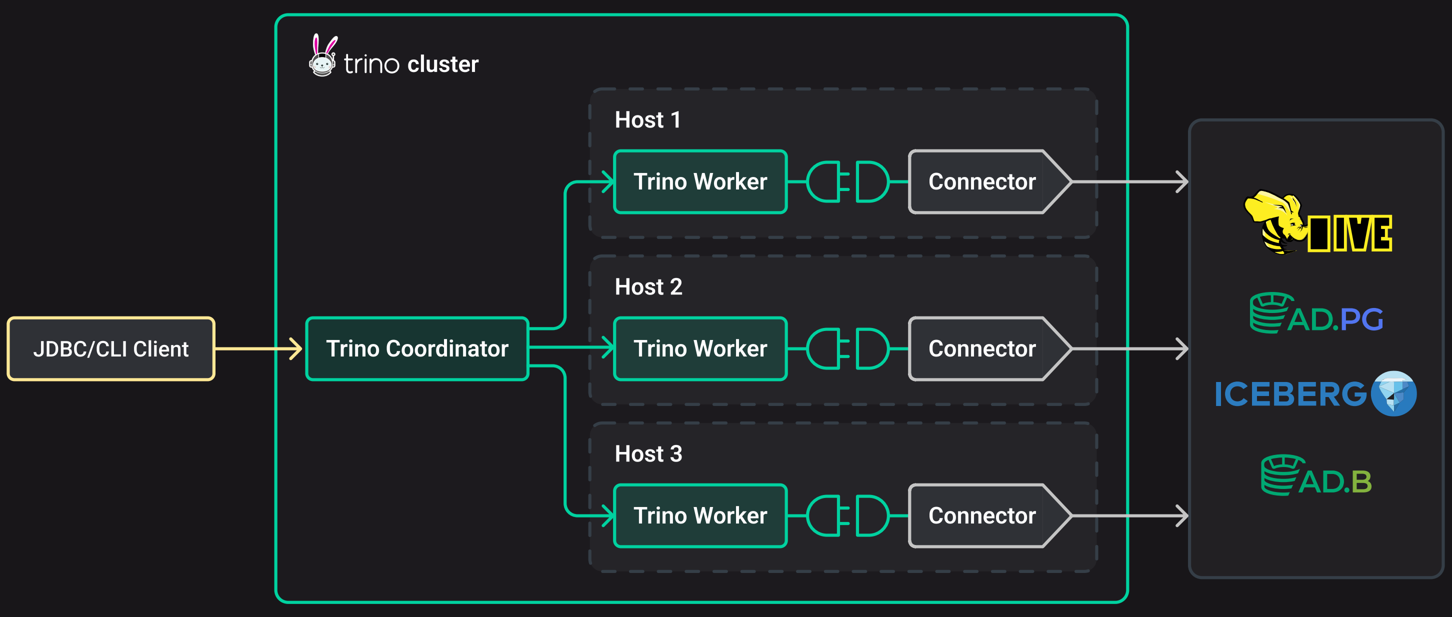
Назад к содержанию

Обзор
Поддерживаемые коннекторы
Настройка коннекторов
Создание пользовательского коннектора

Обзор

Коннекторы в Trino — это Java-плагины, которые позволяют Trino взаимодействовать с различными источниками данных, такими как озера данных Hadoop, хранилища Hive, таблицы Apache Iceberg, базы данных PostgreSQL и многие другие. Trino получает доступ к источнику данных с помощью коннектора, который настраивается в каталоге. Коннекторы отвечают за коммуникацию между Trino и целевым источником данных, а также транслируют ANSI SQL-команды Trino в команды, соответствующие целевому источнику данных. Один коннектор можно использовать для доступа к одному или нескольким источникам данных одного типа.

Роль коннекторов во взаимодействии с Trino представлена на следующей высокоуровневой схеме.



Коннекторы Trino

Поддерживаемые коннекторы

Полный список коннекторов, поддерживаемых Trino, доступен в документации Trino.

Настройка коннекторов

Коннекторы Trino настраиваются с помощью конфигурационных файлов каталога. Коннектор, который должен использоваться для доступа к источнику данных каталога, указывается в конфигурационном файле *<catalog_name>.properties* с помощью свойства `connector.name`. Например:

```
connector.name=hive
...
```

ПРИМЕЧАНИЕ

Один и тот же коннектор можно использовать для работы с несколькими каталогами. Например, если у вас есть два хранилища Hive с отдельными базами данных под Metastore, вы можете настроить два отдельных каталога, которые используют один и тот же коннектор Hive (`connector.name=hive`), но содержат разные параметры соединения для каждого каталога.

Конфигурационные файлы каталогов могут также содержать дополнительные параметры, специфичные для конкретного типа коннекторов, для тонкой настройки взаимодействия с источником данных. Например:

```
hive.metastore.thrift.impersonation.enabled=True
iceberg.catalog.type=hive_metastore
iceberg.compression-codec=SNAPPY
iceberg.file-format=PARQUET
...
```

Более подробную информацию о конфигурационных свойствах каждого коннектора можно найти в соответствующем разделе документации Trino.

Создание пользовательского коннектора

Помимо списка коннекторов, поддерживаемых ADH-сервисом Trino "из коробки", вы можете создавать собственные коннекторы для работы с любым источником данных. Даже если источник данных не использует такие сущности, как таблицы, столбцы, строки, партиции и прочие, адаптировав источник данных к Trino API с помощью коннектора, вы сможете писать запросы на SQL и получать результаты из этого источника данных.

С технической точки зрения новые коннекторы Trino добавляются в виде плагинов. Коннектор должен реализовывать service provider interface (SPI), который позволяет Trino взаимодействовать с источником данных. В каждом коннекторе должен быть реализован следующий функционал API:

- Операции получения метаданных о таблицах, представлениях и схемах.
- Операции по созданию логических единиц партиционирования данных, позволяющие Trino распараллеливать операции чтения и записи.
- Генераторы и ресиверы данных, которые преобразуют исходные данные в in-memory объекты, обрабатываемые движком Trino.

Реализовав SPI для коннектора Trino, можно использовать стандартные операции Trino для подключения к любому источнику данных и выполнения операций над любым источником данных. Реализация конкретного коннектора должна учитывать все детали, связанные с особенностями источника данных, с которым взаимодействует коннектор.

Например, базовый Trino-коннектор, поддерживающий чтение данных из источника данных, должен реализовать определенный набор методов, включая `listTables()`. Движок Trino вызывает этот метод для всех коннекторов, чтобы получить список доступных таблиц в схеме/базе данных. Движок Trino не располагает информацией, как именно каждый конкретный коннектор получает список таблиц. Некоторые коннекторы могут получить этот список из информационной схемы, другие могут запросить информацию из хранилища метаданных, а некоторые получают список через API источника данных. Вся логика получения списка таблиц для конкретного источника данных реализуется только в коннекторе.

Подробности и примеры создания пользовательских коннекторов доступны в следующих разделах документации Trino:

- Пример HTTP-коннектора
- Пример коннектора JDBC

Также вы можете использовать исходный код существующих коннекторов Trino в GitHub.

Каталог Iceberg в Trino

Константин Алпашкин

Содержание

Назад к содержанию

- Активация каталога Iceberg
- Поддерживаемые форматы таблиц
- Сопоставление типов данных
- Свойства конфигурации каталога Iceberg
- Поддержка SQL
  - Управление схемой
  - Примеры DDL
  - Примеры DML

Сервис Trino поставляется с преднастроенным каталогом для работы с таблицами Iceberg. Данный каталог использует коннектор Iceberg и "из коробки" готов к работе с Iceberg-таблицами, которые хранятся в вашем ADH-кластере. Встроенный каталог Iceberg использует Hive Metastore в качестве реализации Iceberg-каталога. То есть при выполнении запроса к Iceberg-таблице Trino запрашивает метаданные о таблице из Hive Metastore.

Параметры конфигурации встроенного Iceberg-каталога генерируются автоматически и обеспечивают доступ к Hive Metastore, а также к HDFS-директориям сервиса Hive. Имя встроенного каталога — `iceberg-adh`. ADH формирует каталог автоматически и генерирует файл *iceberg-adh.properties* в директории */etc/trino/conf/catalog/*, если:

- В ADH-кластере установлен сервис Hive. Если сервис Hive был установлен после Trino, сервис Trino необходимо перезапустить.
- Конфигурация Iceberg-каталога активна.

Активация каталога Iceberg

По умолчанию встроенный Iceberg-каталог сервиса Trino отключен. Для его активации используйте ADCM. Для этого включите опцию ***Iceberg configuration*** в сервисе Trino (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Configuration***), а затем перезапустите сервис.

После активации следует проверить доступность встроенного каталога. Для этого подключитесь к Trino Coordinator (используя Trino CLI или SQL-клиент, например, DBeaver) и выполните команду:

```
SHOW CATALOGS;
```

В списке появится встроенный Iceberg-каталог:

```
Catalog    |
-----------+
iceberg-adh|
system     |
```

РЕКОМЕНДАЦИЯ

Актуальная JDBC-строка для подключения к сервису Trino доступна на странице конфигурации сервиса в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Info***).

Поддерживаемые форматы таблиц

Iceberg-коннектор сервиса Trino может работать с таблицами Iceberg в следующих форматах:

- Parquet
- ORC
- Avro

Сопоставление типов данных

При чтении и записи данных в Iceberg-таблицы Trino автоматически преобразует определенные типы данных, чтобы соответствовать спецификации Iceberg и форматам Avro/ORC/Parquet. Правила преобразования могут отличаться в зависимости от направления конверсии.

Информация о преобразовании типов данных Iceberg в форматы Parquet/ORC/Avro доступна в спецификации Iceberg:

- Iceberg → Avro
- Iceberg → ORC
- Iceberg → Parquet

Преобразование типов Iceberg в Trino

Тип данных Iceberg	Тип данных Trino
BOOLEAN	BOOLEAN
INT	INTEGER
LONG	BIGINT
FLOAT	REAL
DOUBLE	DOUBLE
DECIMAL(p,s)	DECIMAL(p,s)
DATE	DATE
TIME	TIME(6)
TIMESTAMP	TIMESTAMP(6)
TIMESTAMPTZ	TIMESTAMP(6) WITH TIME ZONE
STRING	VARCHAR
UUID	UUID
BINARY	VARBINARY
FIXED (L)	VARBINARY
STRUCT(...)	ROW(...)
LIST(e)	ARRAY(e)
MAP(k,v)	MAP(k,v)

Преобразование типов Trino в Iceberg

Тип данных Trino	Тип данных Iceberg
BOOLEAN	BOOLEAN
INTEGER	INT
BIGINT	LONG
REAL	FLOAT
DOUBLE	DOUBLE
DECIMAL(p,s)	DECIMAL(p,s)
DATE	DATE
TIME(6)	TIME
TIMESTAMP(6)	TIMESTAMP
TIMESTAMP(6) WITH TIME ZONE	TIMESTAMPTZ
VARCHAR	STRING
UUID	UUID
VARBINARY	BINARY
ROW(...)	STRUCT(...)
ARRAY(e)	LIST(e)
MAP(k,v)	MAP(k,v)

Свойства конфигурации каталога Iceberg

Iceberg-каталог сервиса Trino поддерживает множество конфигурационных свойств для тонкой настройки взаимодействия с таблицами Iceberg. Актуальный список параметров доступен в документации Trino. Используя ADCM, можно установить новые или переопределить дефолтные параметры. Для этого используйте раздел настроек ***Custom Iceberg configuration*** в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Configuration***).

Поддержка SQL

ВАЖНО

Чтобы Trino мог записывать данные в Iceberg-таблицы, при подключении к Trino (например, через Trino CLI или DBeaver) необходимо указывать имя пользователя, соответствующее Unix-пользователю на хосте ADH, у которого имеются достаточные права для записи в HDFS-директории хранилища Hive. В противном случае вы можете наблюдать следующую ошибку:

```
Failed to create external path hdfs://adh/apps/hive/warehouse/trino_demo_schema1.db for database
trino_demo_schema1. This may result in access not being allowed if the StorageBasedAuthorizationProvider is
enabled: null
```

Управление схемой

Схемы в Trino используются для группировки нескольких таблиц/представлений и эквивалентны базам данных в Hive. Ниже показаны базовые команды для работы со схемами Trino:

```
CREATE SCHEMA "iceberg-adh"."trino_demo_schema_iceberg"; ①
CREATE SCHEMA "iceberg-adh"."trino_demo_schema_iceberg_with_loc"
    WITH (location='/user/hive/warehouse/some_custom_path'); ②
SHOW SCHEMAS;
USE "iceberg-adh"."trino_demo_schema_iceberg"; ③
DROP SCHEMA "trino_demo_schema_iceberg"; ④
```

① Создание новой схемы Trino (базы данных Hive). `iceberg-adh` — имя встроенного Iceberg-каталога Trino. Обратите внимание на двойные кавычки, используемые для обрамления идентификаторов.

② Создание схемы (базы данных Hive) с указанием директории хранения в HDFS.

③ Выбор схемы для сессии. Все последующие SQL-команды Trino будет выполнять на указанной базе данных Hive в рамках текущей сессии.

④ Удаление непустой схемы запрещено. Чтобы удалить схему, необходимо сначала удалить все вложенные таблицы.

Для выполнения запросов к произвольным таблицам, которые принадлежат разным схемам, можно указать каталог, схему и имя таблицы в SQL-запросе, например:

```
SELECT * FROM "iceberg-adh"."trino_demo_schema_iceberg"."transactions"
```

РЕКОМЕНДАЦИЯ

Используя Trino UI, можно просматривать и анализировать SQL-запросы, отправленные в Trino.

Примеры DDL

Trino поддерживает большинство DDL-команд (`CREATE`, `DROP`, `ALTER` и т.д.) для управления таблицами и материализованными представлениями Iceberg. Trino также позволяет устанавливать свойства таблиц Iceberg, которые задают метаданные для таблиц. Эти свойства можно назначить таблице Iceberg, используя предложение `WITH`, как показано далее в примерах. Ниже представлены примеры DDL-команд с использованием каталога Iceberg.

```
CREATE TABLE "iceberg-adh"."trino_demo_schema_iceberg"."transactions" ( ①
    txn_id BIGINT NOT NULL,
    txn_value DOUBLE,
    acc_id BIGINT,
    txn_date DATE,
    comment VARCHAR
);

CREATE TABLE "iceberg-adh"."trino_demo_schema_iceberg"."transactions_pttnd" ( ②
    txn_id BIGINT,
    txn_value DOUBLE,
    acc_id BIGINT,
    txn_date DATE,
    comment VARCHAR
) WITH (
    format = 'PARQUET',
    partitioning = ARRAY['acc_id']
);

CREATE TABLE "iceberg-adh"."trino_demo_schema_iceberg"."transactions_pttnd_ts" ( ③
    txn_id BIGINT,
    txn_value DOUBLE,
    acc_id BIGINT,
    txn_date DATE,
    comment VARCHAR
) WITH (
    partitioning = ARRAY['month(txn_date)', 'bucket(acc_id, 5)']
);

CREATE TABLE "iceberg-adh"."trino_demo_schema_iceberg"."transactions_sorted" ( ④
    txn_id BIGINT,
    txn_value DOUBLE,
    acc_id BIGINT,
    txn_date DATE,
    comment VARCHAR
) WITH (
    sorted_by = ARRAY['txn_value']
);

CREATE TABLE "iceberg-adh"."trino_demo_schema_iceberg"."transactions_orc_as" ⑤
WITH (
    format = 'ORC',
    orc_bloom_filter_columns = ARRAY['acc_id','txn_date'],
    orc_bloom_filter_fpp = 0.1
) AS
    SELECT * FROM "iceberg-adh"."trino_demo_schema_iceberg"."transactions"
    WHERE txn_id < 10;

ALTER TABLE "iceberg-adh"."trino_demo_schema_iceberg"."transactions_pttnd" ⑥
SET PROPERTIES partitioning = ARRAY['acc_id', 'txn_date'];

ALTER TABLE "iceberg-adh"."trino_demo_schema_iceberg"."transactions" ⑦
EXECUTE expire_snapshots(retention_threshold => '1d');


CREATE MATERIALIZED VIEW "iceberg-adh"."trino_demo_schema_iceberg"."transactions_mv"  ⑧
WITH (
    partitioning = ARRAY['txn_date']
) AS
```

① Создание Iceberg-таблицы (`table_type=ICEBERG`) в хранилище Hive. Обратите внимание, что Trino принимает запросы в формате ANSI SQL с соответствующими типами данных, в то время как спецификация Iceberg может использовать иные типы данных. Например, при выполнении данного запроса тип `VARCHAR` будет приведен к `STRING`.

② Создание партиционированной Iceberg-таблицы в формате Parquet. Обратите внимание на синтаксис `ARRAY['col1', 'col2']` для указания партиций.

③ Создание таблицы, которая будет разбита на партиции по месяцам значения `txn_date`, а также по хешу значения `acc_id` (5 партиций).

④ Создание таблицы, данные которой сохраняются в файлы Iceberg отсортированными по столбцу `txn_value`. Сортировка данных таблицы может быть выгодна с точки зрения производительности при последующих операциях чтения.

⑤ Использование синтаксиса CTAS (`CREATE TABLE AS`) при создании Iceberg-таблицы в формате ORC. Также устанавливаются ORC-параметры для bloom-фильтров.

⑥ Добавление партиций в таблицу с помощью команды `ALTER TABLE SET`. Больше информации о свойствах таблиц Iceberg, которые можно указать с помощью `ALTER TABLE SET`, доступно в документации Trino.

⑦ Удаление всех снепшотов, метаданных и файлов данных таблицы Iceberg, возраст которых превышает указанное значение (в данном примере — 7 дней). Периодическое выполнение этой команды позволяет очистить ненужные файлы исторических данных и поддерживать компактность метаданных таблицы. Информацию о других функциях Iceberg (`remove_orphan_files`, `drop_extended_stats` и так далее), которые можно вызвать с помощью `ALTER TABLE EXECUTE`, можно найти в разделе документации Trino.

⑧ Создание материализованного представления на основе выборки `SELECT`.

Больше информации о DDL-командах, поддерживаемых каталогом Iceberg в Trino, доступно в документации Trino.

Примеры DML

Trino поддерживает большинство DML-команд для Iceberg-таблиц, таких как `INSERT`, `UPDATE`, `DELETE`, `MERGE` и так далее. Ниже представлены примеры DML-операций с использованием каталога Iceberg.

```
SELECT * FROM "iceberg-adh"."trino_demo_schema_iceberg"."transactions_pttnd$partitions"; ①

SELECT * FROM "iceberg-adh"."trino_demo_schema_iceberg"."transactions_pttnd$snapshots"; ②

SELECT * FROM TABLE( ③
    system.table_changes(
      schema_name => "trino_demo_schema_iceberg",
      table_name => "transactions",
      start_snapshot_id => 3196536194725230246,
      end_snapshot_id => 248634242462154958
    )
  );

SELECT * FROM "iceberg-adh"."trino_demo_schema_iceberg"."transactions_pttnd"
FOR VERSION AS OF 2602443793554441031; ④

SELECT * FROM "iceberg-adh"."trino_demo_schema_iceberg"."transactions_pttnd" FOR TIMESTAMP AS OF TIMESTAMP '2025-01-17 01:41:38.000
+0300' ⑤

SELECT * FROM "iceberg-adh"."trino_demo_schema_iceberg"."transactions" txns_icb
JOIN "hive-adh"."trino_demo_schema_hive"."transactions" txns_hive
ON txns_icb.txn_id = txns_hive.txn_id; ⑥

DELETE FROM "iceberg-adh"."trino_demo_schema_iceberg"."transactions_pttnd"
WHERE acc_id = 1001; ⑦
```

① Получение подробной информации о партициях Iceberg-таблицы из таблицы метаданных `$partitions`. Аналогичным образом можно использовать и другие таблицы метаданных, например `$manifests`, `$files`, `$history` и так далее. Больше информации о работе с таблицами метаданных можно найти в документации Trino.

② Получение подробной информации о снепшотах, доступных для таблицы Iceberg.

③ Получение информации о row-level изменениях в двух версиях Iceberg-таблицы.

④ Time-travel выборка из таблицы Iceberg с указанием ID снепшота. Состояние таблицы, из которого следует считывать данные, определяется конкретным снепшотом. Такой запрос игнорирует все изменения, внесенные в таблицу после указанного снепшота.

⑤ Time-travel выборка с указанием временной метки. Состояние таблицы, из которого следует считывать данные, определяется снепшотом, ближайшим по времени создания к заданной временной метке. Такой запрос игнорирует все изменения, внесенные в таблицу после указанного времени.

⑥ Получение данных из двух каталогов Trino: Iceberg-каталога `iceberg-adh` и Hive-каталога `hive-adh`. Получив результаты выборки из каждого каталога, Trino применяет операцию `JOIN` и возвращает клиенту конечный набор результатов. Чтобы данный запрос работал корректно, оба каталога должны быть включены в сервисе Trino.

⑦ Удаление всей партиции, учитывая, что в предложении `WHERE` задан фильтр, полностью соответствующий партиции.

Каталог Hive в Trino

Константин Алпашкин

Содержание

Назад к содержанию

- Активация каталога Hive
- Поддерживаемые форматы Hive
- Параметры конфигурации Hive-каталога
- Поддержка SQL
  - Управление схемой
  - Примеры DDL
  - Примеры DML
- Оптимизация производительности

Сервис Trino поставляется с преднастроенным каталогом для работы с Hive. Данный каталог использует коннектор Hive и "из коробки" готов к работе с сервисом Hive, развернутым в ADH-кластере.

> Архитектура Trino-коннектора для Hive отличается от СУБД-ориентированных и других типов коннекторов. Основное отличие состоит в том, что Trino не задействует движок Hive для операций чтения/записи. То есть при выполнении запросов с помощью коннектора Hive Trino не вызывает никаких функций движка Hive. Вместо этого Trino напрямую обращается к Hive Metastore и задействует HDFS-клиент для выполнения операций над "сырыми" файлами Hive в HDFS. Из этого вытекает необходимость наличия компонента HDFS Client на каждом хосте, где установлен компонент Trino Worker.

Параметры конфигурации встроенного Hive-каталога генерируются автоматически и обеспечивают доступ к Hive Metastore, а также к HDFS-директориям сервиса Hive. Имя встроенного каталога Hive — `hive-adh`. ADH формирует каталог автоматически и генерирует файл *hive-adh.properties* в директории */etc/trino/conf/catalog/*, если:

- В ADH-кластере установлен сервис Hive. Если сервис Hive был установлен после Trino, сервис Trino необходимо перезапустить.
- Конфигурация Hive-каталога активна.

Активация каталога Hive

По умолчанию встроенный каталог Hive отключен в сервисе Trino. Для активации каталога используйте ADCM. Для этого включите опцию ***Hive configuration*** сервиса Trino (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Configuration***), а затем перезапустите сервис.

После активации следует проверить доступность встроенного каталога Hive. Для этого подключитесь к Trino Coordinator (используя Trino CLI или SQL-клиент, например, DBeaver) и выполните команду:

```sql
SHOW CATALOGS;
```

В списке появится встроенный каталог Hive:

```
Catalog |
--------+
hive-adh|
system  |
```

> **РЕКОМЕНДАЦИЯ**
>
> Актуальная JDBC-строка для подключения к сервису Trino доступна на странице конфигурации сервиса в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Info***).

Поддерживаемые форматы Hive

Hive-каталог сервиса Trino поддерживает следующие форматы таблиц Hive:

- ORC
- Parquet
- Avro

Также поддерживаются следующие SerDe-форматы:

- TextFile
- CSV с использованием `org.apache.hadoop.hive.serde2.OpenCSVSerde`
- JSON с использованием `org.apache.hive.hcatalog.data.JsonSerDe`
- RCFile с использованием `ColumnarSerDe`
- RCFile с использованием `LazyBinaryColumnarSerDe`
- SequenceFile

Параметры конфигурации Hive-каталога

Hive-каталог сервиса Trino поддерживает множество параметров конфигурации для тонкой настройки взаимодействия с файлами Hive. Актуальный список параметров доступен в документации Trino. Используя ADCM, можно установить новые или переопределить дефолтные параметры. Для этого используйте раздел настроек ***Custom Hive configuration*** в ADCM (***Clusters*** → ***<clusterName>*** → ***Services*** → ***Trino*** → ***Configuration***).

Поддержка SQL

> **ВАЖНО**
>
> Чтобы Trino мог записывать данные в таблицы Hive, при подключении к Trino (например, через Trino CLI или DBeaver) необходимо указывать имя пользователя, соответствующее Unix-пользователю на хосте ADH, у которого имеются достаточные права для записи в HDFS-директории хранилища Hive. В противном случае вы можете наблюдать следующую ошибку:
>
> ```
> Failed to create external path hdfs://adh/apps/hive/warehouse/trino_demo_schema1.db for database
> trino_demo_schema1. This may result in access not being allowed if the StorageBasedAuthorizationProvider is
> enabled: null
> ```

Управление схемой

Схемы Trino используются для группировки нескольких таблиц/представлений и эквивалентны базам данных в Hive. Ниже показаны базовые команды для работы со схемами:

```sql
CREATE SCHEMA "hive-adh"."trino_demo_schema_hive"; ①
USE "hive-adh"."trino_demo_schema_hive"; ②
SHOW SCHEMAS;
DROP "hive-adh"."trino_demo_schema_hive"; ③
```

① Создание новой схемы Trino (базы данных Hive). `hive-adh` — имя встроенного каталога Hive. Обратите внимание на двойные кавычки, используемые для обрамления идентификаторов.

② Выбор схемы для сессии. Все последующие SQL-команды Trino будет выполнять в указанной базе данных Hive в рамках текущей сессии.

③ Удаление схемы, в которой имеются таблицы, запрещено. Чтобы удалить схему, необходимо сначала удалить все вложенные таблицы.

Для выполнения запросов к произвольным таблицам, которые принадлежат разным схемам, можно указать каталог, схему и имя таблицы в SQL-запросе, например:

```sql
SELECT * FROM "hive-adh"."trino_demo_schema_hive"."transactions" txns_hive
JOIN "iceberg-adh"."trino_demo_schema_iceberg"."transactions" txns_icb
ON txns_hive.txn_id = txns_icb.txn_id;
```

> **РЕКОМЕНДАЦИЯ**
>
> Используя Trino UI, можно просматривать и анализировать SQL-запросы, отправленные в Trino.

Примеры DDL

Trino поддерживает большинство DDL-операций Hive для управления сущностями, такими как таблицы, базы данных, процедуры и так далее. Trino также поддерживает набор конфигурационных свойств для таблиц Hive, которые позволяют задавать метаданные для таблиц. Эти свойства можно назначить таблице Hive, используя предложение `WITH`, как показано далее в примерах. Ниже представлены примеры базовых DDL-команд с использованием каталога Hive.

```sql
CREATE TABLE "hive-adh"."trino_demo_schema_hive"."transactions" (id BIGINT, value VARCHAR); ①

CREATE TABLE "hive-adh"."trino_demo_schema_hive"."transactions_with" (id BIGINT, value VARCHAR) ②
WITH (
    format = 'ORC',
    transactional=true
);

CREATE TABLE "hive-adh"."trino_demo_schema_hive"."transactions_pttnd" ( ③
    txn_id BIGINT,
    txn_amount DOUBLE,
    acc_id BIGINT
 ) WITH (
    partitioned_by=ARRAY['acc_id']
);

CREATE TABLE "hive-adh"."trino_demo_schema_hive"."transactions_bckttd" ( ④
    txn_id BIGINT,
    txn_amount DOUBLE,
    acc_id BIGINT
 ) WITH (
    bucketed_by=ARRAY['acc_id'],
    bucket_count=20
);

CREATE TABLE "hive-adh"."trino_demo_schema_hive"."transactions_ext" (
        id BIGINT,
        payload VARCHAR,
        request_time TIMESTAMP
) WITH (
  format = 'TEXTFILE',
  external_location = 'hdfs://adh/apps/hive/ext/test_ext_table' ⑤
);

SET SESSION "hive-adh".non_transactional_optimize_enabled=true; ⑥
ALTER TABLE "hive-adh"."trino_demo_schema_hive"."transactions_pttnd"
EXECUTE optimize (file_size_threshold => '128MB'); ⑦

ANALYZE "hive-adh"."trino_demo_schema_hive"."transactions_pttnd"; ⑧

CALL system.create_empty_partition( ⑨
    schema_name => "hive-adh",
    table_name => "trino_demo_schema_hive",
    partition_columns => ARRAY['acc_id'],
    partition_values => ARRAY['1005']
);
```

① Создание базовой таблицы Hive. Обратите внимание, что Trino использует ANSI SQL, в то время как Hive использует HiveQL, который имеет некоторые отличия, включая разницу в типах данных. Например, при выполнении этого запроса тип данных `VARCHAR` будет приведен к типу данных Hive `STRING`.

② Создание транзакционной таблицы в формате ORC. Для создания транзакционных таблиц через Trino необходимо использовать формат ORC. Если формат таблицы не указан явно, Trino создаст нетранзакционную таблицу, используя формат, указанный в конфигурационном свойстве каталога `hive.storage-format`.

③ Создание партиционированной таблицы Hive. Обратите внимание на синтаксис `ARRAY['col1', 'col2']` для указания партиций.

④ Создание таблицы с определенным количеством бакетов.

⑤ Создание external-таблицы с указанием кастомной директории в HDFS.

⑥ Установка свойства сессии (session property). В данном примере установка `non_transactional_optimize_enabled=true` необходима для выполнения команды `OPTIMIZE`.

⑦ Запуск команды `OPTIMIZE`, которая переписывает содержимое таблицы, объединяя множество мелких файлов в файлы большего размера.

⑧ Сбор статистики таблицы. Trino не поддерживает сбор статистики для транзакционных таблиц Hive.

⑨ Вызов системной процедуры для создания пустой партиции. Эта команда создает пустую директорию в хранилище HDFS, например, */apps/hive/warehouse/trino_demo_schema.db/transactions_pttnd/acc=1005*. Больше подробной информации о процедурах, поддерживаемых каталогом Hive, можно найти в документации Trino.

Примеры DML

Trino поддерживает большинство DML-операций Hive, например `INSERT`, `UPDATE`, `DELETE`, `MERGE` и так далее. Фактическое поведение этих команд зависит от формата таблиц и используемого хранилища метаданных (Metastore). Ниже приведены некоторые ограничения и особенности выполнения CRUD-операций над таблицами Hive с помощью Trino:

- Операции `UPDATE` поддерживаются только для транзакционных таблиц. Необходимо, чтобы целевая таблица была транзакционной, ACID и имела формат ORC.
- Команда `DELETE`, применяемая к нетранзакционным таблицам, поддерживается только в случае, если таблица разбита на партиции, а фильтр предложения `WHERE` строго соответствует партиции. Для транзакционных таблиц Hive в формате ORC поддерживается построчное удаление, а предложение `WHERE` может быть произвольным.
- Операция `MERGE` поддерживается только для ACID-таблиц.
- Создание ACID-таблиц с помощью Hive Streaming Ingest не поддерживается.
- Сбор статистики с помощью `ANALYZE` не поддерживается для транзакционных таблиц Hive.
- Операция `TRUNCATE` не поддерживается каталогом Hive.

Больше подробной информации о лучших практиках и ограничениях работы с каталогом Hive доступно в документации Trino. Ниже представлены примеры DML-операций с использованием Trino-каталога Hive.

```sql
INSERT INTO "hive-adh"."trino_demo_schema_hive"."transactions_pttnd"
    VALUES (1, 15, 1001),
    (2, 100, 1002),
    (3, 75, 1001);

SELECT * FROM "hive-adh"."trino_demo_schema_hive"."transactions_pttnd"; ①

SELECT * FROM "hive-adh"."trino_demo_schema_hive"."transactions_pttnd$properties"; ②

SELECT * FROM "hive-adh"."trino_demo_schema_hive"."transactions_pttnd$partitions"; ③

SELECT *, "$path", "$file_size", "$partition" FROM "hive-adh"."trino_demo_schema_hive"."transactions_pttnd"; ④

SELECT *, "$path", "$file_size" FROM "hive-adh"."trino_demo_schema_hive"."transactions_pttnd"
WHERE "$partition" = 'acc_id=1001'; ⑤

SELECT * FROM "hive-adh"."trino_demo_schema_hive"."transactions" txns_hv ⑥
JOIN "iceberg-adh"."trino_demo_schema_iceberg"."transactions" txns_icb
ON txns_hv.txn_id = txns_icb.txn_id; ⑥

DELETE FROM "hive-adh"."trino_demo_schema_hive"."transactions_pttnd" ⑦
WHERE txn_id = 2;
```

① ANSI `SELECT` для выборки из базы данных Hive.

② Получение метаданных из скрытой таблицы `$properties`.

③ Получение списка всех доступных партиций в партиционированной таблице.

④ Получение метаданных из таблицы метаданных для столбцов.

⑤ Получение информации о файлах таблицы и размере файлов для всех записей, принадлежащих определенной партиции (`acc_id=1001`).

⑥ Получение данных из двух каталогов Trino: Hive-каталога `hive-adh` и Iceberg-каталога `iceberg-adh`. Получив результаты выборки из каждого каталога, Trino применяет операцию `JOIN` и возвращает клиенту конечный набор результатов. Чтобы данный запрос работал корректно, оба каталога должны быть включены в сервисе Trino.

⑦ Удаление строки. Целевая таблица должна быть транзакционной, иначе Trino выдаст ошибку.

Больше информации о DML-командах, поддерживаемых Trino коннектором для Hive, можно найти в документации Trino.

Оптимизация производительности

Trino по умолчанию использует некоторые методы оптимизаций для ускорения работы с таблицами Hive. К ним относятся:

- Сбор статистики таблицы. Коннектор Hive использует статистику таблиц (для транзакционных таблиц) для повышения производительности обработки запросов.
- Динамическая фильтрация. Обеспечивает прирост скорости выполнения запросов с выборочными операциями `JOIN`. Позволяет избежать чтения "ненужных" данных, которые отфильтровываются условием `JOIN`.
- Кеширование файловой системы. Кеширует фрагменты данных, которые часто считываются из HDFS и к которым обращаются Trino Workers при выполнении схожих запросов.
- Перенаправление таблиц. Метод, позволяющий перенаправлять CRUD-операции в другой каталог.

Trino ADB connector

Константин Алпашкин

Назад к содержанию

Arenadata DB (ADB) — это распределенная аналитическая СУБД, построенная на базе MPP-системы с открытым исходным кодом Greenplum.

ADH-сервис Trino предоставляет движок для выполнения запросов SQL, используемый для параллельной обработки данных, распределенных по нескольким хранилищам, таким как объектные хранилища, базы данных и файловые системы. Сервис Trino поставляется с коннектором, который позволяет выполнять запросы в ADB-кластере.

В следующих статьях содержится подробное описание работы с Trino-коннектором для ADB:

- Обзор
- Требования
- Пример использования
- Настройки

Обзор Trino ADB connector

Константин Алпашкин

Содержание

Назад к содержанию

Режимы работы коннектора
  Использование режима параллельного курсора

Для подключения к кластеру ADB через Trino-коннектор необходимо создать новый каталог Trino. В конфигурации каталога должен использоваться тип коннектора `adb`, а также должны быть указаны следующие параметры (минимальный набор):

```
connector.name=adb ①
connection-url=jdbc:postgresql://<adb_master>:5432/database ②
connection-user=<adb_username> ③
connection-password=<password>
```

① Установка коннектора для работы с ADB.

② JDBC-строка для подключения к мастер-хосту кластера ADB. Коннектор использует JDBC-драйвер PostgreSQL и поддерживает все его свойства конфигурации, а также наследует ограничения. Больше информации доступно на странице PostgreSQL connector.

③ Пользователь ADB, используемый коннектором для подключения к ADB-кластеру. Пользователь должен быть заранее создан на стороне ADB и должен иметь достаточные права для выполнения запросов.

Подробная информация о настройке Trino-каталога для ADB доступна в разделе Параметры каталога Trino, а также в примере.

Режимы работы коннектора

Trino-коннектор может взаимодействовать с кластером ADB в двух режимах:

- **Операции чтения/записи выполняются с помощью gpfdist**.

  `gpfdist` — это утилита для распределенных операций чтения и записи данных из файлов, которая позволяет параллельно загружать и выгружать данные из внешних таблиц ADB. Под капотом коннектор Trino разворачивает серверы `gpfdist` на хостах ADH, а сегменты ADB используют `gpfdist`-клиенты для параллельного обмена данными с коннектором.

- **Запись данных в ADB через gpfdist, чтение с помощью gp_parallel_retrieve_cursor**.

  Модуль `gp_parallel_retrieve_cursor` — это улучшенная реализация курсора (cursor), которая позволяет параллельно получать результаты запросов непосредственно из сегментов ADB. Данный тип курсора обеспечивает более быстрое параллельное чтение, добавляет дополнительную гибкость за счет чтения данных в виде фрагментов и в целом более эффективен при обработке больших наборов результатов.

Использование режима параллельного курсора

Чтобы активировать режим чтения данных из ADB с помощью `gp_parallel_retrieve_cursor`, выполните следующие шаги:

1. В ADB-кластере создайте расширение `gp_parallel_retrieve_cursor`:

   ```
   CREATE EXTENSION gp_parallel_retrieve_cursor;
   ```

2. При создании нового Trino-каталога для ADB установите свойство `adb.connector.gp-parallel-cursor.enabled=true`.

Требования к Trino ADB connector

Константин Алпашкин

Содержание

Назад к содержанию

Требования к ADB

Требования к ADH

Общие требования

В этой статье описаны требования к кластерам ADB и ADH, необходимые для взаимодействия Trino-коннектора с кластером ADB. Эти требования обязательны для корректного запуска примера использования коннектора.

Требования к ADB

- Кластер ADB развернут согласно руководству Online-установка.
- Кластер ADB использует порт по умолчанию (`5432`) для входящих JDBC-подключений. SSL выключен.
- Пользователь ADB, который используется Trino-коннектором для доступа к ADB-кластеру, имеет достаточные права для чтения/записи данных. Более подробно о предоставлении прав пользователю ADB описано в примере.

Требования к ADH

- Кластер ADH развернут согласно руководству Online-установка. Минимальная версия ADH — 4.0.0.
- В кластере ADH установлен сервис Trino.

Общие требования

- Хосты ADB и ADH доступны друг другу по сети. Порты, используемые компонентами ADB/ADH, не заняты.
- Наличие JDBC-клиента для подключения к сервису Trino (например, DBeaver).

Пример использования Trino ADB connector

Константин Алпашкин

Содержание

Назад к содержанию

Подготовка кластера ADB

Выполнение сценария

В данной статье описан пошаговый сценарий, в котором показаны основные операции с ADB-кластером, выполняемые с использованием коннектора Trino.

Подготовка кластера ADB

Для корректного выполнения сценария убедитесь, что кластер ADB настроен в соответствии со следующими требованиями:

- В кластере ADB создан пользователь `adb_trino_demo_user` с правами `CREATEEXTTABLE` и паролем `123`.

  Чтобы добавить пользователя, на мастер-хосте ADB запустите `psql`:

  ```
  $ sudo su - gpadmin
  $ psql adb
  ```

  В терминале `psql` выполните следующий запрос:

  ```
  CREATE ROLE adb_trino_demo_user
  WITH  CREATEEXTTABLE(protocol='gpfdist',type='readable')
        CREATEEXTTABLE(protocol='gpfdist',type='writable')
        LOGIN
        PASSWORD '123';
  ```

- В кластере ADB существует база данных `adb_trino_demo_db`. Чтобы создать БД, выполните запрос:

  ```
  CREATE DATABASE adb_trino_demo_db;
  ```

- Пользователь `adb_trino_demo_user` имеет доступ к базе данных `adb_trino_demo_db`. Для этого в файл *pg_hba.conf* необходимо добавить следующую запись:

  ```
  host    all  adb_trino_demo_user       0.0.0.0/0     md5
  ```

  Обратите внимание, что данная конфигурация предназначена для использования исключительно в тестовых целях. Вместо `0.0.0.0/0` можно указать IP-адреса (включая подсеть) хостов ADH-кластера, на которых установлены компоненты сервиса Trino.

> **РЕКОМЕНДАЦИЯ**
>
> Файл *pg_hba.conf* можно редактировать в ADCM. Для этого используйте опцию ***Custom pg_hba section*** на вкладке ***Configuration*** сервиса ADB. Чтобы изменения вступили в силу, нажмите ***Save*** и выполните сервисное действие ***Reconfigure & Restart***.

- В базе данных `adb_trino_demo_db` создана тестовая таблица `author` (в схеме `public`). Для создания и наполнения тестовой таблицы используйте следующие запросы в `psql`.

  Подключитесь к тестовой базе данных:

  ```
  \c adb_trino_demo_db
  ```

  Затем выполните запросы:

  ```
  CREATE TABLE public.author (id INT NOT NULL, name TEXT NOT NULL)
  WITH (appendoptimized=true)
  DISTRIBUTED BY(id);
  ```

  ```
  INSERT INTO public.author(id, name) VALUES
  (1,'Virginia Woolf'),
  (2,'J.R.R. Tolkien'),
  (3,'Harper Lee'),
  (4,'J.D. Salinger'),
  (5,'George Orwell'),
  (6,'John Steinbeck'),
  (7,'Margaret Mitchell'),
  (8,'Alan Moore'),
  (9,'Jack Kerouac'),
  (10,'Ernest Hemingway');
  ```

- ADB-пользователь `adb_trino_demo_user` обладает достаточными правами для выполнения операций с таблицей `author`. Для предоставления прав на чтение/запись выполните запрос:

  ```
  GRANT SELECT, INSERT ON public.author TO adb_trino_demo_user;
  ```

Выполнение сценария

Перед запуском примеров убедитесь, что все требования выполнены.

1. Создайте новый каталог Trino для работы с ADB. Для этого перейдите в настройки сервиса Trino в ADCM, активируйте ***Custom catalogs*** и укажите следующие параметра каталога:

catalog_name	key	value
adbcatalog	connector.name	adb
	connection-url	*jdbc:postgresql://10.92.40.79:5432/adb_trino_demo_db* Где: ■ `10.92.40.79` — пример адреса мастер-хоста ADB. Используйте IP вашего хоста. ■ `adb_trino_demo_db` — тестовая база данных ADB. БД должна быть создана заранее, пример создания описан в требованиях.
	connection-user	adb_trino_demo_user
	connection-password	123

2. Сохраните конфигурацию и перезапустите сервис Trino.

> **РЕКОМЕНДАЦИЯ**
>
> Используя динамическое управление каталогами, можно создавать/удалять каталоги с помощью команд `CREATE` / `DELETE` без необходимости перезапускать сервис Trino.

3. Подключитесь к Trino, используя JDBC-клиент, например DBeaver (или клиент Trino CLI). При подключении укажите JDBC URL сервиса Trino, например:

   ```
   jdbc:trino://ka-adh-2.ru-central1.internal:18188
   ```

   Актуальная строка подключения доступна в ADCM (***Clusters*** → ***<ADH_cluster>*** → ***Services*** → ***Trino*** → ***Info***).

4. Подключившись к сервису Trino, выведите список каталогов Trino с помощью команды:

   ```
   SHOW CATALOGS;
   ```

   Убедитесь, что тестовый каталог ADB доступен:

   ```
   Catalog   |
   ----------+
   adbcatalog|
   system    |
   ```

> **РЕКОМЕНДАЦИЯ**
>
> Для анализа запросов, которые Trino отправляет в ADB, используйте Trino UI.

5. Выберите тестовый каталог и схему для дальнейших запросов:

   ```
   USE adbcatalog.public;
   ```

6. Запросите данные из тестовой таблицы ADB. Шаги по созданию тестовой таблицы описаны в требованиях.

   ```
   SELECT * FROM author;
   ```

   Вывод:

   ```
   id|name             |
   --+-----------------+
    2|J.R.R. Tolkien   |
    1|Virginia Woolf   |
    6|John Steinbeck   |
    9|Jack Kerouac     |
    5|George Orwell    |
    3|Harper Lee       |
    7|Margaret Mitchell|
    8|Alan Moore       |
    4|J.D. Salinger    |
   10|Ernest Hemingway |
   ```

7. Запишите данные в тестовую таблицу:

   ```
   INSERT INTO author (id, name)
   VALUES (11, 'L.N. Tolstoy');
   ```

8. Проверьте результаты записи на стороне ADB.

   Для этого запустите `psql` на мастер-хосте ADB:

   ```
   $ sudo su - gpadmin
   $ psql adb_trino_demo_db
   ```

   В терминале `psql` выполните запрос:

   ```
   SELECT * FROM public.author
   WHERE name LIKE '%Tolstoy%';
   ```

   Вывод содержит запись, записанную в ADB через Trino-коннектор:

   ```
    id |     name
   ----+--------------
    11 | L.N. Tolstoy
   (1 row)
   ```

9. Создайте новую таблицу в ADB с помощью коннектора Trino. Для этого выполните следующий запрос в DBeaver:

   ```
   CREATE TABLE public.transactions (
     txn_id integer,
     acc_id varchar,
     txn_value double,
     txn_date date
   )
   WITH (
     distributed_by = ARRAY['acc_id', 'txn_date'], ①
     appendoptimized = true ②
   );
   ```

   ① Хеш-распределение данных по нескольким столбцам.

   ② Установка типа таблицы append-optimized.

   Подробная информация о свойствах ADB-таблиц, которые можно установить с помощью Trino-коннектора, доступна в разделе Свойства таблиц ADB.

   ```
   INSERT INTO public.transactions
   (txn_id, acc_id, txn_value, txn_date)
   VALUES
       (1, '1001', cast(20.00 as double), cast('2025-01-02' AS date)),
       (2, '1002', cast(110.50 as double), cast('2025-01-01' AS date)),
       (3, '1003', cast(23 as DOUBLE), cast('2025-01-01' AS date));
   ```

10. Проверьте создание таблицы на мастер-хосте ADB с помощью `psql`:

    ```
    \dt;
    ```

    Новая таблица доступна в списке:

    ```
                             List of relations
     Schema |     Name     | Type  |        Owner        |   Storage
    --------+--------------+-------+---------------------+-------------
     public | author       | table | gpadmin             | append only
     public | transactions | table | adb_trino_demo_user | append only
    (2 rows)
    ```

11. Для удаления таблицы используйте команду:

    ```
    DROP TABLE IF EXISTS public.transactions;
    ```

> **ПРИМЕЧАНИЕ**
>
> Коннектор Trino может удалять таблицы, только если в настройках ADB-каталога указан пользователь ADB, являющийся владельцем таблицы (table owner).

Настройки Trino ADB connector

Константин Алпашкин

Содержание

Назад к содержанию

- Параметры каталога Trino
- Свойства таблиц ADB
- Сопоставление типов данных
  - Параметры сопоставления типов данных
  - Особенности работы с DECIMAL
  - Особенности работы с ARRAY

В данной статье описаны параметры конфигурации коннектора Trino для ADB.

Параметры каталога Trino

Следующие конфигурационные параметры могут использоваться при создании нового каталога Trino для работы с кластером ADB. Данные параметры можно указать в ADCM или с помощью предложения `WITH`. Например:

```
CREATE CATALOG test_adb_catalog USING adb
WITH (
  "connection-url" = 'jdbc:postgresql://<adb_master>:5432/trino',
  ...
  "configuration_param1" = "value"
  "configuration_param2" = "value"
);
```

Параметр	Описание	Значение по умолчанию
adb.fetch-size	Максимальное количество строк для выборки. Если значение не указано, оно вычисляется по формуле: $\max\left(\frac{100000}{\text{column\_count}}, 1000\right)$	—
adb.array-mapping	Способ преобразования данных типа массив. Возможные значения: `DISABLED`, `AS_ARRAY`	DISABLED
adb.enable-string-pushdown-with-collate	Включает оптимизацию pushdown для строковых типов данных с учетом collate	false
adb.include-system-tables	Указывает, следует ли включать системные таблицы	false
adb.max-scan-parallelism	Максимальный уровень параллелизма для сканирования таблиц	1
adb.connector.write-buffer-size	Максимальный объем памяти, выделяемый на один приемник (sink) для запросов на запись	64MB
adb.connector.read-buffer-size	Максимальный объем памяти, выделяемый на объект курсора для запросов на чтение	64MB
adb.connector.gpfdist.retry-timeout	Значение ADB-свойства `gpfdist_retry_timeout`. Если не указано, используется значение ADB по умолчанию	—
adb.connector.gp-parallel-cursor.enabled	Активирует режим gp-parallel-cursor для чтения данных из ADB	false

Свойства таблиц ADB

Следующие свойства таблиц ADB могут быть установлены с помощью предложения `WITH` при выполнении команд `CREATE TABLE` / `CREATE TABLE AS`. Данные свойства по умолчанию имеют значение `NULL`. Если свойство явно не указано при создании таблицы с помощью коннектора, будут использоваться дефолтные значения кластера ADB.

Свойство таблицы	Описание
distributed	Указывает способ распределения данных таблицы, который не зависит от конкретных столбцов. Допустимые значения: `randomly`, `replicated`. Для хеш-распределения используйте параметр `distributed_by`
distributed_by	Указывает хеш-распределение данных таблицы. Значение должно представлять собой массив имен столбцов, например `ARRAY['col1', 'col2']`
appendoptimized	Если установлено значение `true`, создается таблица типа append-optimized. При значении `false` создается таблица типа heap
blocksize	Устанавливает размер блока в байтах для таблиц типа append-optimized
orientation	Устанавливает тип ориентации данных в таблицах типа append-optimized. Допустимые значения: `column`, `row`
checksum	Установка `true` включает проверку контрольных сумм CRC для таблиц типа append-optimized
compresstype	Указывает алгоритм сжатия данных. Допустимые значения: `zlib`, `zstd`, `rle_type`, `none`. Фактическая поддержка конкретного алгоритма зависит от версии ADB
compresslevel	Устанавливает уровень сжатия. Допустимые значения зависят от алгоритма сжатия: ▪ zlib: `1` — `9`; ▪ zstd: `1` — `19`; ▪ rle_type: `1` — `4`.
fillfactor	Устанавливает коэффициент заполнения для heap-таблиц. Допустимые значения: от `10` до `100`

Ниже приведен пример установки параметров ADB-таблицы с помощью коннектора Trino.

```
CREATE TABLE nations (
  n_nationkey integer,
  n_name char(25),
  n_regionkey integer,
  n_comment varchar(152)
)
WITH (
  distributed_by = ARRAY['nationkey', 'regionkey'], ①
  appendoptimized = true, ②
  orientation = 'column', ③
  compresstype = 'zstd' ④
)
```

① Хеш-распределение по нескольким столбцам.

② Установка типа таблицы append-optimized.

③ Использование колоночной ориентации данных.

④ Установка уровня сжатия с использованием алгоритма ZSTD.

Сопоставление типов данных

При обмене данными между ADB и коннектором Trino типы данных преобразуются в соответствии со следующими правилами.

Преобразование типов ADB в Trino

Тип данных ADB	Тип данных Trino
BIT	BOOLEAN
BOOLEAN	BOOLEAN
SMALLINT	SMALLINT
INTEGER	INTEGER
BIGINT	BIGINT
REAL	REAL
DOUBLE	DOUBLE
NUMERIC(p, s)	DECIMAL(p, s) Тип DECIMAL(p, s) является алиасом NUMERIC(p, s)
CHAR(n)	CHAR(n)
VARCHAR(n)	VARCHAR(n)
ENUM	VARCHAR
BYTEA	VARBINARY
DATE	DATE
TIME(n)	TIME(n)
TIMESTAMP(n)	TIMESTAMP(n)
TIMESTAMPTZ(n)	TIMESTAMP(n) WITH TIME ZONE
MONEY	VARCHAR
UUID	UUID
JSON	JSON
JSONB	JSON
HSTORE	MAP(VARCHAR, VARCHAR)
ARRAY	DISABLED, ARRAY Больше информации доступно в разделе Особенности работы с ARRAY

Преобразование типов Trino в ADB

Тип данных ADB	Тип данных Trino
BOOLEAN	BOOLEAN
SMALLINT	SMALLINT
TINYINT	SMALLINT
INTEGER	INTEGER
BIGINT	BIGINT
REAL	REAL
DOUBLE	DOUBLE
DECIMAL(p, s)	NUMERIC(p, s) Тип DECIMAL(p, s) является алиасом NUMERIC(p, s)
CHAR(n)	CHAR(n)
VARCHAR(n)	VARCHAR(n)
VARBINARY	BYTEA
DATE	DATE
TIME(n)	TIME(n)
TIMESTAMP(n)	TIMESTAMP(n)
TIMESTAMP(n) WITH TIME ZONE	TIMESTAMPTZ(n)
UUID	UUID
JSON	JSONB
ARRAY	ARRAY Больше информации доступно в разделе Особенности работы с ARRAY

Параметры сопоставления типов данных

Следующие конфигурационные параметры позволяют изменить логику приведения типов данных.

Параметр	Описание	Значение по умолчанию
unsupported-type-handling	Определяет поведение при обработке столбцов с неподдерживаемым типом данных. Допустимые значения: ▪ `IGNORE` — данные столбца игнорируются. ▪ `CONVERT_TO_VARCHAR` — значение столбца преобразуется в тип `VARCHAR` неограниченной длины. Соответствующий параметр сессии — `unsupported_type_handling`	IGNORE
jdbc-types-mapped-to-varchar	Список типов данных, которые необходимо принудительно привести к типу `VARCHAR` неограниченной длины (даже если указанный тип поддерживается коннектором). Например, `jdbc-types-mapped-to-varchar=VARBINARY,UUID,`	—

Особенности работы с DECIMAL

По умолчанию коннектор Trino выдает ошибку при обработке столбцов типа `DECIMAL` без явного указания параметров точности (precision) и масштаба (scale). Если конфигурационный параметр `decimal-mapping` (или параметр сессии `decimal_mapping`) имеет значение `allow_overflow`, то столбцы типа `DECIMAL` без явного указания точности/масштаба будут преобразованы в тип данных коннектора `DECIMAL(38,0)`. Используя конфигурационный параметр `decimal-default-scale` (или параметр сессии `decimal_default_scale`), можно указать собственное значение масштаба. Значение точности всегда равно `38`.

Если значение `DECIMAL` не помещается в отведенные разряды точности/масштаба без округления, коннектор выдает ошибку. Это поведение можно изменить, включив автоматическое округление с помощью параметра коннектора `decimal-rounding-mode` (или параметра сессии `decimal_rounding_mode`). Поддерживаются следующие значения округления: `UNNECESSARY` (по умолчанию), `UP`, `DOWN`, `CEILING`, `FLOOR`, `HALF_UP`, `HALF_DOWN` и `HALF_EVEN`. Эти значения соответствуют правилам округления, принятым в Java 17.

Особенности работы с ARRAY

Тип данных `ARRAY` в ADB не поддерживает массивы фиксированной размерности, в то время как коннектор Trino для ADB поддерживает только одномерные массивы. С помощью параметра `adb.array-mapping` (параметр сессии `array_mapping`) можно указать, как коннектор должен обрабатывать типы данных `ARRAY`. Допустимые значения параметра:

- `DISABLED` (по умолчанию) — коннектор возвращает ошибку при обработке столбцов типа `ARRAY`.
- `AS_ARRAY` — коннектор обрабатывает массивы ADB как тип данных `ARRAY`. Это значение подходит для массивов ADB, которые фактически являются одномерными. Для многомерного массива будет сформировано исключение.

Обзор Trino OpenAPI connector

Константин Алпашкин

Содержание

Назад к содержанию

- Анализ схемы OpenAPI
- Расширения OpenAPI
- Сообщения об ошибках
- Дополнительные возможности и ограничения
  - Ограничения

Trino OpenAPI-коннектор позволяет взаимодействовать с REST API, отвечающим стандартам OpenAPI. OpenAPI — это стандарт описания функционала REST API, с помощью которого указываются доступные эндпойнты, поддерживаемые HTTP-методы, типы ответов, параметры запросов и так далее. Коннектор анализирует OpenAPI-схему (JSON/YAML) и представляет REST-эндпойнты сервера в виде отдельных таблиц. Затем коннектор преобразует запросы SQL в HTTP-запросы, обрабатывает ответы и возвращает результаты пользователю Trino.

ПРИМЕЧАНИЕ

Настоятельно рекомендуется использовать коннектор только для чтения. Операции вставки и обновления реализованы в экспериментальном режиме.

Trino OpenAPI-коннектор поставляется с ADH и готов к использованию "из коробки". Чтобы приступить к работе с коннектором, смотрите статью Пример использования OpenAPI connector. Подробности настройки коннектора описаны в статье Настройка OpenAPI connector.

Анализ схемы OpenAPI

Анализируя схему OpenAPI, коннектор создает SQL-сущности (таблицы, столбцы, запросы) по следующим правилам:

- REST-эндпойнты представляются в виде отдельных таблиц. Например, эндпойнт *[BaseURL]/transactions* представлен для Trino-клиента в виде таблицы `transactions`. Регистр `fooBar` преобразуется в `foo_bar`, а специальные символы, такие как `/`, преобразуются в `_`.
- Операции SQL преобразуются в HTTP-запросы следующим образом:
  - `SELECT` ⇒ `GET` или `POST` (если `GET` недоступен);
  - `INSERT` ⇒ `POST` / `PUT`;
  - `UPDATE` ⇒ `PATCH` / `POST`;
  - `DELETE` ⇒ `DELETE`.
- Параметры запросов преобразуются в столбцы, включая параметры в пути (path), query-параметры и заголовки.
- Поля в ответах `HTTP OK (200)` преобразуются в столбцы таблиц.
- Сгенерированные столбцы уникальны:
  - Если ответ сервера содержит поля с одинаковыми именами, но разными типами данных, такому полю добавляется постфикс `_2`, `_3` и так далее.
  - Если в запросе к серверу присутствует поле, у которого есть такое же наименование в ответе, к имени поля из тела запроса добавляется постфикс `_req`.

Расширения OpenAPI

OpenAPI поддерживает пользовательские расширения, которые указываются в схеме с помощью блоков с префиксом `x-`. Расширения OpenAPI могут использоваться для тонкой настройки или для реализации дополнительных возможностей коннектора (например, x-unwrap, x-pagination).

РЕКОМЕНДАЦИЯ

Если схема OpenAPI, возвращаемая REST-сервером, не может быть изменена для включения расширений `x-`, можно сохранить файл схемы локально и изменить его при необходимости.

Сообщения об ошибках

Когда коннектор обращается к API и получает ошибку, клиенту Trino возвращается ошибка SQL. Обычно такие ошибки содержат только код HTTP-ответа, а сообщение ошибки не отображается (сообщение может быть нечитаемым или содержать конфиденциальные данные).

Для отображения информативных сообщений об ошибках, используйте свойство `errorPath` в блоке `x-trino` OpenAPI-схемы. Например:

```
paths:
  /records:
    get:
      responses:
        # ...
      x-trino:
       errorPath: "$response.body#/message"
```

Дополнительные возможности и ограничения

Реализация коннектора, поставляемая с ADH, включает некоторые улучшения и дополнительные возможности, а именно:

- Распаковка массивов. При анализе HTTP-ответа коннектор может автоматически "распаковать" содержимое полей с массивами.
- Пагинация (pagination) для сдвига `LIMIT`. Позволяет получать от REST API разбитые на страницы (page) ответы.

Ограничения

Коннектор имеет следующие ограничения:

- Ограниченная поддержка операций `INSERT` / `DELETE` / `UPDATE`. Коннектор рекомендуется использовать только для операций чтения.
- Ограниченная поддержка pushdown-предикатов. Pushdown-предикаты корректно работают только с API, которые поддерживают фильтрацию параметров запроса и полностью описаны в OpenAPI-схеме. Могут наблюдаться проблемы с производительностью на больших объемах данных, поскольку из-за отсутствия pushdown-предикатов коннектор сначала загружает весь набор данных в память и только затем выполняет фильтрацию.

Пример использования OpenAPI connector

Константин Алпашкин

Содержание

Назад к содержанию

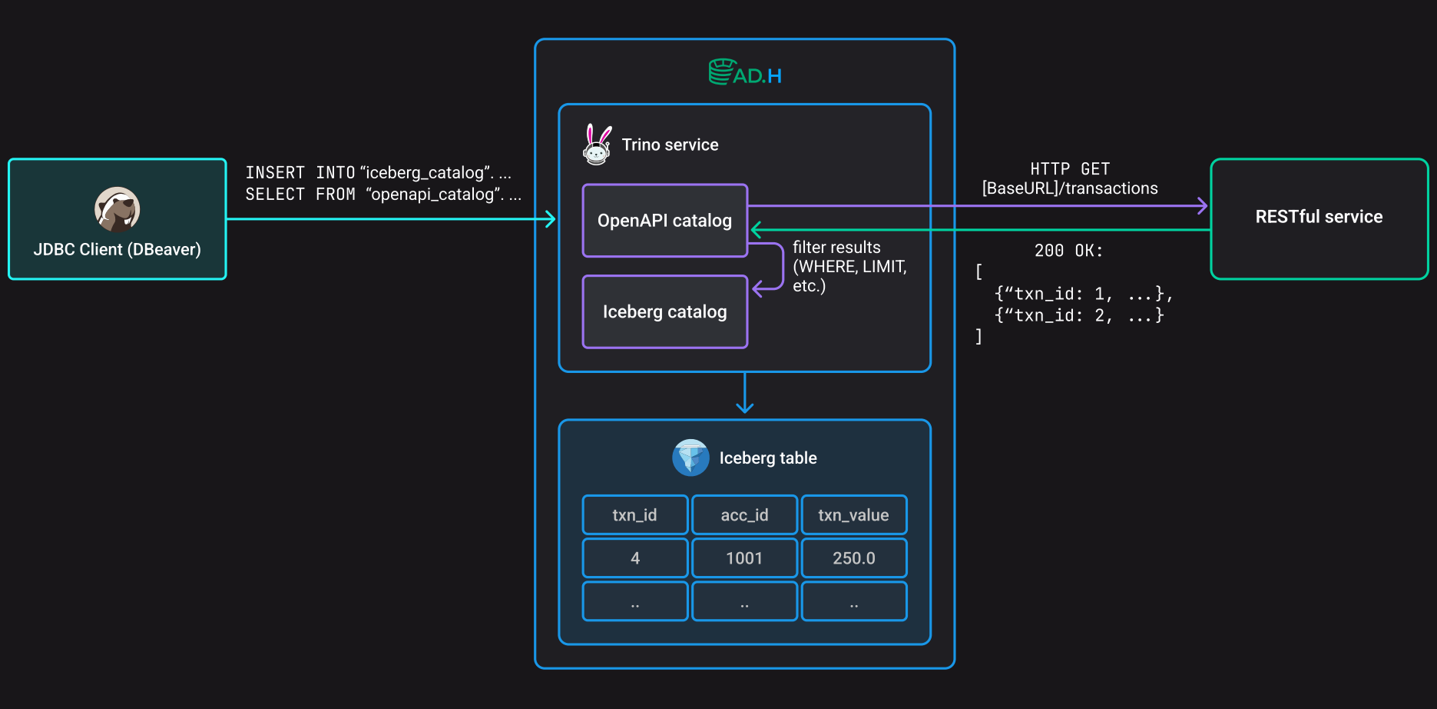
Требования

Подготовка тестового OpenAPI-сервера

- Запуск тестового сервера

Выполнение сценария

В данной статье описан пошаговый сценарий, демонстрирующий использование Trino OpenAPI-коннектора. Алгоритм сценария представлен на следующей схеме.



Пример использования коннектора Trino OpenAPI

Требования

Для выполнения сценария необходимо следующее:

- Кластер ADH версии 4.1.0 или выше установлен и запущен. Кластер содержит следующие сервисы:
  - ADPG
  - Core configuration
  - HDFS
  - Hive
  - Trino
  - YARN
  - ZooKeeper
- Тестовый REST API с OpenAPI-схемой. В этом примере используется приложение Python, эмулирующее RESTful-сервер с банковскими транзакциями. Пример кода и инструкции по запуску приложения приведены в разделе Подготовка тестового OpenAPI-сервера.
- Созданы два каталога Trino:
  - `openapi-catalog`. Каталог для чтения данных из тестового сервера.
  - `iceberg-adh`. Каталог для записи отфильтрованных данных в таблицу Iceberg.

  Этапы создания каталогов описаны в сценарии.
- JDBC-клиент (например, DBeaver) для подключения к сервису Trino.

Подготовка тестового OpenAPI-сервера

В данном примере приложение Python на базе фреймворка FastAPI эмулирует REST-сервер, который возвращает банковские транзакции по HTTP. Тестовый сервер предоставляет следующие REST-эндпойнты:

GET [BaseURL]/transactions

Возвращает список всех доступных транзакций, например:

```
    [
        {
            "txn_id": 1,
            "acc_id": 1002,
            "txn_value": 184.0,
            "txn_type": "withdrawal"
        },
        {
            "txn_id": 2,
            "acc_id": 1000,
            "txn_value": 282.95,
            "txn_type": "spb"
        },
        {
            "txn_id": 3,
            "acc_id": 1001,
            "txn_value": 100.00,
            "txn_type": "deposit"
        }
    ]
```

GET [BaseURL]/transactions_batch

Возвращает пакет транзакций фиксированного размера:

```
{
    "transactions_count": 2,
    "items":
    [
        {
            "txn_id": 1,
            "acc_id": 1002,
            "txn_value": 184.0,
            "txn_type": "withdrawal"
        },
        {
            "txn_id": 2,
            "acc_id": 1000,
            "txn_value": 282.95,
            "txn_type": "spb"
        }
    ]
}
```

GET [BaseURL]/openapi.yaml

Возвращает OpenAPI-схему, описывающую возможности API сервера. Фреймворк FastAPI позволяет автоматически генерировать схему, а пользовательские поля типа `x-…` вставляются в схему программно. Например:

```
openapi: 3.1.0
info:
  title: Transaction API
  description: Transaction API with x-unwrap demo
  version: 1.0.0
paths:
  /transactions:
    get:
      summary: Get Transactions
      operationId: get_transactions_transactions_get
      responses:
        '200':
          description: Successful Response
          content:
            application/json:
              schema:
                items:
                  $ref: '#/components/schemas/Transaction'
                type: array
                title: Response Get Transactions Transactions Get
  /transactions_batch:
    get:
      summary: Get Transactions Batch
      operationId: get_transactions_batch_transactions_batch_get
      responses:
        '200':
          description: Successful Response
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/TransactionBatch'
      x-unwrap:
        resultParam: $response.body#/items
        includeRoot: 'false'
# ...
```

Запуск тестового сервера

Чтобы запустить тестовый сервер, выполните следующие действия:

> **РЕКОМЕНДАЦИЯ**
>
> Используйте интерпретатор Python, поставляемый с ADH (*/opt/python3.10/*).

1. Создайте и активируйте новую виртуальную среду (virtual environment).

```
$ mkdir demo_mock_server
$ cd demo_mock_server
$ /opt/python3.10/bin/python3 -m venv venv
$ source venv/bin/activate
```

2. Создайте файл *server.py* и вставьте в него приведенный ниже код:

server.py

```
from fastapi import FastAPI
from fastapi.openapi.utils import get_openapi
from fastapi.responses import Response
from pydantic import BaseModel
from typing import List
import random
import uvicorn
import yaml


class Transaction(BaseModel):
    txn_id: int
    acc_id: int
    txn_value: float
    txn_type: str


class TransactionBatch(BaseModel):
    total_count: int
    items: List[Transaction]


app = FastAPI(title="Transaction API", version="1.0.0")


def generate_transactions(n: int = 10) -> List[Transaction]:
    acc_ids = [1000, 1001, 1002]
    txn_types = ["deposit", "withdrawal", "spb"]
    transactions = []
    for txn_id in range(1, n + 1):
        transactions.append(
            Transaction(
                txn_id=txn_id,
                acc_id=random.choice(acc_ids),
                txn_value=round(random.uniform(0, 500), 2),
                txn_type=random.choice(txn_types),
            )
        )
    return transactions


transactions = generate_transactions(10)


@app.get("/transactions", response_model=List[Transaction])
def get_transactions():
    return transactions


# solely for x-unwrap demonstration
@app.get("/transactions_batch", response_model=TransactionBatch)
def get_transactions_batch():
    return {
        "total_count": len(transactions) / 2,
        "items": transactions[:5],
    }


def custom_openapi():
    if app.openapi_schema:
        return app.openapi_schema

    openapi_schema = get_openapi(
        title=app.title,
        version=app.version,
        description="Transaction API with x-unwrap demo",
        routes=app.routes,
    )

    # Add x-unwrap for /transactions_batch
    openapi_schema["paths"]["/transactions_batch"]["get"]["x-unwrap"] = {
        "resultParam": "$response.body#/items",
        "includeRoot": "false"
    }

    app.openapi_schema = openapi_schema
    return app.openapi_schema


app.openapi = custom_openapi


@app.get("/openapi.yaml", include_in_schema=False)
def get_openapi_yaml():
    openapi_schema = custom_openapi()
    yaml_schema = yaml.dump(openapi_schema, allow_unicode=True,
                            sort_keys=False)
    return Response(content=yaml_schema, media_type="application/yaml")


if __name__ == "__main__":
    uvicorn.run(app, host="0.0.0.0", port=8003)
```

3. Установите зависимости, используя *requirements.txt*:

```
$ pip install -r requirements.txt
```

requirements.txt

```
annotated-types==0.7.0
anyio==4.10.0
click==8.2.1
exceptiongroup==1.3.0
fastapi==0.116.1
h11==0.16.0
idna==3.10
pydantic==2.11.7
pydantic_core==2.33.2
PyYAML==6.0.2
sniffio==1.3.1
starlette==0.47.2
typing-inspection==0.4.1
typing_extensions==4.14.1
uvicorn==0.35.0
```

4. Запустите приложение:

```
$ python server.py
```

Вывод:

```
INFO:     Started server process [26147]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8003 (Press CTRL+C to quit)
...
```

Выполнение сценария

1. Убедитесь, что тестовый сервер запущен и работает.
2. В ADCM активируйте встроенный каталог Iceberg (***Clusters** → **<clusterName>** → **Services** → **Trino** → **Iceberg configuration***).
3. (Опционально) Включите динамическое управление каталогами (***Clusters** → **<clusterName>** → **Services** → **Trino** → **Components** → **Trino Coordinator** → **Trino catalog management** → **catalog.management***). Это позволит создавать каталоги с помощью команды `CREATE CATALOG`. Перезапустите сервис Trino.
4. Подключитесь к Trino Coordinator с помощью DBeaver. Строка подключения JDBC доступна в ADCM (***Clusters** → **<clusterName>** → **Services** → **Trino** → **Info***). Для подключения используйте имя пользователя, имеющего права на запись в HDFS. Это необходимо для выполнения определенных операций записи каталогом Iceberg на последующих шагах (запись сырых данных в HDFS).
5. В DBeaver выполните следующий SQL для создания Trino-каталога:

```
CREATE CATALOG "openapi-catalog" USING openapi
WITH (
    "spec-location" = 'http://10.92.41.251:8003/openapi.yaml', ①
    "base-uri" = 'http://10.92.41.251:8003', ②
    "authentication.type" = 'none' ③
);
```

① Эндпойнт, который возвращает OpenAPI-схему тестового сервера. В данном сценарии схема генерируется с помощью FastAPI.

② Базовый URL REST API сервера.

③ В данном примере аутентификация не используется. Дополнительные сведения о поддерживаемых типах аутентификации доступны в статье Настройка OpenAPI connector.

6. Проверьте созданные каталоги:

```
SHOW CATALOGS;
```

Вывод:

```
Catalog             |
--------------------+
iceberg-adh         |
system              |
openapi-catalog     |
```

7. Создайте тестовую таблицу Iceberg:

```
USE "iceberg-adh".default;
CREATE TABLE transactions_spb_only(txn_id int, acc_id int, txn_value double);
```

8. Запросите список транзакций из тестового сервера, отфильтруйте их по типу и запишите результат в таблицу Iceberg с помощью SQL-команды:

```
INSERT INTO "iceberg-adh".default.transactions_spb_only (txn_id, acc_id, txn_value)
SELECT txn_id, acc_id, txn_value
FROM "openapi-catalog".default.transactions
WHERE txn_type = 'spb';
```

9. Проверьте данные в таблице Iceberg:

```
SELECT * FROM "iceberg-adh".default.transactions_spb_only;
```

Пример вывода:

```
txn_id|acc_id|txn_value|
------+------+---------+
     1|  1001|   275.32|
     2|  1001|    12.24|
     6|  1000|    89.72|
     7|  1000|    40.71|
     8|  1002|   470.56|
```

Проверьте запись данных в HDFS:

```
$ hdfs dfs -ls /apps/hive/warehouse/transactions_spb_only-7947dbdc9e8d4cf6b5db7558a64f16c5
```

Пример вывода:

```
drwxrwxr-x   - hdfs hadoop          0 2025-09-17 16:01 hdfs://adh/apps/hive/warehouse/transactions_spb_only-
7947dbdc9e8d4cf6b5db7558a64f16c5/data
drwxrwxr-x   - hdfs hadoop          0 2025-09-17 16:01 hdfs://adh/apps/hive/warehouse/transactions_spb_only-
7947dbdc9e8d4cf6b5db7558a64f16c5/metadata
```

Настройка OpenAPI connector

Константин Алпашкин

Содержание

Назад к содержанию

Распаковка массивов

Пагинация

Для настройки коннектора используются параметры, описанные в данной статье. Эти параметры можно указать при создании нового Trino-каталога с помощью ADCM или внутри предложения `WITH`, например:

```
CREATE CATALOG "openapi-catalog" USING openapi
WITH (
    "spec-location" = ...,
    "base-uri" = ...,
    ...
);
```

Параметр	Описание
spec-location	URL или имя файла, содержащего схему OpenAPI (JSON/YAML)
base-uri	Базовый URL REST API
authentication.type	Тип аутентификации по умолчанию, который будет использоваться, если таковой не указан в схеме OpenAPI. Возможные значения: `none`, `http`, `api_key`, `oauth`
authentication.scheme	Схема аутентификации для типа аутентификации `http`. Возможные значения: `basic`, `bearer`
authentication.client-id	OAuth Client ID
authentication.client-secret	OAuth Client secret
authentication.username	Имя пользователя для типов аутентификации `http` и `oauth`
authentication.password	Пароль для типов аутентификации `http` и `oauth`
authentication.bearer-token	Bearer-токен для аутентификации `http`
authentication.api-key-name	Имя ключа API
authentication.api-key-value	Значение ключа API
max-requests-per-second	Максимальное количество HTTP-запросов в секунду, которые могут быть отправлены с одного узла Trino
max-splits-per-second	Максимальное количество сплитов (split) в секунду, генерируемых при выполнении запроса

Под капотом коннектор использует Airlift HTTP client, который поддерживает свой собственный набор параметров, например:

```
openApi.http-client.log.enabled=true
openApi.http-client.log.path=logs
```

Подробная информация о настройке клиента доступна на Github-странице проекта.

Распаковка массивов

При анализе HTTP-ответов от API коннектор может автоматически "распаковывать" поля с массивами. Например, коннектор получает следующий HTTP-ответ:

```
{
    "transactions_count": 10,
    "items":
    [
        {
            "txn_id": 1,
            "acc_id": 1002,
            "txn_value": 184.0,
            "txn_type": "withdrawal"
        },
        {
            "txn_id": 2,
            "acc_id": 1000,
            "txn_value": 282.95,
            "txn_type": "spb"
        },
        {
            "txn_id": 3,
            "acc_id": 1001,
            "txn_value": 100.00,
            "txn_type": "deposit"
        }
    ]
}
```

По умолчанию Trino преобразует поле `items` в одну строку таблицы, содержащую все элементы массива:

```
total_count|                                         items|
-----------+----------------------------------------------+
         10|[{"txn_id": 1, ...}, {"txn_id": 1, ...}, ...]|
```

Чтобы Trino "развернул" такой массив и представил его содержимое клиенту в виде отдельной таблицы, укажите блок `x-unwrap` в OpenAPI-схеме. Например, для вышеуказанного HTTP-ответа блок `x-unwrap` выглядит следующим образом:

```
/transactions_batch:
   get:
     summary: Get Transactions Batch
     operationId: get_transactions_batch_transactions_batch_get
     responses:
       '200':
         description: Successful Response
         content:
           application/json:
             schema:
               $ref: '#/components/schemas/TransactionBatch'
     x-unwrap:
       resultParam: $response.body#/items ①
       includeRoot: 'false' ②
```

① Поле HTTP-ответа, содержащее массив, который необходимо развернуть и отобразить в виде отдельной таблицы.

② Указывает, следует ли включать дублирующиеся поля (`transactions_count`).

При такой конфигурации `x-unwrap` коннектор преобразует содержимое поля `items` в отдельную таблицу:

```
txn_id|acc_id|txn_value|txn_type  |
------+------+---------+----------+
     1|  1002|184.00   |withdrawal|
     2|  1000|282.95   |spb       |
     3|  1001|100.00   |deposit   |
```

Пагинация

Коннектор поддерживает пагинацию для pushdown-предиката `LIMIT`. Чтобы активировать пагинацию, укажите блок `x-pagination` на нужном эндпойнте в схеме OpenAPI:

```
x-pagination:
  pageParam: "page"
  limitParam: "size"
  resultsPath: "$response.body#/elements"
  totalResultsPath: "$response.body#/totalCount"
```

Использование пагинации не рекомендуется, так как это может привести к некорректным результатам, особенно при работе с большими наборами данных.

Аутентификация LDAP в Trino

Сергей Тихомиров

Содержание

Назад к содержанию

Активация LDAP

Проверка

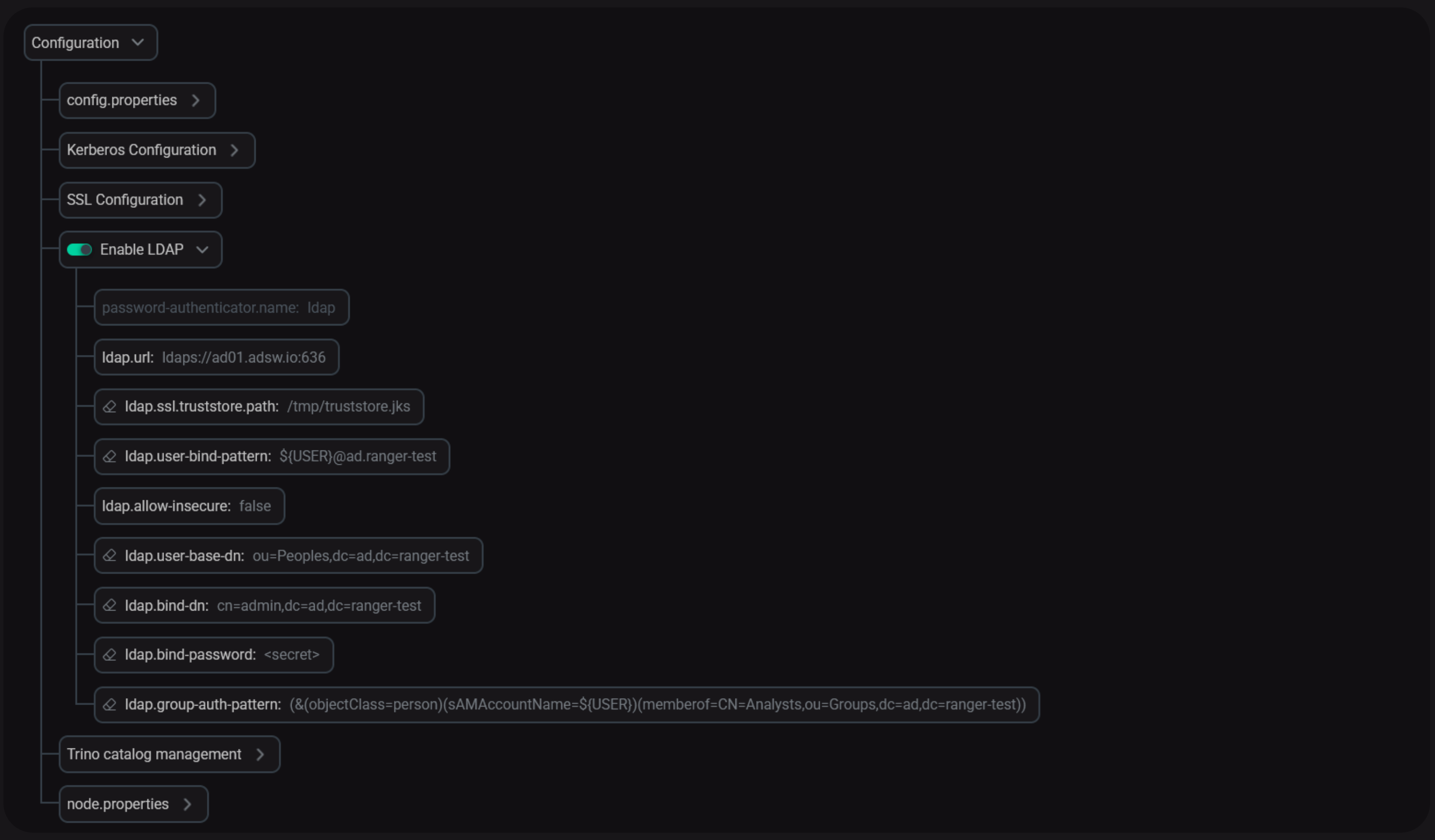
Активация LDAP

ПРИМЕЧАНИЕ

LDAP-аутентификация работает только в кластерах с активированным SSL.

Для активации LDAP-аутентификации следуйте шагам ниже:

1. На странице ***Clusters*** выберите ваш кластер ADH и перейдите на вкладку ***Services*** в меню кластера.
2. Выберите сервис ***Trino*** и перейдите на вкладку ***Components***.
3. Выберите компонент ***Trino Coordinator*** и активируйте группу параметров ***Enable LDAP***.
4. Заполните параметры LDAP для Active Directory или FreeIPA и перезапустите ***Trino Coordinator***.



password-authenticator.name: ldap
ldap.url: ldaps://ad01.adsw.io:636
ldap.ssl.truststore.path: /tmp/truststore.jks
ldap.user-bind-pattern: ${USER}@ad.ranger-test
ldap.allow-insecure: false
ldap.user-base-dn: ou=Peoples,dc=ad,dc=ranger-test
ldap.bind-dn: cn=admin,dc=ad,dc=ranger-test
ldap.bind-password: <secret>
ldap.group-auth-pattern: (&(objectClass=person)(sAMAccountName=${USER})(memberof=CN=Analysts,ou=Groups,dc=ad,dc=ranger-test))

Конфигурация LDAP с AD

Проверка

Для проверки правильности работы вы можете использовать утилиту Trino CLI.

1. Подключитесь к Trino Coordinator с указанием дополнительных параметров:

```
$ ./trino https://<coordinator_host>:18188 --keystore-path <keystore-path> --keystore-password <keystore_password> --truststore-path <truststore-path> --truststore-password <truststore_password> --user=<username> --password
```

2. Введите пароль.
3. В консоли Trino запустите команду `SHOW CATALOGS;`. Её вывод должен быть похож на следующий:

```
Catalog
-------------
hive-adh
iceberg-adh
system
(3 rows)

Query 20250306_005901_00001_6smm5, FINISHED, 1 node
Splits: 7 total, 7 done (100.00%)
0.28 [0 rows, 0B] [0 rows/s, 0B/s]
```

Аутентификация Kerberos в Trino

Сергей Тихомиров

Назад к содержанию

Kerberos-аутентификация для сервиса Trino требует выполнения дополнительных действий, которые зависят от того, включен плагин Ranger Trino или нет. В данной статье рассматриваются оба случая.

Для настройки Kerberos-аутентификации для Trino следуйте шагам ниже:

1. Активируйте SSL для вашего кластера ADH.
2. Активируйте Kerberos для вашего кластера ADH.
3. Перейдите в настройки компонента Trino Coordinator и измените файл-параметр ***user-mapping.json***, чтобы его содержимое было следующим:

```
{
  "rules": [
    {
      "pattern": "(.+)@(.*)"
    },
    {
      "pattern": "^(.*?)/.*$"
    }
  ]
}
```

4. Настройте имперсонацию и права доступа.

С плагином Ranger

1. В Ranger создайте пользователя для принципала Kerberos, который будет выполнять команды Trino CLI.
2. Добавьте этого пользователя в следующие политики Trino: `all - trinouser`, `all - catalog, schema, table, column` и `all - queryid`.

Без плагина Ranger

На странице настройки сервиса Trino в ADCM раскройте группу параметров ***access-control.properties*** и измените параметр ***rules_json_patter***, чтобы он включал в себя настройки имперсонации и прав. Например, следующим образом:

```
{
  "system": {
    "impersonation": [
      {
        "user": "<user with permissions, like `trino`>",
        "allow": {
          "user": "<user who will run trino-cli>"
        }
      }
    ]
  },
  "catalogs": [
    {
      "role": "admin",
      "catalog": "hive-adh",
      "allow": "all"
    },
    {
      "user": "trino",
      "catalog": "iceberg-adh",
      "allow": "all"
    },
    {
      "user": "<user who will run trino-cli>",
      "catalog": "iceberg-adh",
      "allow": "all"
    }
  ]
}
```

Проверьте корректность настройки с помощью запуска команды Trino CLI:

```
$ trino --execute 'select * from "hive-adh"."default"."solar_objects";'
```

Ожидаемым результатом является содержимое таблицы:

```
"1","Sun","1989100000","1392000"
"2","Mercury","330","4879"
"3","Venus","4867","12104"
"4","Earth","5972","12742"
"5","Mars","642","6780"
"6","Jupiter","1898187","139822"
"7","Saturn","568317","116464"
"8","Uranus","86813","50724"
"9","Neptune","102413","49244"
```

Авторизация в Ranger с политиками Hive

Сергей Тихомиров

Содержание

Назад к содержанию

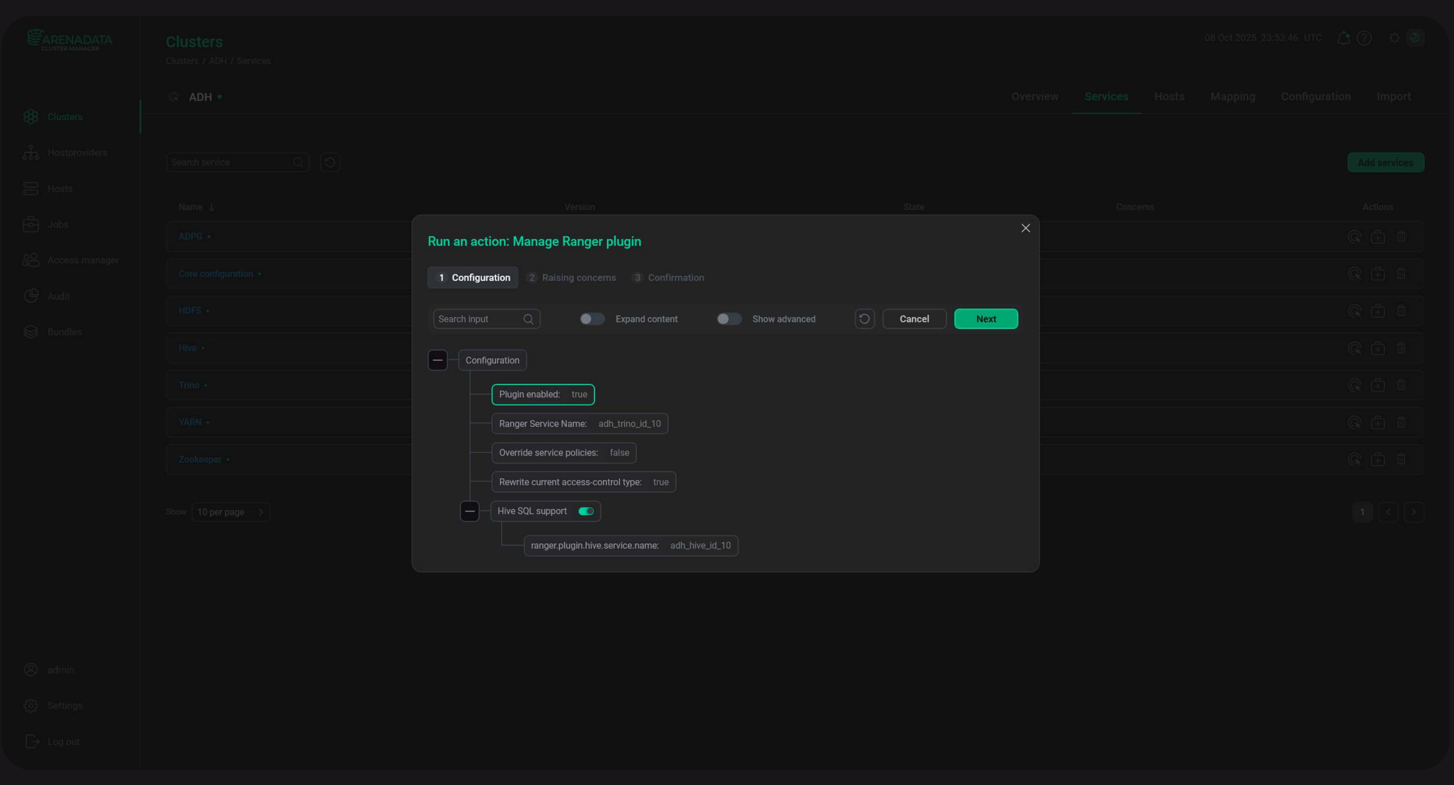
Настройка

Проверка

Плагин Ranger Trino может быть настроен для политик Hadoop SQL. В данной статье описывается, как провести настройку и протестировать данную возможность.

Настройка

1. Активируйте плагин Ranger Hive.
2. Активируйте плагин Ranger Trino со включенным параметром ***Hive SQL Support***. Убедитесь, что значение поля ***ranger.plugin.hive.service.name*** совпадает с названием сервиса Hive в Ranger.



Run an action: Manage Ranger plugin

Configuration

Plugin enabled: true

Ranger Service Name: adh_trino_id_10

Override service policies: false

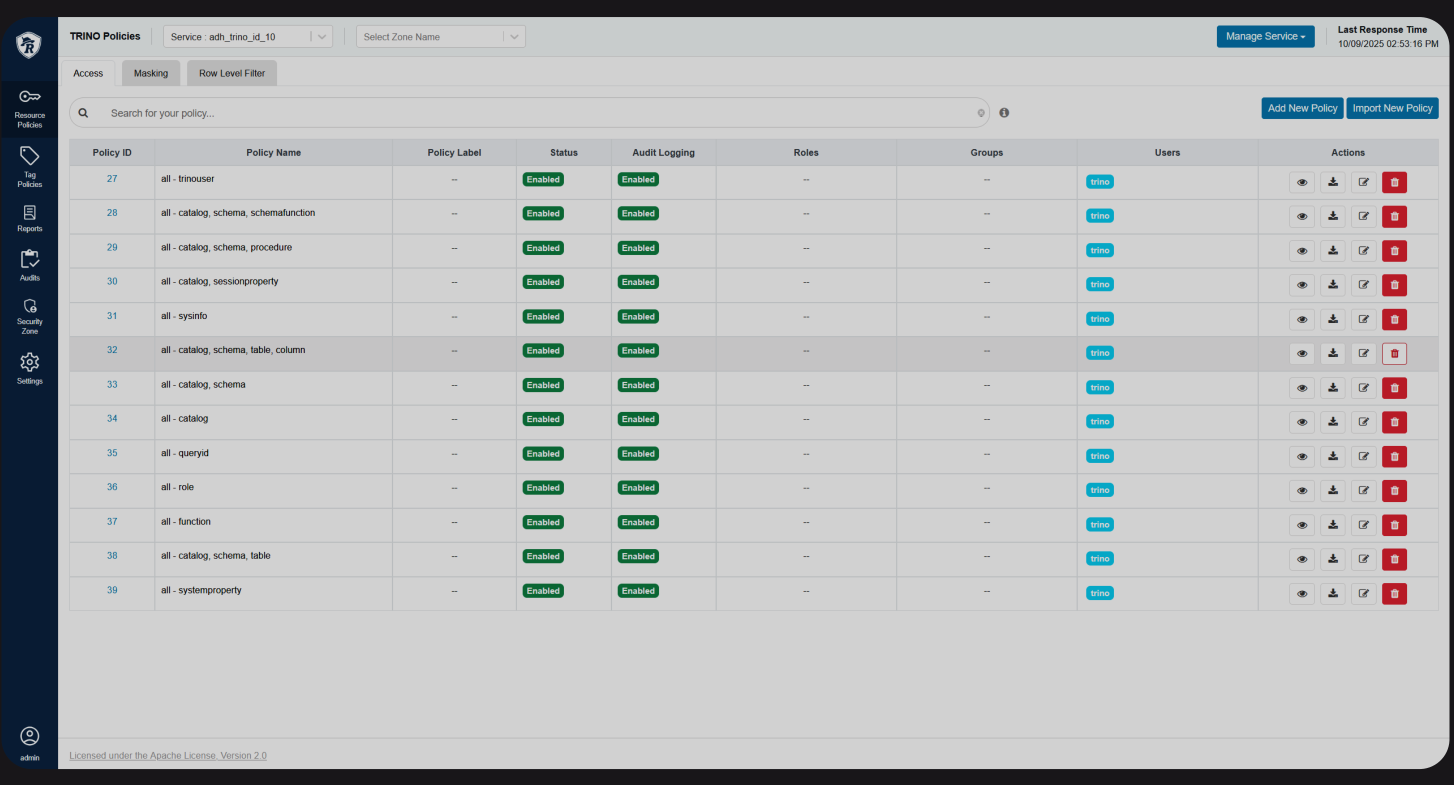
Rewrite current access-control type: true

Hive SQL support

ranger.plugin.hive.service.name: adh_hive_id_10

Параметры плагина Ranger Trino

3. Удалите политику `all - catalog, schema, table, column` для сервиса Trino в Ranger.



TRINO Policies

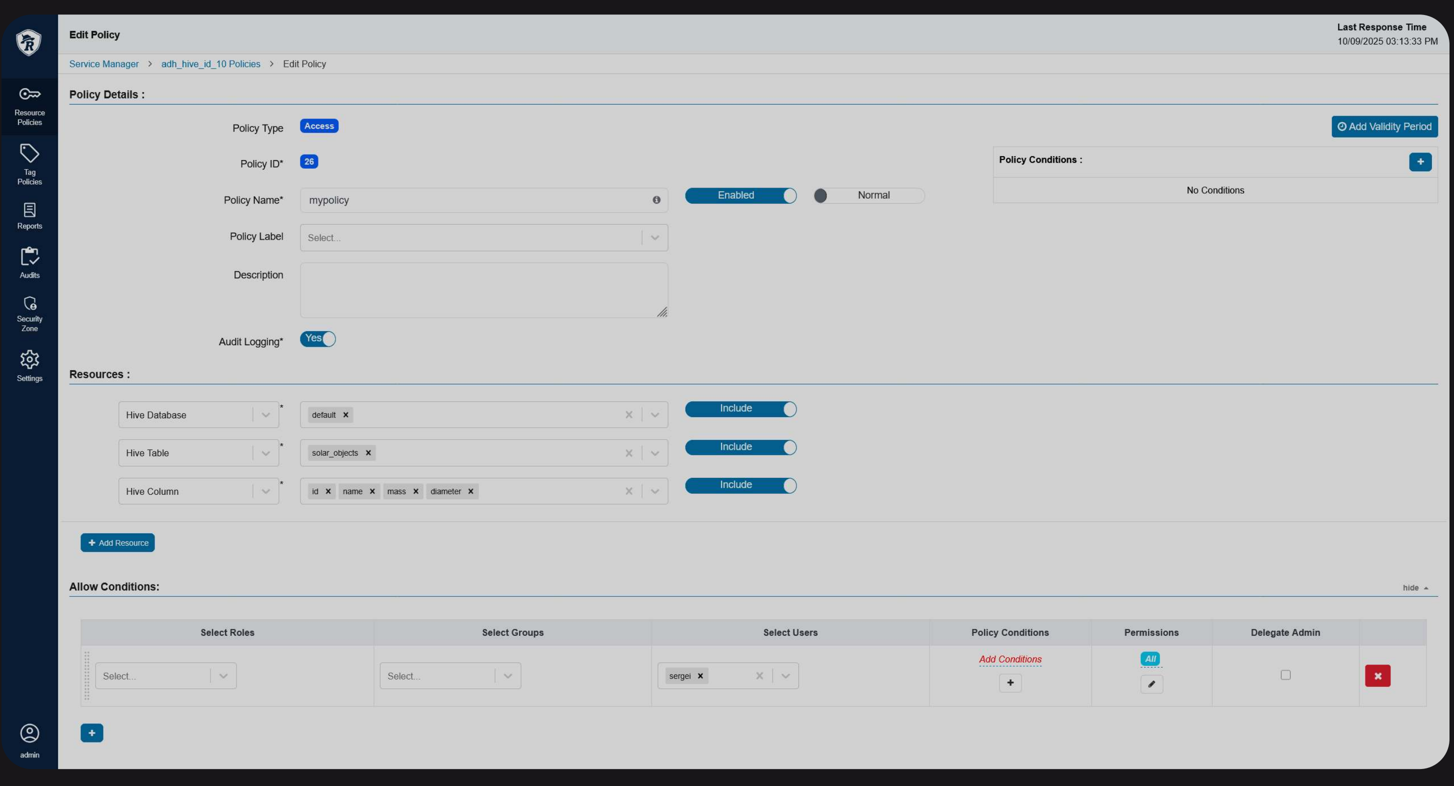
Service : adh_trino_id_10

Last Response Time 10/09/2025 02:53:16 PM

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users
27	all - trinouser	--	Enabled	Enabled	--	--	trino
28	all - catalog, schema, schemafunction	--	Enabled	Enabled	--	--	trino
29	all - catalog, schema, procedure	--	Enabled	Enabled	--	--	trino
30	all - catalog, sessionproperty	--	Enabled	Enabled	--	--	trino
31	all - sysinfo	--	Enabled	Enabled	--	--	trino
32	all - catalog, schema, table, column	--	Enabled	Enabled	--	--	trino
33	all - catalog, schema	--	Enabled	Enabled	--	--	trino
34	all - catalog	--	Enabled	Enabled	--	--	trino
35	all - queryid	--	Enabled	Enabled	--	--	trino
36	all - role	--	Enabled	Enabled	--	--	trino
37	all - function	--	Enabled	Enabled	--	--	trino
38	all - catalog, schema, table	--	Enabled	Enabled	--	--	trino
39	all - systemproperty	--	Enabled	Enabled	--	--	trino

Удаление политики в Trino

4. Добавьте пользователя, который должен иметь возможность авторизации, в следующие политики Trino: `all - trinouser`, `all - catalog, schema` и `all - queryid`.
5. В сервисе Hive создайте политику, позволяющую пользователю иметь доступ ко всем колонкам определенной таблицы в базе данных.



Edit Policy

Service Manager > adh_hive_id_10 Policies > Edit Policy

Policy Type: Access

Policy ID: 26

Policy Name: mypolicy

Audit Logging: Yes

Hive Database: default

Hive Table: solar_objects

Hive Column: id, name, mass, diameter

Select Users: sergei

Permissions: All

Создание политики для Hive

Проверка

Чтобы проверить корректность настройки, следуйте шагам ниже:

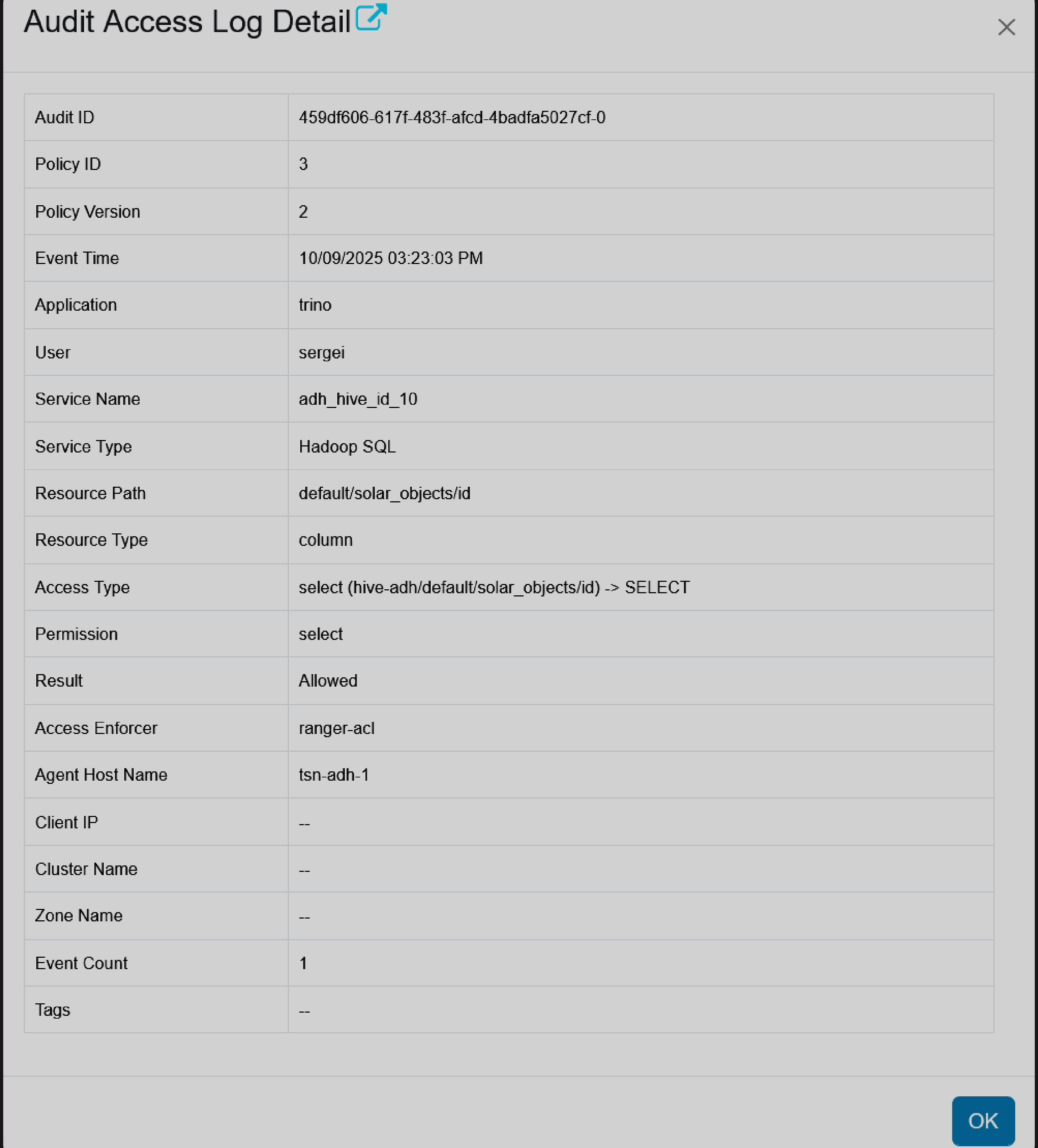
1. На хосте кластера ADH выполните запрос с помощью Trino CLI:

```
$ trino --execute 'select * from "hive-adh"."default"."solar_objects";'
```

Ожидаемым результатом является содержимое таблицы:

```
"1","Sun","1989100000","1392000"
"2","Mercury","330","4879"
"3","Venus","4867","12104"
"4","Earth","5972","12742"
"5","Mars","642","6780"
"6","Jupiter","1898187","139822"
"7","Saturn","568317","116464"
"8","Uranus","86813","50724"
"9","Neptune","102413","49244"
```

2. Чтобы убедиться в корректности работы имперсонации, в веб-интерфейсе Ranger Admin откройте страницу ***Audits*** → ***Access*** и кликните на строку с вашим запросом.



Audit Access Log Detail

Audit ID	459df606-617f-483f-afcd-4badfa5027cf-0
Policy ID	3
Policy Version	2
Event Time	10/09/2025 03:23:03 PM
Application	trino
User	sergei
Service Name	adh_hive_id_10
Service Type	Hadoop SQL
Resource Path	default/solar_objects/id
Resource Type	column
Access Type	select (hive-adh/default/solar_objects/id) -> SELECT
Permission	select
Result	Allowed
Access Enforcer	ranger-acl
Agent Host Name	tsn-adh-1
Client IP	--
Cluster Name	--
Zone Name	--
Event Count	1
Tags	--

Страница Ranger Audits

Управление сервисом Trino через ADCM

Владимир Адаменков

Содержание

Назад к содержанию

Обзор

Обзор

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Trino. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Trino, перечислены в таблице ниже.

Действия с сервисом Trino

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавлять, удалять и распределять компоненты Trino
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются. Для данного действия доступна дополнительная опция ***Init DB credentials***. Если опция активна, в сервисе ADPG, используемом DBCatalogManager, будут созданы необходимые объекты (база данных, пользователь, роли)
Manage Ranger plugin	Включает или отключает плагин Ranger для Trino

Сервис Trino включает в себя следующие компоненты:

- Trino Coordinator
- Trino Worker
- Trino Client
- HAProxy Trino

Большинство компонентов имеют одинаковый набор действий, описанных в таблице ниже.

Действия с компонентами Trino

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

Действия Trino Client

Действие	Описание
Apply client configs	Применяет конфигурацию Trino Client, указанную в ADCM, на всех хостах, где установлен компонент

Обзор UI YARN

Елена Костюченко

Содержание

Назад к содержанию

MapReduce History Server
NodeManager
ResourceManager
Timeline server

Веб-интерфейсы (UI) YARN предоставляют диагностическую информацию и обеспечивают доступ к логам, конфигурации, состоянию заданий и компонентов YARN. Пользовательские интерфейсы доступны по URL-адресу *http://<имя_хоста>:<порт>/*, где `<имя_хоста>` — это IP-адрес хоста или, если вы настроили перенаправление в файле *hosts*, FQDN хоста. А `<порт>` — это порт пользовательского интерфейса компонента YARN.

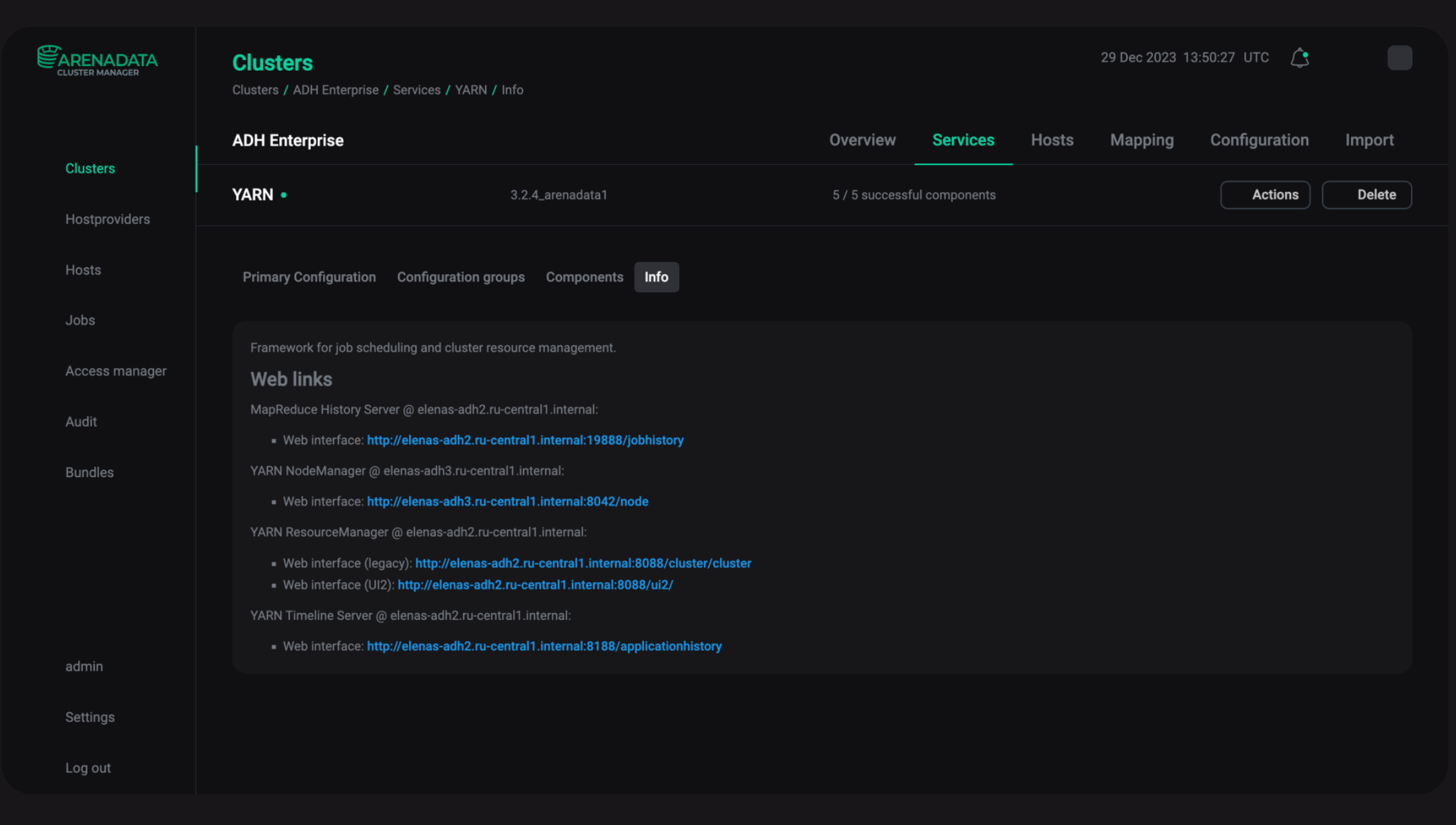
Если у вас настроен сертификат SSL, используйте *https://* вместо *http://*.

Веб-интерфейсы YARN

Компонент	Адрес
MapReduce History Server	http://<IP>:19888/jobhistory
YARN NodeManager	http://<IP>:8042/node
YARN ResourceManager (legacy)	http://<IP>:8088/cluster/cluster
YARN ResourceManager (UI2)	http://<IP>:8088/ui2/
YARN Timeline Server	http://<IP>:8188/applicationhistory/

Ссылки на пользовательские интерфейсы YARN также доступны в ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите ***YARN***.
3. Выберите ***Info***.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH Enterprise / Services / YARN / Info
29 Dec 2023 13:50:27 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
ADH Enterprise
Overview
Services
Hosts
Mapping
Configuration
Import
YARN
3.2.4_arenadata1
5 / 5 successful components
Actions
Delete
Primary Configuration
Configuration groups
Components
Info

Framework for job scheduling and cluster resource management.

Web links

MapReduce History Server @ elenas-adh2.ru-central1.internal:
- Web interface: http://elenas-adh2.ru-central1.internal:19888/jobhistory

YARN NodeManager @ elenas-adh3.ru-central1.internal:
- Web interface: http://elenas-adh3.ru-central1.internal:8042/node

YARN ResourceManager @ elenas-adh2.ru-central1.internal:
- Web interface (legacy): http://elenas-adh2.ru-central1.internal:8088/cluster/cluster
- Web interface (UI2): http://elenas-adh2.ru-central1.internal:8088/ui2/

YARN Timeline Server @ elenas-adh2.ru-central1.internal:
- Web interface: http://elenas-adh2.ru-central1.internal:8188/applicationhistory

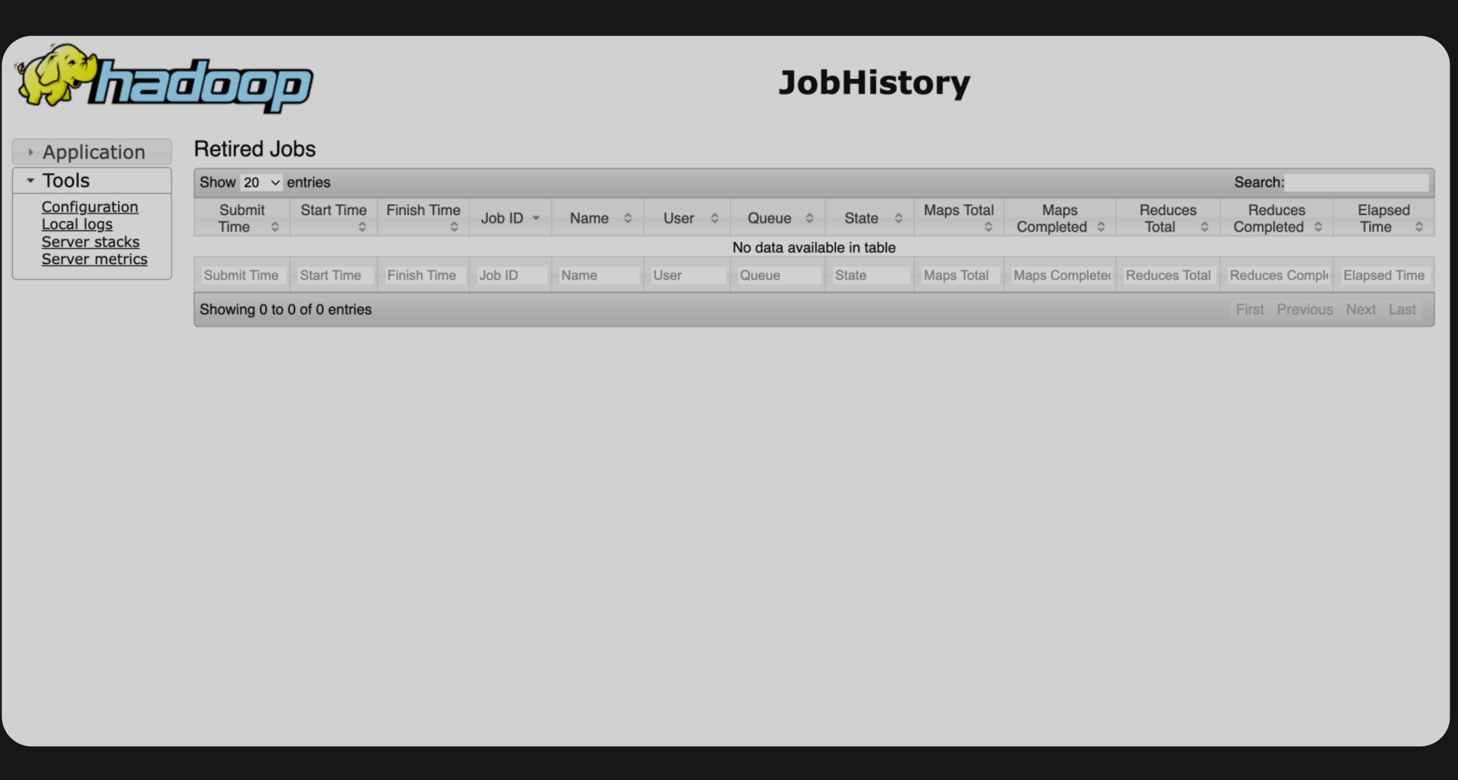
Ссылки на веб-интерфейс YARN в ADCM

MapReduce History Server

На главной странице пользовательского интерфейса MapReduce History Server слева расположено навигационное меню с двумя разделами: ***Application***, которое содержит информацию о версии сборки и времени запуска сервера, и ***Tools***.

Раздел ***Tools*** содержит ссылки на следующую информацию:

- ***Configuration*** — XML-файл, содержащий файлы конфигурации HDFS и YARN;
- ***Local logs*** — список логов MapReduce History Server, расположенных на хосте;
- ***Server stacks*** — список активных потоков и их статус;
- ***Server metrics*** — список метрик JMX.


hadoop
JobHistory
Application
Tools
Configuration
Local logs
Server stacks
Server metrics
Retired Jobs
Show 20 entries
Search:
First Previous Next Last

Submit Time	Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Completed	Reduces Total	Reduces Completed	Elapsed Time
No data available in table												

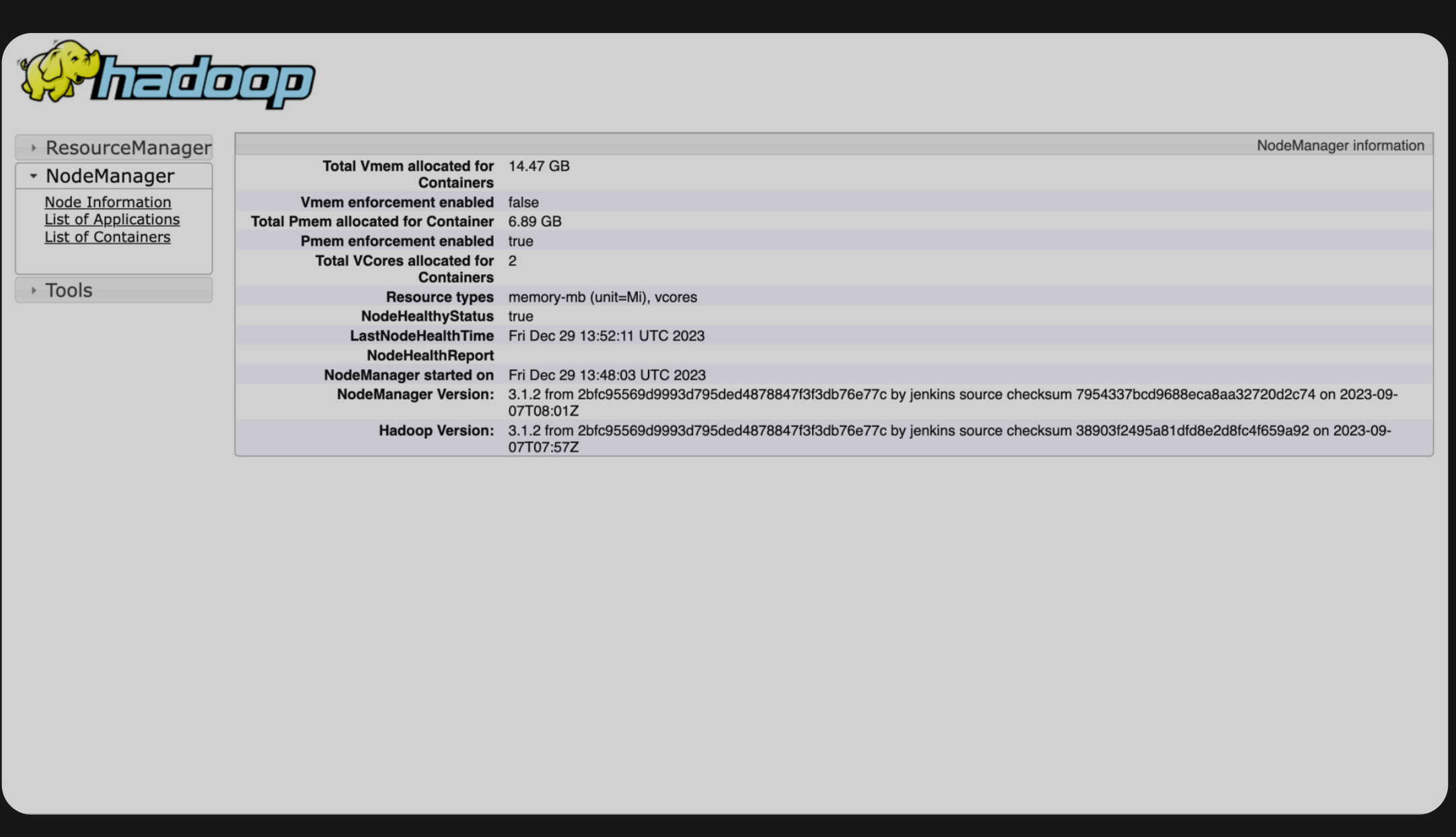
Showing 0 to 0 of 0 entries

MapReduce History Server: главная страница

NodeManager

На главной странице пользовательского интерфейса NodeManager слева расположено навигационное меню с тремя разделами:

- ***ResourceManager*** — содержит ссылку на пользовательский интерфейс ResourceManager.
- ***NodeManager*** — содержит информацию о NodeManager, список запущенных приложений и список запущенных контейнеров.
- ***Tools*** — содержит ссылки на следующую информацию:
  - ***Configuration*** — XML-файл, содержащий файлы конфигурации HDFS и YARN;
  - ***Local logs*** — логи NodeManager, расположенные на хосте;
  - ***Server stacks*** — список активных потоков и их статус;
  - ***Server metrics*** — список метрик JMX.


hadoop
ResourceManager
NodeManager
Node Information
List of Applications
List of Containers
Tools
NodeManager information

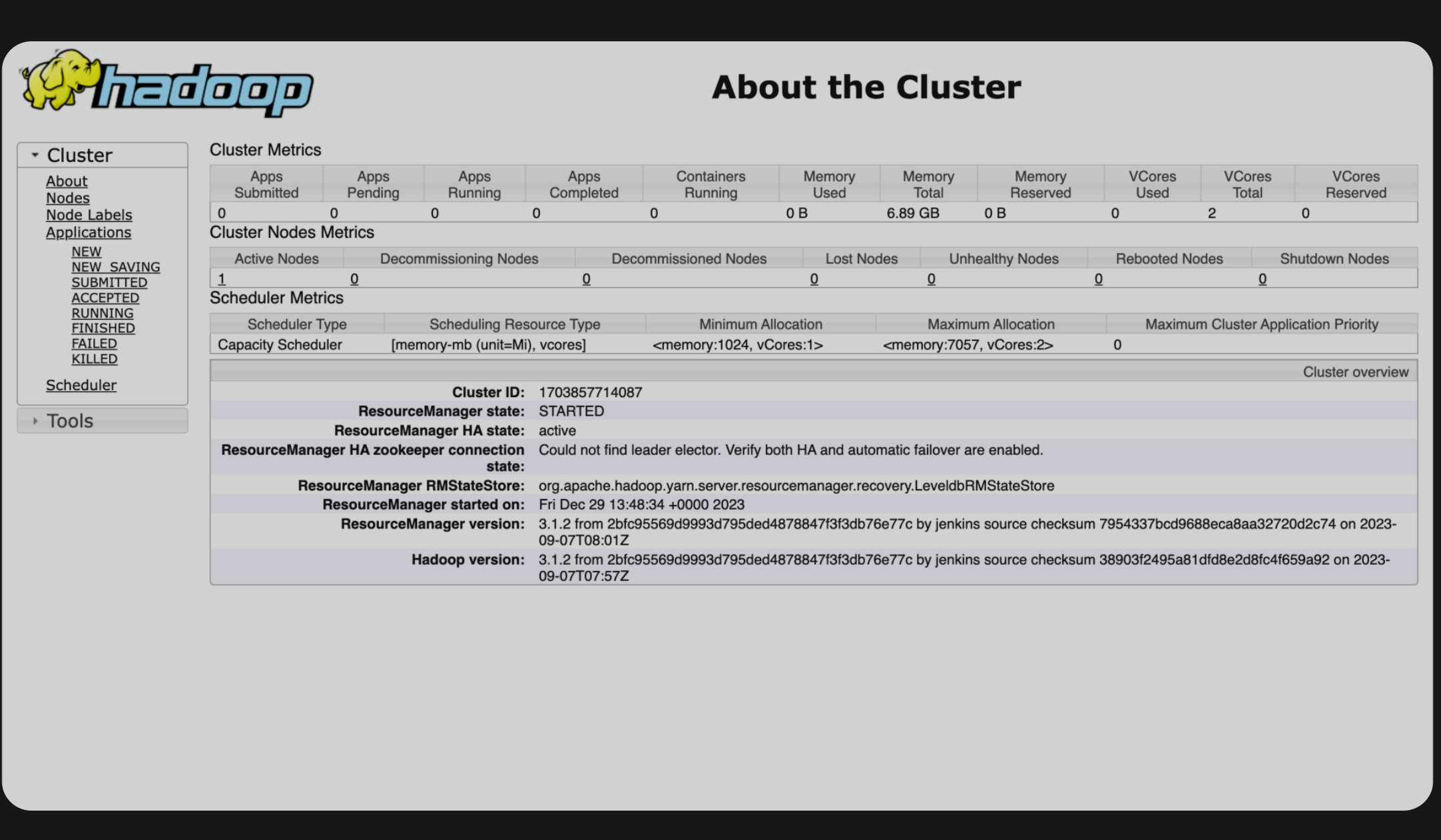
Total Vmem allocated for Containers: 14.47 GB
Vmem enforcement enabled: false
Total Pmem allocated for Container: 6.89 GB
Pmem enforcement enabled: true
Total VCores allocated for Containers: 2
Resource types: memory-mb (unit=Mi), vcores
NodeHealthyStatus: true
LastNodeHealthTime: Fri Dec 29 13:52:11 UTC 2023
NodeHealthReport:
NodeManager started on: Fri Dec 29 13:48:03 UTC 2023
NodeManager Version: 3.1.2 from 2bfc95569d9993d795ded4878847f3f3db76e77c by jenkins source checksum 7954337bcd9688eca8aa32720d2c74 on 2023-09-07T08:01Z
Hadoop Version: 3.1.2 from 2bfc95569d9993d795ded4878847f3f3db76e77c by jenkins source checksum 38903f2495a81dfd8e2d8fc4f659a92 on 2023-09-07T07:57Z

NodeManager UI: главная страница

ResourceManager

У ResourceManager есть два интерфейса:

- Устаревший (Legacy UI): *http://<IP>:8088/cluster/cluster*.
- Новый (UI2): *http://<IP>:8088/ui2/*.


hadoop
About the Cluster
Cluster
About
Nodes
Node Labels
Applications
NEW
NEW_SAVING
SUBMITTED
ACCEPTED
RUNNING
FINISHED
FAILED
KILLED
Scheduler
Tools
Cluster overview

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved
0	0	0	0	0	0 B	6.89 GB	0 B	0	2	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	Shutdown Nodes
1	0	0	0	0	0	0

Scheduler Metrics

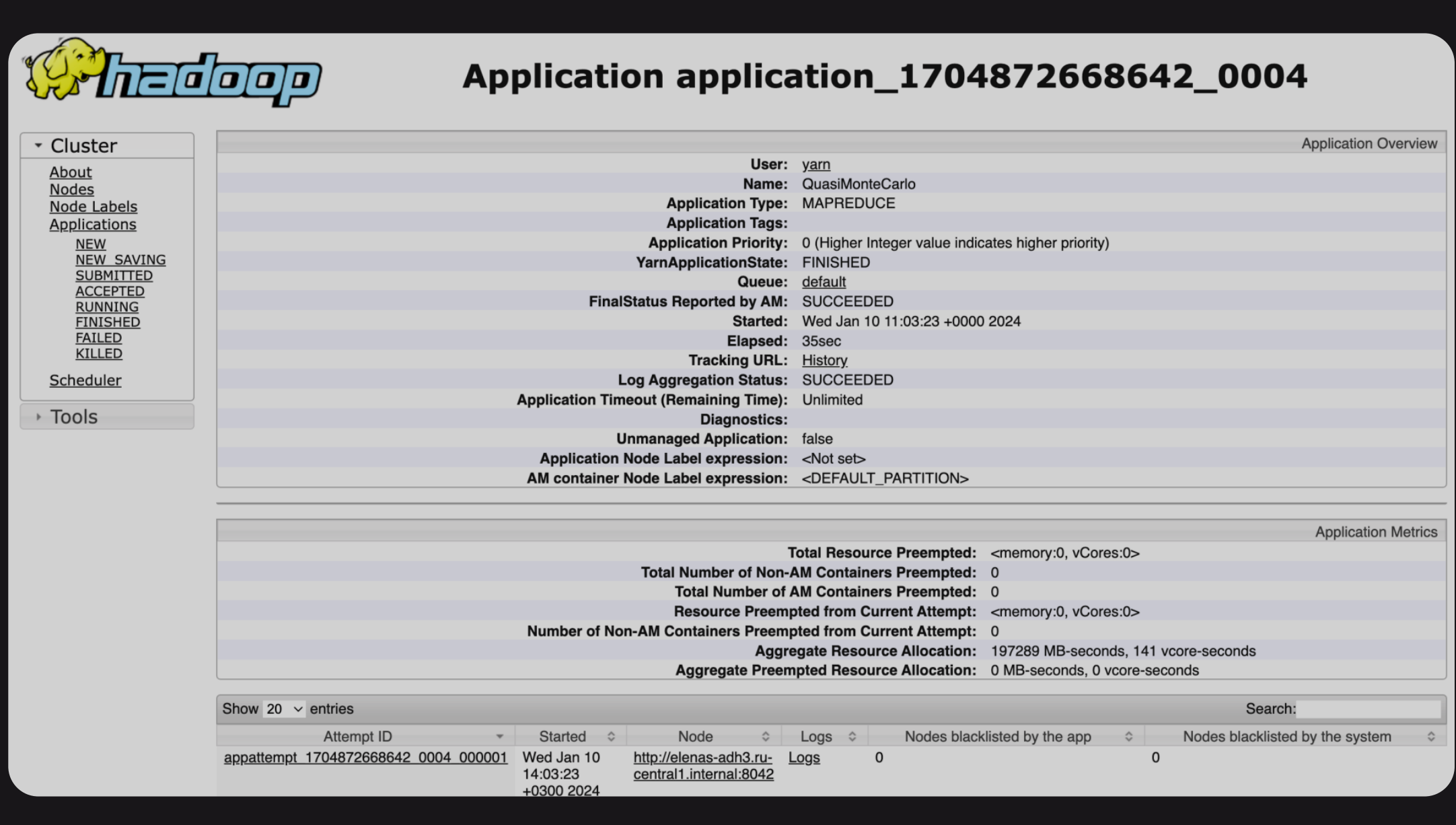
Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:7057, vCores:2>	0

Cluster ID: 1703857714087
ResourceManager state: STARTED
ResourceManager HA state: active
ResourceManager HA zookeeper connection state: Could not find leader elector. Verify both HA and automatic failover are enabled.
ResourceManager RMStateStore: org.apache.hadoop.yarn.server.resourcemanager.recovery.LeveldbRMStateStore
ResourceManager started on: Fri Dec 29 13:48:34 +0000 2023
ResourceManager version: 3.1.2 from 2bfc95569d9993d795ded4878847f3f3db76e77c by jenkins source checksum 7954337bcd9688eca8aa32720d2c74 on 2023-09-07T08:01Z
Hadoop version: 3.1.2 from 2bfc95569d9993d795ded4878847f3f3db76e77c by jenkins source checksum 38903f2495a81dfd8e2d8fc4f659a92 on 2023-09-07T07:57Z

Устаревший UI ResourceManager: главная страница

Оба интерфейса содержат информацию о кластере:

- ***About*** — информация о кластере, список его метрик, а также метрики хостов с компонентами NodeManager и метрики планировщика.
- ***Nodes*** — информация о NodeManager и таблица метрик из раздела ***About***.
- ***Node Labels*** — метки узлов кластера и информация о них.
- ***Applications*** — список приложений с информацией о них и ссылками на страницы каждого приложения.


hadoop
Application application_1704872668642_0004
Cluster
About
Nodes
Node Labels
Applications
NEW
NEW_SAVING
SUBMITTED
ACCEPTED
RUNNING
FINISHED
FAILED
KILLED
Scheduler
Tools
Application Overview
Application Metrics
Show 20 entries
Search:

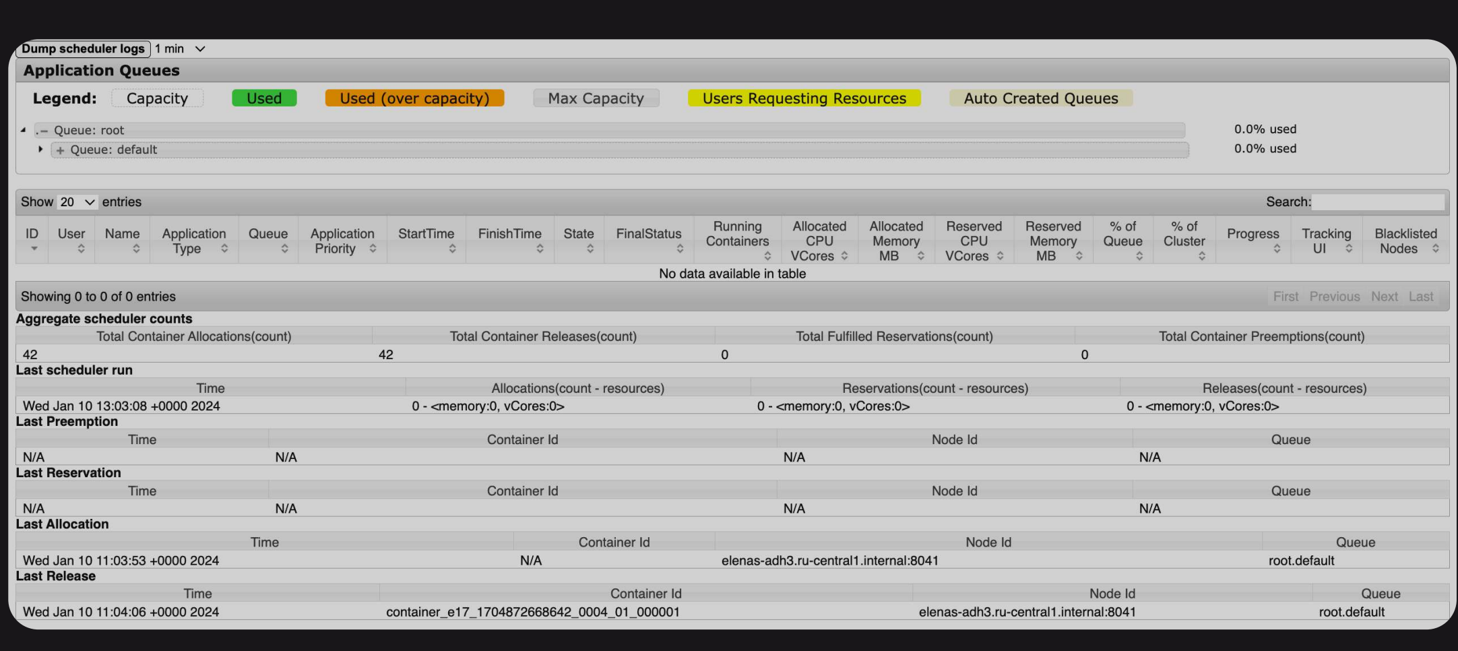
User: yarn
Name: QuasiMonteCarlo
Application Type: MAPREDUCE
Application Tags:
Application Priority: 0 (Higher Integer value indicates higher priority)
YarnApplicationState: FINISHED
Queue: default
FinalStatus Reported by AM: SUCCEEDED
Started: Wed Jan 10 11:03:23 +0000 2024
Elapsed: 35sec
Tracking URL: History
Log Aggregation Status: SUCCEEDED
Application Timeout (Remaining Time): Unlimited
Diagnostics:
Unmanaged Application: false
Application Node Label expression: <Not set>
AM container Node Label expression: <DEFAULT_PARTITION>

Total Resource Preempted: <memory:0, vCores:0>
Total Number of Non-AM Containers Preempted: 0
Total Number of AM Containers Preempted: 0
Resource Preempted from Current Attempt: <memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt: 0
Aggregate Resource Allocation: 197289 MB-seconds, 141 vcore-seconds
Aggregate Preempted Resource Allocation: 0 MB-seconds, 0 vcore-seconds

Attempt ID	Started	Node	Logs	Nodes blacklisted by the app	Nodes blacklisted by the system
appattempt_1704872668642_0004_000001	Wed Jan 10 14:03:23 +0300 2024	http://elenas-adh3.ru-central1.internal:8042	Logs	0	0

Устаревший UI ResourceManager: страница приложения

- ***Scheduler*** — информация об очередях приложений и таблица метрик из раздела ***About***.


Dump scheduler logs
1 min
Application Queues
Legend:
Capacity
Used
Used (over capacity)
Max Capacity
Users Requesting Resources
Auto Created Queues
Queue: root
0.0% used
Queue: default
0.0% used
Show 20 entries
Search:
First Previous Next Last

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Reserved CPU VCores	Reserved Memory MB	% of Queue	% of Cluster	Progress	Tracking UI	Blacklisted Nodes
No data available in table																			

Showing 0 to 0 of 0 entries

Aggregate scheduler counts

Total Container Allocations(count)	Total Container Releases(count)	Total Fulfilled Reservations(count)	Total Container Preemptions(count)
42	42	0	0

Last scheduler run

Time	Allocations(count - resources)	Reservations(count - resources)	Releases(count - resources)
Wed Jan 10 13:03:08 +0000 2024	0 - <memory:0, vCores:0>	0 - <memory:0, vCores:0>	0 - <memory:0, vCores:0>

Last Preemption

Time	Container Id	Node Id	Queue
N/A	N/A	N/A	N/A

Last Reservation

Time	Container Id	Node Id	Queue
N/A	N/A	N/A	N/A

Last Allocation

Time	Container Id	Node Id	Queue
Wed Jan 10 11:03:53 +0000 2024	N/A	elenas-adh3.ru-central1.internal:8041	root.default

Last Release

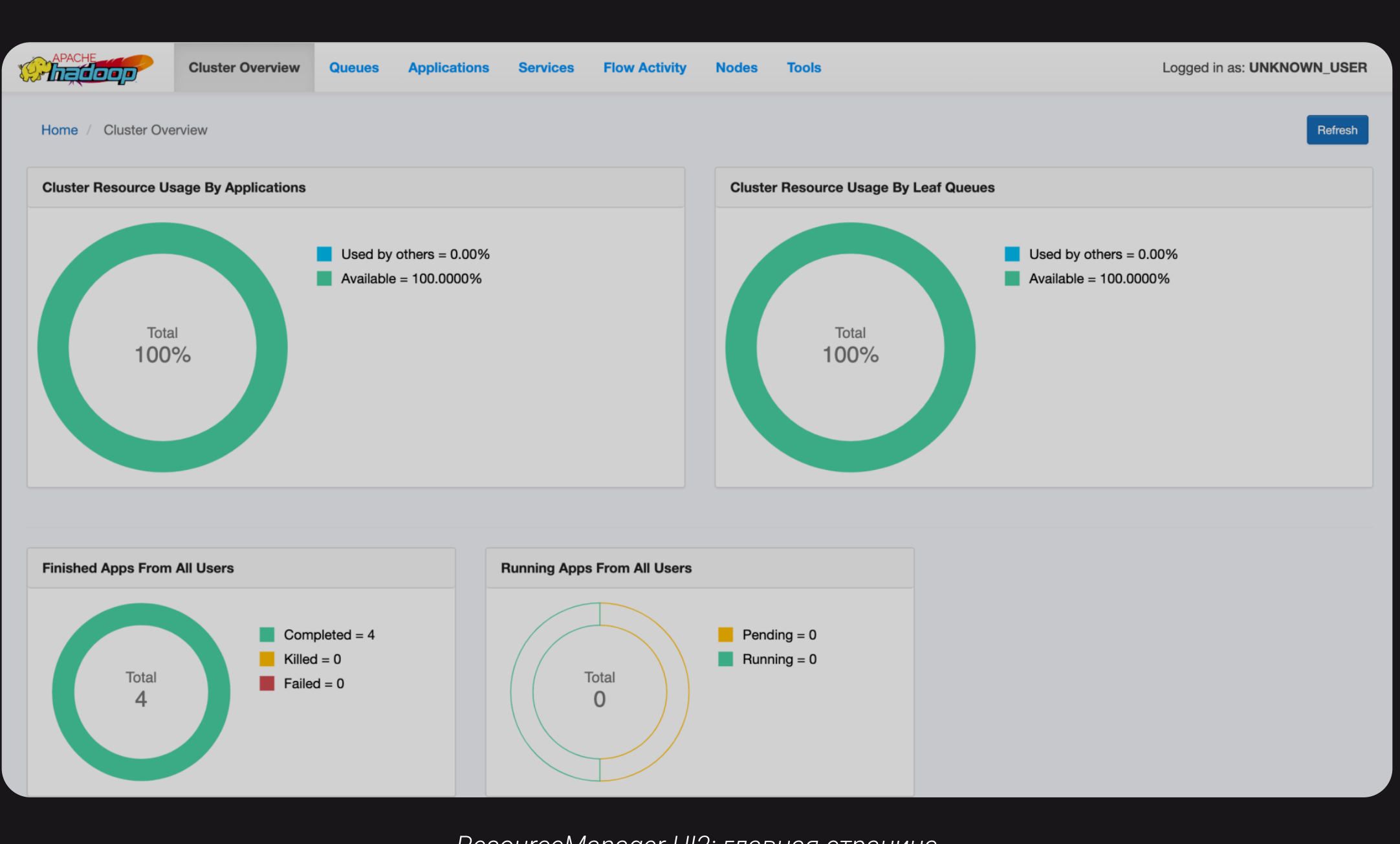
Time	Container Id	Node Id	Queue
Wed Jan 10 11:04:06 +0000 2024	container_e17_1704872668642_0004_01_000001	elenas-adh3.ru-central1.internal:8041	root.default

Устаревший UI ResourceManager: таблица очереди приложений

Как и другие веб-интерфейсы YARN, оба интерфейса ResourceManager содержат ссылки на инструменты YARN (***Tools***):

- ***Configuration*** — XML-файл, содержащий файлы конфигурации HDFS и YARN;
- ***Local logs*** — логи YARN, расположенные на хосте;
- ***Server stacks*** — список активных потоков и их статус;
- ***Server metrics*** — список метрик JMX.

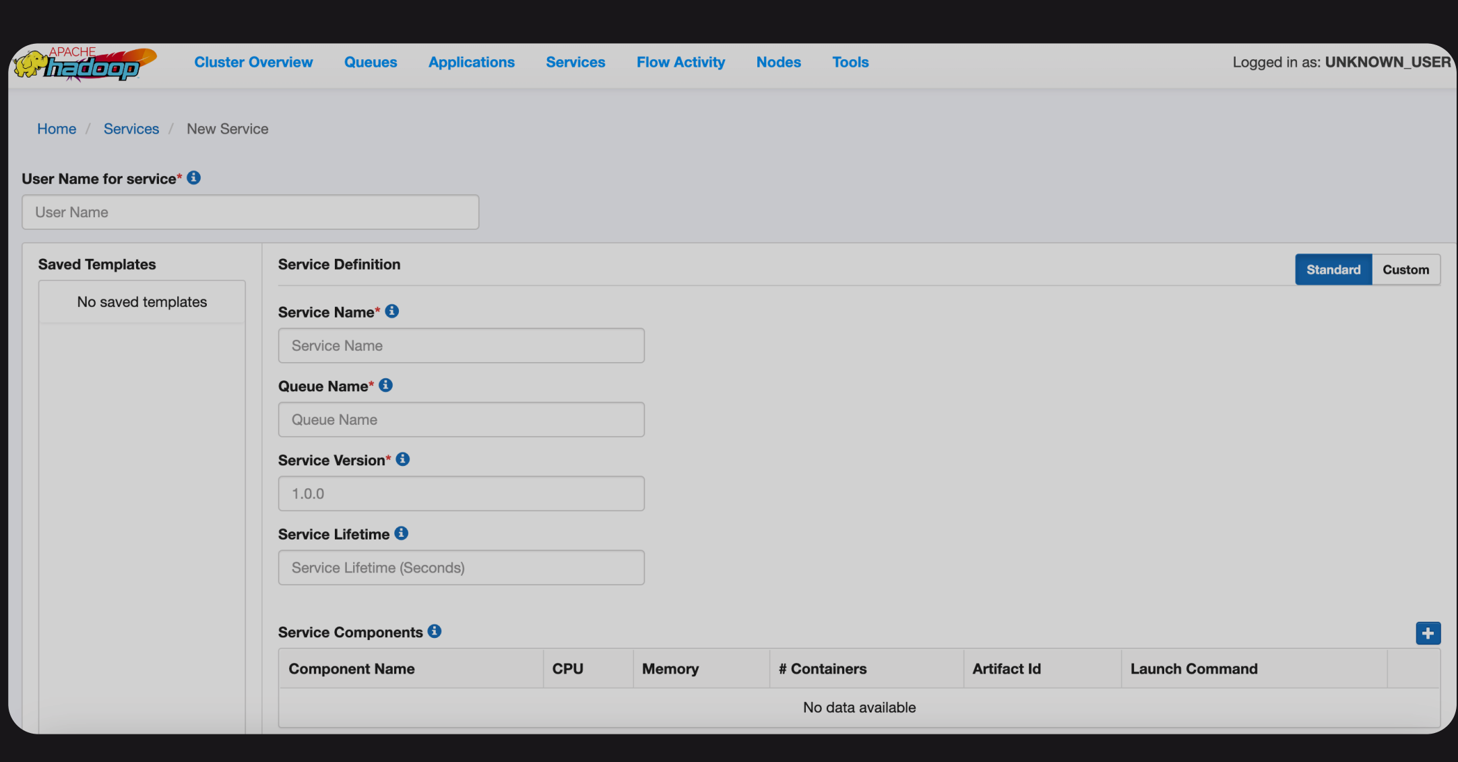
Новый веб-интерфейс ResourceManager (UI2) предоставляет ту же информацию, что и устаревший интерфейс, но в виде вкладок, а не ссылок. Кроме того, некоторая информация из таблиц представлена в виде графиков.


APACHE hadoop
Cluster Overview
Queues
Applications
Services
Flow Activity
Nodes
Tools
Logged in as: UNKNOWN_USER
Home / Cluster Overview
Refresh
Cluster Resource Usage By Applications
Used by others = 0.00%
Available = 100.0000%
Total
100%
Cluster Resource Usage By Leaf Queues
Used by others = 0.00%
Available = 100.0000%
Total
100%
Finished Apps From All Users
Completed = 4
Killed = 0
Failed = 0
Total
4
Running Apps From All Users
Pending = 0
Running = 0
Total
0

ResourceManager UI2: главная страница

Вкладки веб-интерфейса ResourceManager UI2 содержат следующую информацию:

- ***Cluster Overview*** — статистика ресурсов кластера и метрики приложений в виде графиков.
- ***Queues*** — графики очередей планировщика и статистика использования ресурсов кластера.
- ***Applications*** — список приложений с информацией о них и ссылками на страницы каждого приложения.
- ***Services*** — список сервисов, запускающих приложения, и инструмент для их создания.


APACHE hadoop
Cluster Overview
Queues
Applications
Services
Flow Activity
Nodes
Tools
Logged in as: UNKNOWN_USER
Home / Services / New Service
User Name for service*
User Name
Saved Templates
No saved templates
Service Definition
Standard
Custom
Service Name*
Service Name
Queue Name*
Queue Name
Service Version*
1.0.0
Service Lifetime
Service Lifetime (Seconds)
Service Components
+

Component Name	CPU	Memory	# Containers	Artifact Id	Launch Command
No data available					

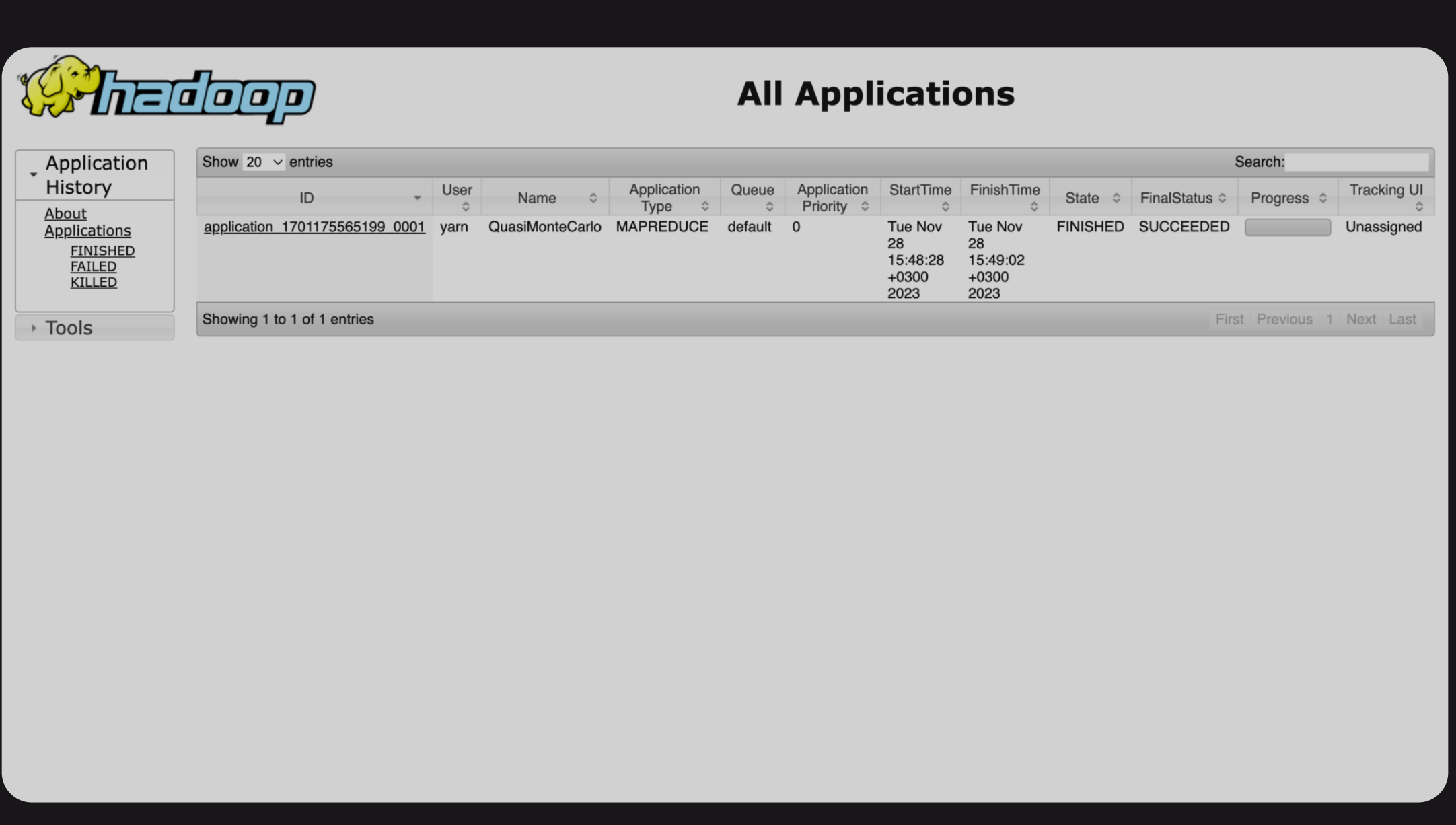
ResourceManager UI2: страница создания сервиса

- ***Flow Activity*** — страница с информацией от Timeline server.
- ***Nodes*** — сведения о хостах кластера, на которых размещены NodeManager.
- ***Tools*** — веб-интерфейс для просмотра логов YARN и файлов конфигурации.

Timeline server

На главной странице пользовательского интерфейса Timeline server слева расположено навигационное меню с двумя разделами:

- ***Application History*** — содержит общую информацию о Timeline server и список всех запущенных приложений, которые можно фильтровать по состоянию: успешно завершено (***FINISHED***), завершено с ошибкой (***FAILED***) и остановлено (**KILLED*).
- ***Tools*** — содержит ссылки на следующую информацию:
  - ***Configuration*** — XML-файл, содержащий файлы конфигурации HDFS и YARN;
  - ***Local logs*** — логи Timeline server, расположенные на хосте;
  - ***Server stacks*** — список активных потоков и их статус;
  - ***Server metrics*** — список метрик JMX.


hadoop
All Applications
Application History
About
Applications
FINISHED
FAILED
KILLED
Tools
Show 20 entries
Search:
First Previous 1 Next Last

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1701175565199_0001	yarn	QuasiMonteCarlo	MAPREDUCE	default	0	Tue Nov 28 15:48:28 +0300 2023	Tue Nov 28 15:49:02 +0300 2023	FINISHED	SUCCEEDED		Unassigned

Showing 1 to 1 of 1 entries

Timeline server UI: главная страница

Подключение к YARN через CLI

Елена Костюченко

Содержание

Назад к содержанию

Подключение

Полезные команды
- Мониторинг нагрузки кластера
- Вывод списка приложений
- Принудительное завершение

Подключение

YARN предоставляет интерфейс командной строки, реализованный в виде скрипта bin/yarn. Чтобы взаимодействовать с YARN через CLI, подключитесь к хосту кластера с YARN через SSH и запустите нужную команду.

Для проверки работы YARN CLI можно запросить версию с помощью команды:

```
$ yarn version
```

Пример вывода:

```
Hadoop 3.1.2
Source code repository git@ssh.gitlab.adsw.io:arenadata/infrastructure/code/ci/prj_adh.git -r
2bfc95569d9993d795ded4878847f3f3db76e77c
Compiled by jenkins on 2023-09-07T07:57Z
Compiled with protoc 2.5.0
From source with checksum 38903f2495a81dfd8e2d8fc4f659a92
This command was run using /usr/lib/hadoop/hadoop-common-3.1.2.jar
```

Запуск команды `yarn` без аргументов выводит список всех команд.

Команды имеют следующий синтаксис:

```
$ yarn [SHELL_OPTIONS] COMMAND [GENERIC_OPTIONS] [SUB_COMMAND] [COMMAND_OPTIONS]
```

Полезные команды

В этом разделе приведены команды, которые могут быть полезны при работе с YARN.

Мониторинг нагрузки кластера

Чтобы увидеть обновляемый в реальном времени список приложений и используемых ими ресурсов, запустите:

```
$ yarn top
```

Пример вывода:

```
YARN top - 11:54:03, up 0d, 2:9, 1 active users, queue(s): root
NodeManager(s): 1 total, 1 active, 0 unhealthy, 0 decommissioned, 0 lost, 0 rebooted
Queue(s) Applications: 1 running, 1 submitted, 0 pending, 0 completed, 0 killed, 0 failed
Queue(s) Mem(GB): 0 available, 6 allocated, 9 pending, 0 reserved
Queue(s) VCores: 0 available, 5 allocated, 9 pending, 0 reserved
Queue(s) Containers: 5 allocated, 9 pending, 0 reserved

                  APPLICATIONID USER             TYPE      QUEUE PRIOR   #CONT  #RCONT  VCORES RVCORES     MEM    RMEM  VCORESECS
MEMSECS %PROGR       T
 application_1705311887839_0001 yarn        mapreduce    default     0       5       0       5       0      6G      0G         57
75  16.25   00:00
```

Эта команда выводит обновляющийся список приложений. Чтобы увидеть статический список запущенных приложений, используйте команду `application -list`.

Вывод списка приложений

Чтобы увидеть список приложений, запущенных на хосте, выполните:

```
$ yarn application -list
```

Вы можете фильтровать список приложений, используя опции:

- `-appTypes` — фильтровать по типам приложений;
- `-appStates` — фильтровать по статусу приложений;
- `-appTags` — фильтровать по тегам приложений.

Пример вывода:

```
2024-01-15 11:56:01,792 INFO client.RMProxy: Connecting to ResourceManager at elenas-adh2.ru-central1.internal/127.0.0.1:8032
2024-01-15 11:56:02,241 INFO client.AHSProxy: Connecting to Application History server at elenas-adh2.ru-central1.internal/127.0.
0.1:10200
Total number of applications (application-types: [], states: [SUBMITTED, ACCEPTED, RUNNING] and tags: []):1
                Application-Id      Application-Name        Application-Type          User           Queue
State             Final-State         Progress                        Tracking-URL
application_1705311887839_0002       QuasiMonteCarlo                MAPREDUCE          yarn         default
RUNNING               UNDEFINED               5%  http://elenas-adh3.ru-central1.internal:33585
```

Принудительное завершение

Чтобы завершить работу приложения, которое перестало отвечать, выполните:

```
$ yarn application -kill <Application ID>
```

Где `<Application ID>` — это идентификатор приложения, которое нужно остановить. В команде можно передать несколько ID, разделенных пробелами.

Более подробную информацию о командах YARN можно получить в статье YARN CLI.

Подключение к YARN через REST API

Елена Костюченко

Содержание

Назад к содержанию

Обзор
ResourceManager API
NodeManager
Timeline Server

Обзор

Hadoop предоставляет набор REST API для всех компонентов YARN. Они обеспечивают URI-доступ к кластеру, нодам, приложениям и истории приложений.

Большинство API-интерфейсов YARN поддерживают только запросы `GET`, но некоторые компоненты YARN, такие как ResourceManager, поддерживают `POST`. В заголовке запроса используются только поля `Accept` и `Accept-Encoding`, все остальные поля заголовка игнорируются.

Чтобы указать, какой формат нужен в ответе, необходимо передать в поле `Accept` значение `XML` или `JSON`. Формат по умолчанию — JSON.

Чтобы получить ответ в формате gzip-архива, передайте `gzip` в поле `Accept-Encoding`.

Синтаксис URI для YARN REST API выглядит так:

http://<IP:PORT>/ws/<version>/<resourcepath>

Здесь:

- `<IP:PORT>` — IP или FQDN хоста компонента и его порт;
- `<version>` — версия API;
- `<resourcepath>` — путь, который определяет, какую функцию API вызывать.

Аутентификация

Использование API требует аутентификации. Если в кластере не настроена безопасность, добавьте в запросы параметр `?user.name=<user>`. Где `<user>` — это имя пользователя, под которым необходимо выполнить запрос. Например, `admin`.

Если безопасность настроена, метод аутентификации зависит от модели безопасности.

Если включен Kerberos SPNEGO, используйте параметры `--negotiate` и `-u`:

```
$ curl -i --negotiate -u : "http://<HOST>:<HTTP_PORT>/ws/<version>/<resourcepath>"
```

Если включена аутентификация с помощью токена делегирования Hadoop, передайте токен вместо имени пользователя:

```
$ curl -i "http://<HOST>:<HTTP_PORT>/ws/<version>/<resourcepath>?delegation=<TOKEN>"
```

Более подробно о методах аутентификации читайте в статье Authentication for Hadoop HTTP web-consoles.

Пример запроса к Timeline server:

```
$ curl http://127.0.0.1:8188/ws/v1/timeline?user.name=admin
```

В ответе содержится информация о сервере:

```
{"About":"Timeline API","timeline-service-version":"3.1.2","timeline-service-build-version":"3.1.2 from
2bfc95569d9993d795ded4878847f3f3db76e77c by jenkins source checksum 7954337bcd9688eca8aa32720d2c74","timeline-service-version-
built-on":"2023-09-07T08:01Z","hadoop-version":"3.1.2","hadoop-build-version":"3.1.2 from 2bfc95569d9993d795ded4878847f3f3db76e77c
by jenkins source checksum 38903f2495a81dfd8e2d8fc4f659a92","hadoop-version-built-on":"2023-09-07T07:57Z"}
```

ПРИМЕЧАНИЕ

Поля в теле ответа не ограничены по количеству и не имеют определенного порядка. При разработке приложений на базе YARN REST API используйте парсеры, которые не зависят от конкретного порядка.

ResourceManager API

ResourceManager REST API обеспечивает доступ к информации о кластере и позволяет управлять приложениями. Полная спецификация API доступна в статье ResourceManager REST API's.

Пример использования REST API ResourceManager для добавления нового приложения:

1. Запросите ID для приложения с помощью команды:

```
$ curl -X POST http://127.0.0.1:8088/ws/v1/cluster/apps/new-application?user.name=admin
```

Ответ сервера:

```
{"application-id":"application_1705573015955_0001","maximum-resource-capability":
{"memory":7057,"vCores":2,"resourceInformations":{"resourceInformation":
[{"maximumAllocation":9223372036854775807,"minimumAllocation":0,"name":"memory-
mb","resourceType":"COUNTABLE","units":"Mi","value":7057},
{"maximumAllocation":9223372036854775807,"minimumAllocation":0,"name":"vcores","resourceType":"COUNTABLE","units":"","value":2}]}
}}[admin@elenas-adh2 ~]$
```

2. В ответе сервера содержится ID приложения и информация о доступных ресурсах. Создайте JSON-файл с параметрами приложения и передайте его в ResourceManager с помощью команды:

```
$ curl -v -X POST -d @example-app.json -H "Content-type: application/json"  http://127.0.0.1:8088/ws/v1/cluster/apps/?user.
name=admin
```

Пример информации о приложении

```
 {
    "application-id":"application_1705573015955_0001",
    "application-name":"test",
    "am-container-spec":
    {
       "local-resources":
       {
          "entry":
         [
             {
                "key":"hadoop-mapreduce-examples.jar",
                "value":
                {
                   "resource":"hdfs://127.0.0.2:8020/tmp/hadoop-mapreduce-examples.jar",
                   "type":"FILE",
                   "visibility":"APPLICATION",
                   "size": "30897",
                   "timestamp": "1405452071209"
                }
             }
          ]
       },
       "commands":
       {
          "command":"{{JAVA_HOME}}/bin/java -Xmx10m org.apache.hadoop.yarn.applications.distributedshell.ApplicationMaster --
container_memory 10 --container_vcores 1 --num_containers 1 --priority 0 1><LOG_DIR>/AppMaster.stdout 2><LOG_DIR>/AppMaster.
stderr"
       },
       "environment":
       {
          "entry":
          [
             {
                "key": "DISTRIBUTEDSHELLSCRIPTTIMESTAMP",
                "value": "1405459400754"
             },
             {
                "key": "CLASSPATH",
                "value": "{{CLASSPATH}}<CPS>./*<CPS>{{HADOOP_CONF_DIR}}<CPS>{{HADOOP_COMMON_HOME}}/share/hadoop/common/*<CPS>
{{HADOOP_COMMON_HOME}}/share/hadoop/common/lib/*<CPS>{{HADOOP_HDFS_HOME}}/share/hadoop/hdfs/*<CPS>
{{HADOOP_HDFS_HOME}}/share/hadoop/hdfs/lib/*<CPS>{{HADOOP_YARN_HOME}}/share/hadoop/yarn/*<CPS>
{{HADOOP_YARN_HOME}}/share/hadoop/yarn/lib/*<CPS>./log4j.properties"
             },
             {
                "key": "DISTRIBUTEDSHELLSCRIPTLEN",
                "value": "6"
             },
             {
                "key": "DISTRIBUTEDSHELLSCRIPTLOCATION",
                "value": "hdfs://127.0.0.2:8020/tmp/hadoop-mapreduce-examples.jar"
             }
          ]
       }
    },
    "unmanaged-AM":"false",
    "max-app-attempts":"2",
    "resource":
    {
       "memory":"1024",
       "vCores":"1"
    },
    "application-type":"YARN",
    "keep-containers-across-application-attempts":"false"
  }
```

Если запрос был отправлен успешно, сервер возвращает код `HTTP/1.1 202 Accepted` и URL-адрес приложения, по которому можно отслеживать его статус.

Чтобы включить поддержку CORS только для ResourceManager, установите следующие параметры конфигурации:

- *core-site.xml* — установите значение параметра `hadoop.http.filter.initializers` равным `org.apache.hadoop.security.HttpCrossOriginFilterInitializer`;
- *yarn-site.xml* — установите значение параметра `yarn.resourcemanager.webapp.cross-origin.enabled` равным `true`.

NodeManager

NodeManager REST API предоставляет информацию о нодах, приложениях и контейнерах, работающих на этой ноде. Полная спецификация API доступна в статье NodeManager REST API's.

Пример команды, которая выводит информацию о ноде:

```
$ curl -X GET http://<IP>:8042/ws/v1/node/info?user.name=admin
```

Здесь `<IP>` — это IP-адрес или FQDN хоста.

Чтобы включить поддержку CORS для NodeManager, установите следующие параметры конфигурации:

- *core-site.xml* — установите значение параметра `hadoop.http.filter.initializers` равным `org.apache.hadoop.security.HttpCrossOriginFilterInitializer`;
- *yarn-site.xml* — установите значение параметра `yarn.nodemanager.webapp.cross-origin.enabled` равным `true`.

Timeline Server

Timeline Server REST API поддерживает два метода: `GET` и `POST`. На данный момент доступны две версии API:

- Timeline Server REST API v1
- Timeline Server REST API v2

Пример команды, которая выводит список доменов указанного владельца:

```
$ curl -X GET http://<IP>:8188/ws/v1/timeline/domain?owner=<owner>
```

Здесь `<IP>` — это IP-адрес или FQDN хоста, а `<owner>` — это имя владельца домена.

FairScheduler

Альберт Багдасарян

Содержание

Назад к содержанию

- Пример распределения ресурсов
- Поддержка иерархических очередей
- Конфигурация
  - Настройка очередей
  - Пример конфигурации очереди

FairScheduler — это подключаемый планировщик для Hadoop, который позволяет приложениям YARN совместно использовать ресурсы в больших кластерах таким образом, что все приложения получают в среднем равную долю ресурсов на некотором интервале времени.

Планировщик FairScheduler распределяет ресурсы между всеми запущенными приложениями так, что приложения, работающие недолго, завершают работу в разумное время, а приложения, работающие долго, не испытывают недостатка ресурсов. Он организует приложения в очереди и "справедливо" распределяет ресурсы между этими очередями.

По умолчанию FairScheduler принимает решения о справедливом планировании, используя в качестве ресурса только оперативную память и единственную очередь под названием `default`. Вы можете настроить его на учет одновременно памяти (`mb` — мегабайты) и процессоров (`vcores` — количество виртуальных процессоров). Приложения, поступающие в обработку, могут потребовать создания новых очередей, как это описано в секции Настройка очередей.

Пример распределения ресурсов

Допустим, есть две очереди:

- `sales` — очередь продаж.
- `finance` — очередь финансовых операций.

В очередь `sales` подается задание. Будучи единственным запущенным заданием, оно получит все ресурсы кластера.

Затем другое задание передается в очередь `finance`. В результате новое задание постепенно получает половину ресурсов кластера.

Таким образом, задания в обеих очередях будут иметь по 50% ресурсов. Теперь в очередь `finance` подается еще одно задание. Это приведет к тому, что половина ресурсов, выделенных для очереди `finance`, будет выделена для этого нового задания.

Два задания в очереди `finance` теперь будут разделять между собой ресурсы, выделенные для этой очереди (50% от общего объема ресурсов), в равных пропорциях. Единственное задание в очереди `sales` будет использовать все 50% ресурсов кластера, выделенных для этой очереди.

Поддержка иерархических очередей

FairScheduler поддерживает иерархические очереди. Очередь является иерархической, если внутри этой очереди можно создавать дочерние очереди.

Существует корневая очередь root, которая включает все остальные очереди. Однако префикс "root" можно опускать в полном названии очереди при настройках. Планировщик распределяет доступные ресурсы между дочерними очередями корневой очереди (верхний уровень) с использованием концепции "справедливого" планирования. Затем он распределяет таким же образом ресурсы очередей верхнего уровня между их дочерними очередями следующего уровня и так далее вниз по дереву иерархии.

Конфигурация

Чтобы использовать FairScheduler в YARN, необходимо начать с назначения соответствующего класса планировщика в файле *yarn-site.xml*:

```
<property>
  <name>yarn.resourcemanager.scheduler.class</name>
  <value>org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler</value>
</property>
```

Настройка очередей

Для настройки очередей используйте следующие XML-элементы в конфигурационном файле *fair-scheduler.xml*:

- `<queue>`. Представляет очередь. Подобные элементы встраивают друг в друга для образования дерева очередей. Некоторые из наиболее важных свойств этого элемента перечислены ниже:
  - `minResources`. Указывает минимально необходимое количество ресурсов для очереди (X mb, Y vcores). Если это требование не соблюдено, планировщик будет предоставлять освобождающиеся ресурсы для этой очереди прежде, чем для других очередей того же родителя.
  - `maxResources`. Указывает максимальный объем ресурсов, который можно выделять для очереди, в абсолютных значениях (X mb, Y vcores) или в процентах от ресурсов кластера (X% памяти, Y% CPU).
  - `weight`. Влияет на пропорцию распределения ресурсов кластера между очередями. Значение веса очереди по умолчанию равно `1`. Если вес очереди равен `2`, то эта очередь должна получать примерно в два раза больше ресурсов, чем очередь с весом по умолчанию.
  - `schedulingPolicy`. Позволяет установить политику планирования для любой очереди. Можно назначать один из объектов, расширяющих класс `org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.SchedulingPolicy`, то есть: `fifo`, `fair`, `drf` или другой. Значение по умолчанию — `fair`.
- `<defaultQueueSchedulingPolicy>`. Устанавливает политику планирования по умолчанию для очередей. Значение по умолчанию — `fair`. Этот элемент переопределяется параметром `schedulingPolicy` в настройках очереди.
- `<queueMaxAppsDefault>`. Устанавливает предельно допустимое количество запущенных приложений по умолчанию для очередей. Этот элемент переопределяется параметром `maxRunningApps` в настройках очереди.
- `<queuePlacementPolicy>`. Содержит список правил (представленных XML-элементами), которые указывают планировщику, как размещать входящие приложения в очередях. Правила применяются в том порядке, в котором они перечислены. Планировщик выполняет первое подходящее правило. Каждое правило имеет атрибут `create`, который указывает, может ли планировщик создать новую очередь. По умолчанию `create` имеет значение `true`. Если значение установлено в `false` и правило помещает приложение в очередь, которая не настроена в файле распределения, планировщик переходит к следующему правилу. Возможные правила перечислены ниже:
  - `specified`. Приложение помещается в запрашиваемую очередь.
  - `user`. Приложение помещается в очередь с именем пользователя, который добавил данное приложение.
  - `primaryGroup`. Приложение помещается в очередь с именем основной группы пользователя, который добавил это приложение.
  - `secondaryGroupExistingQueue`. Приложение помещается в очередь с именем, которое соответствует дополнительной группе пользователя, добавившего это приложение.
  - `nestedUserQueue`. Приложение помещается в очередь с именем пользователя под очередью, предложенной вложенным правилом. Правило этого типа должно иметь вложенное правило, определяющее родительскую очередь.

    Начиная с версии ADH 3.3.6.1 в блоке `nestedUserQueue` следует указывать только одно вложенное правило. Если блок содержит несколько вложенных правил, как показано в примере ниже, будет использовано последнее указанное правило (`primaryGroup`):

    ```
    <rule name="nestedUserQueue" create="true">
        <rule name="secondaryGroupExistingQueue" create="false"/>
        <rule name="primaryGroup" create="false"/>
    </rule>
    ```
  - `default`. Приложение помещается в очередь, указанную в атрибуте `queue` правила по умолчанию. Если атрибут `queue` не указан, приложение помещается в очередь `root.default`.
  - `reject`. Приложение отклоняется.

> **ПРИМЕЧАНИЕ**
>
> Вы можете изменять параметры планировщика (например, минимальные доли, лимиты, вес, preemtion тайм-ауты, политики планирования и прочее) на лету посредством редактирования файла распределения. В таком случае не нужно выполнять команду ***Reload Scheduler config*** — планировщик автоматически перезагрзит файл через 10-15 секунд после изменения файла.

Пример конфигурации очереди

Пусть есть две очереди верхнего уровня (нисходящие от корневой):

- `sales` — очередь продаж.
- `finance` — очередь по финансовым операциям.

В очереди `sales` есть две дочерние очереди:

- `apac`
- `emea`

Очереди могут быть настроены следующим образом:

```
<allocations>
  <queue name="sales">
      <minResources>10000 mb,0vcores</minResources>
      <maxResources>50000 mb,0vcores</maxResources>
      <weight>2.0</weight>
      <schedulingPolicy>fifo</schedulingPolicy>
      <queue name="emea" />
      <queue name="apac" />
  </queue>
  <queue name="finance">
      <minResources>10000 mb,0vcores</minResources>
      <maxResources>70000 mb,0vcores</maxResources>
      <weight>3.0</weight>
      <schedulingPolicy>fair</schedulingPolicy>
  </queue>
  <queuePlacementPolicy>
      <rule name="specified"   />
      <rule name="primaryGroup" create="false" />
      <rule name="default" queue="finance" />
  </queuePlacementPolicy>
</allocations>
```

CapacityScheduler

Альберт Багдасарян

Содержание

Назад к содержанию

- Система безопасности
- Поддержка иерархических очередей
- Конфигурация
- Настройка очередей
- Пример конфигурации очереди

CapacityScheduler — это подключаемый планировщик в Hadoop. Он в наибольшей степени подходит для планирование заданий в кластере, арендуемом множеством организаций (multi-tenancy), путем выделения части ресурсов кластера для каждой организации.

Система безопасности

Планировщик позволяет каждой организации создавать свою собственную очередь, выделяя ей ресурсы кластера, доступные для данной организации.

В кластере с общим доступом безопасность становится очень важной. Для каждой очереди существует список контроля доступа (Access Control List, ACL), который определяет, какие пользователи могут подавать заявки в отдельные очереди. Система безопасности гарантирует, что пользователи не смогут просматривать и изменять приложения других пользователей в других очередях. Кроме того, эта система позволяет создавать роль администратора для каждой очереди и роль системного администратора.

Поддержка иерархических очередей

CapacityScheduler поддерживает иерархические очереди. Очередь является иерархической, если можно создавать дочерние очереди внутри нее. Часть ресурсов кластера, выделенная для очереди, может быть далее распределена между дочерними очередями. CapacityScheduler имеет предопределенную очередь, называемую **корневой (root)**. Все очереди в системе являются дочерними по отношению к корневой очереди.

Дополнительным преимуществом является то, что организация может использовать больше ресурсов кластера, чем было выделено для ее очереди, но только если доступны дополнительные мощности и они не используются другими. Это обеспечивает организациям большую гибкость с точки зрения затрат.

Конфигурация

Чтобы использовать CapacityScheduler в YARN, необходимо начать с назначения соответствующего класса планировщика в файле *yarn-site.xml*:

```
<property>
  <name>yarn.resourcemanager.scheduler.class</name>
  <value>org.apache.hadoop.yarn.server.resourcemanager.scheduler.capacity.CapacityScheduler</value>
</property>
```

Настройка очередей

Чтобы настроить очереди, отредактируйте конфигурационный файл *capacity-scheduler.xml*:

- Настройте очереди верхнего уровня с помощью параметра `yarn.scheduler.capacity.root.queues`. Необходимо ввести список очередей, разделенных запятыми.
- Настройте дочерние очереди в соответствии со структурой дерева очередей. Для каждой очереди вида <queue-path>, которая должна иметь дочерние очереди, введите эти очереди в параметре `yarn.scheduler.capacity.<queue-path>.queues`.
- Укажите объем ресурсов для каждой очереди в процентах с помощью параметра `yarn.scheduler.capacity.<queue-path>.capacity`. Сумма этих величин для всех очередей (на каждом уровне) должна быть равна 100.
- Укажите максимально допустимый объем ресурсов очереди в процентах с помощью параметра `yarn.scheduler.capacity.<queue-path>.maximum-capacity`. Это ограничивает гибкость потребления ресурсов для приложений в очереди. По умолчанию параметр равен `-1`, что отключает ограничение.

Пример конфигурации очереди

Пусть есть две очереди верхнего уровня (нисходящие от корневой):

- `sales` — очередь продаж.
- `finance` — очередь для финансовых операций.

В очереди продаж есть две дочерние очереди:

- `apac`
- `emea`

Очереди следует настроить следующим образом:

```
<property>
  <name>yarn.scheduler.capacity.root.queues</name>
  <value>sales, finance</value>
</property>
<property>
  <name>yarn.scheduler.capacity.root.sales.queues</name>
  <value>apac,emea</value>
</property>
```

Чтобы выделить 70% ресурсов для очереди `sales` и 30% для `finance`, используйте следующую настройку:

```
<property>
  <name>yarn.scheduler.capacity.root.sales.capacity</name>
  <value>70</value>
</property>
<property>
  <name>yarn.scheduler.capacity.root.finance.capacity</name>
  <value>30</value>
</property>
```

Для того чтобы выделить 65% ресурсов очереди `sales` на `apac` и 35% на `emea`, используйте следующую настройку:

```
<property>
  <name>yarn.scheduler.capacity.root.sales.apac.capacity</name>
  <value>65</value>
</property>
<property>
  <name>yarn.scheduler.capacity.root.sales.emea.capacity</name>
  <value>35</value>
</property>
```

Для того чтобы очередь продаж не использовала более 80% ресурсов кластера (даже если ресурсы доступны), примените следующую настройку:

```
<property>
  <name>yarn.scheduler.capacity.root.sales.maximum-capacity</name>
  <value>80</value>
</property>
```

Если пользователь, например `yarn`, поставляет приложения без указания очереди и при этом отсутствует очередь `default`, то необходимо настроить очередь по умолчанию. В следующем примере очередь `marketing` объявлена как очередь по умолчанию для пользователя `yarn`:

```
<property>
   <name>yarn.scheduler.capacity.queue-mappings</name>
   <value>u:yarn:marketing</value>
</property>
```

где:

- `u` означает, что это настройка для пользователя (`g` — для группы);
- `yarn` — имя пользователя;
- `marketing` — название очереди по умолчанию.

Обзор YARN high availability

Константин Алпашкин, Альберт Багдасарян

Содержание

Назад к содержанию

Архитектура

Режим автоматической отказоустойчивости

Выбор лидера

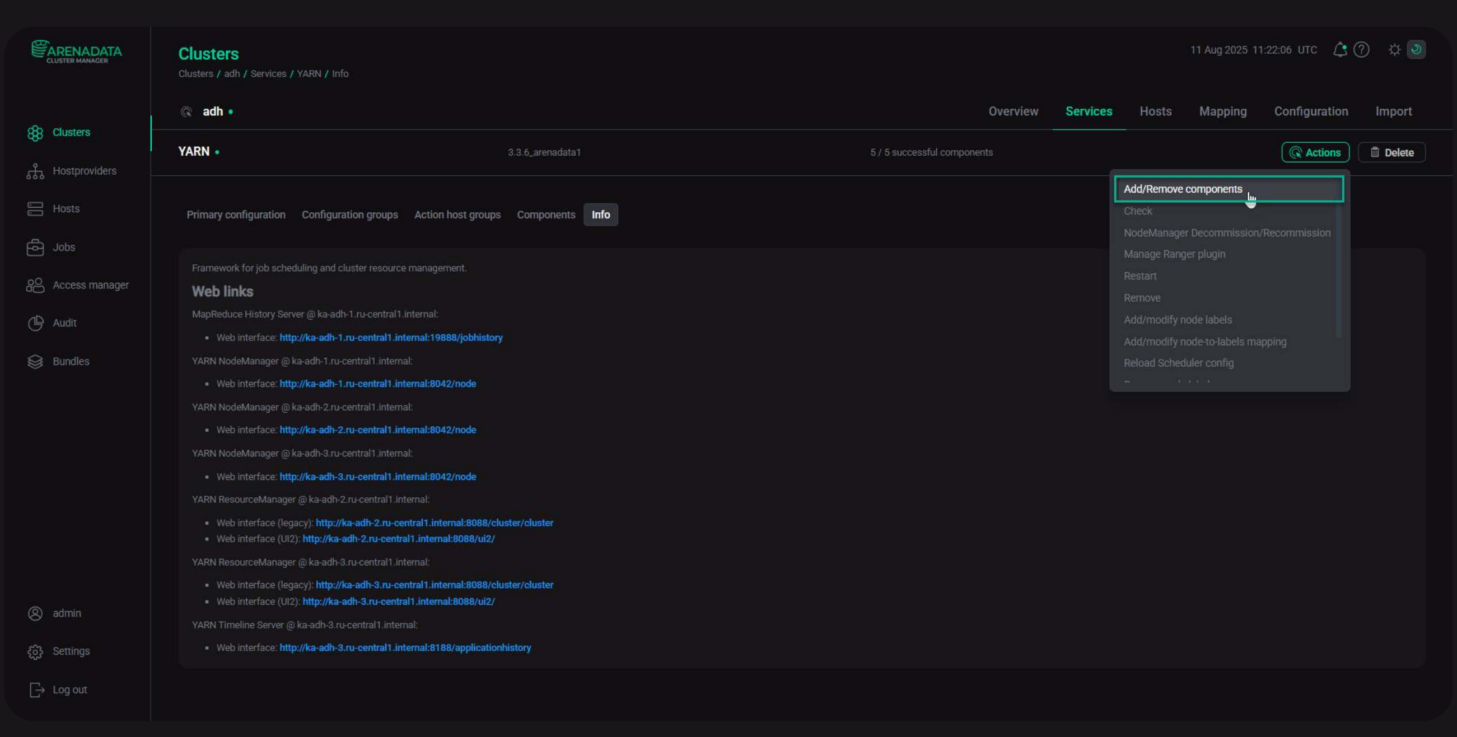
- Алгоритм выбора

Действия клиентов, ApplicationMaster и NodeManager при переключении RM

ResourceManager (RM) является ключевым компонентом YARN, отвечающим за распределение ресурсов (CPU/RAM) в кластере Hyperwave, а также за планирование запуска приложений, таких как MapReduce-задачи.

Два или более RM могут работать в режиме высокой доступности (High Availability, HA), обеспечивая непрерывную работу даже в случае отказа узла кластера с активным RM. Если активный RM вдруг становится недоступен, один из резервных RM автоматически занимает его место с помощью специального алгоритма избрания.

Начиная с версии ADH 2.1.10.b1 активация HA-режима выполняется автоматически и не требует никаких специальных действий. Режим HA включается автоматически при добавлении двух или более компонентов YARN ResourceManager через интерфейс ADCM, как показано на рисунке ниже.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / adh / Services / YARN / Info
11 Aug 2025 11:22:06 UTC
adh
Overview
Services
Hosts
Mapping
Configuration
Import
YARN
3.3.6_arenadata1
5 / 5 successful components
Actions
Delete
Add/Remove components
Check
NodeManager Decommission/Recommission
Manage Ranger plugin
Restart
Remove
Add/modify node labels
Add/modify node-to-labels mapping
Reload Scheduler config
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Primary configuration
Configuration groups
Action host groups
Components
Info

Framework for job scheduling and cluster resource management.

Web links

MapReduce History Server @ ka-adh-1.ru-central1.internal:
- Web interface: http://ka-adh-1.ru-central1.internal:19888/jobhistory

YARN NodeManager @ ka-adh-1.ru-central1.internal:
- Web interface: http://ka-adh-1.ru-central1.internal:8042/node

YARN NodeManager @ ka-adh-2.ru-central1.internal:
- Web interface: http://ka-adh-2.ru-central1.internal:8042/node

YARN NodeManager @ ka-adh-3.ru-central1.internal:
- Web interface: http://ka-adh-3.ru-central1.internal:8042/node

YARN ResourceManager @ ka-adh-2.ru-central1.internal:
- Web interface (legacy): http://ka-adh-2.ru-central1.internal:8088/cluster/cluster
- Web interface (UI2): http://ka-adh-2.ru-central1.internal:8088/ui2/

YARN ResourceManager @ ka-adh-3.ru-central1.internal:
- Web interface (legacy): http://ka-adh-3.ru-central1.internal:8088/cluster/cluster
- Web interface (UI2): http://ka-adh-3.ru-central1.internal:8088/ui2/

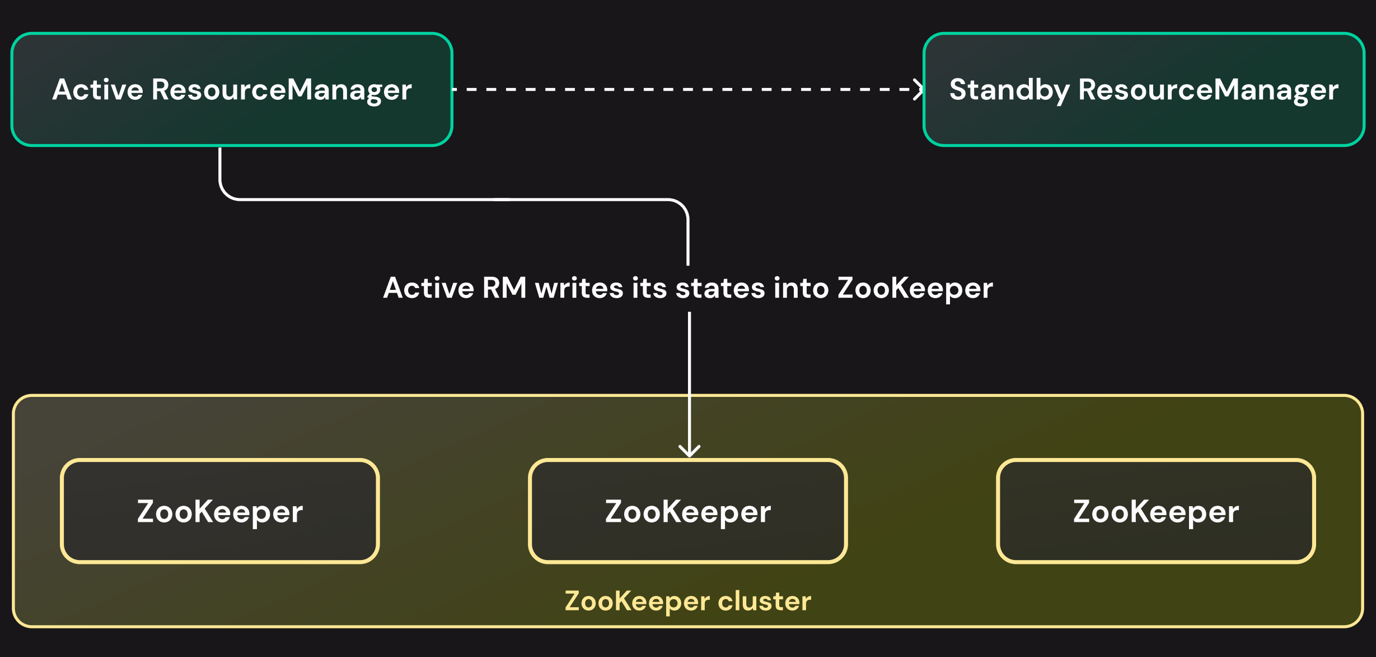
YARN Timeline Server @ ka-adh-3.ru-central1.internal:
- Web interface: http://ka-adh-3.ru-central1.internal:8188/applicationhistory

Добавление ResourceManager

Для обеспечения HA необходимы сервисы ZooKeeper и HDFS.

Архитектура

В основе режима высокой доступности для RM лежит подход "активный-пассивный" (active-standby), то есть в любой момент времени один из RM является активным, а один или несколько других RM являются резервными. Последние находятся в режиме ожидания (standby) и готовы занять место активного RM при необходимости.


Active ResourceManager
Standby ResourceManager
Active RM writes its states into ZooKeeper
ZooKeeper
ZooKeeper
ZooKeeper
ZooKeeper cluster

Компоненты отказоустойчивой системы

Режим автоматической отказоустойчивости

По умолчанию система автоматически переключается на резервный RM в случае сбоя активного RM. При переключении система должна убедиться, что не возникнет сценария **split-brain**.

Сценарий split-brain — это ситуация, когда несколько RM берут на себя роль активного RM. Такая ситуация нежелательна, поскольку несколько RM могут начать одновременно управлять узлами кластера, что приводит к несогласованности в управлении ресурсами и может стать причиной отказа кластера в целом.

Таким образом, в любой момент времени необходимо быть уверенным в том, что в кластере задействован один и только один активный RM. Для этих целей используется ZooKeeper, позволяющий координировать процесс переключения с помощью метода **избрания лидера** (**Leader Election**), который описан далее.

Выбор лидера

Для лучшего понимания процесса выбора активного RM стоит упомянуть ключевые понятия ZooKeeper.

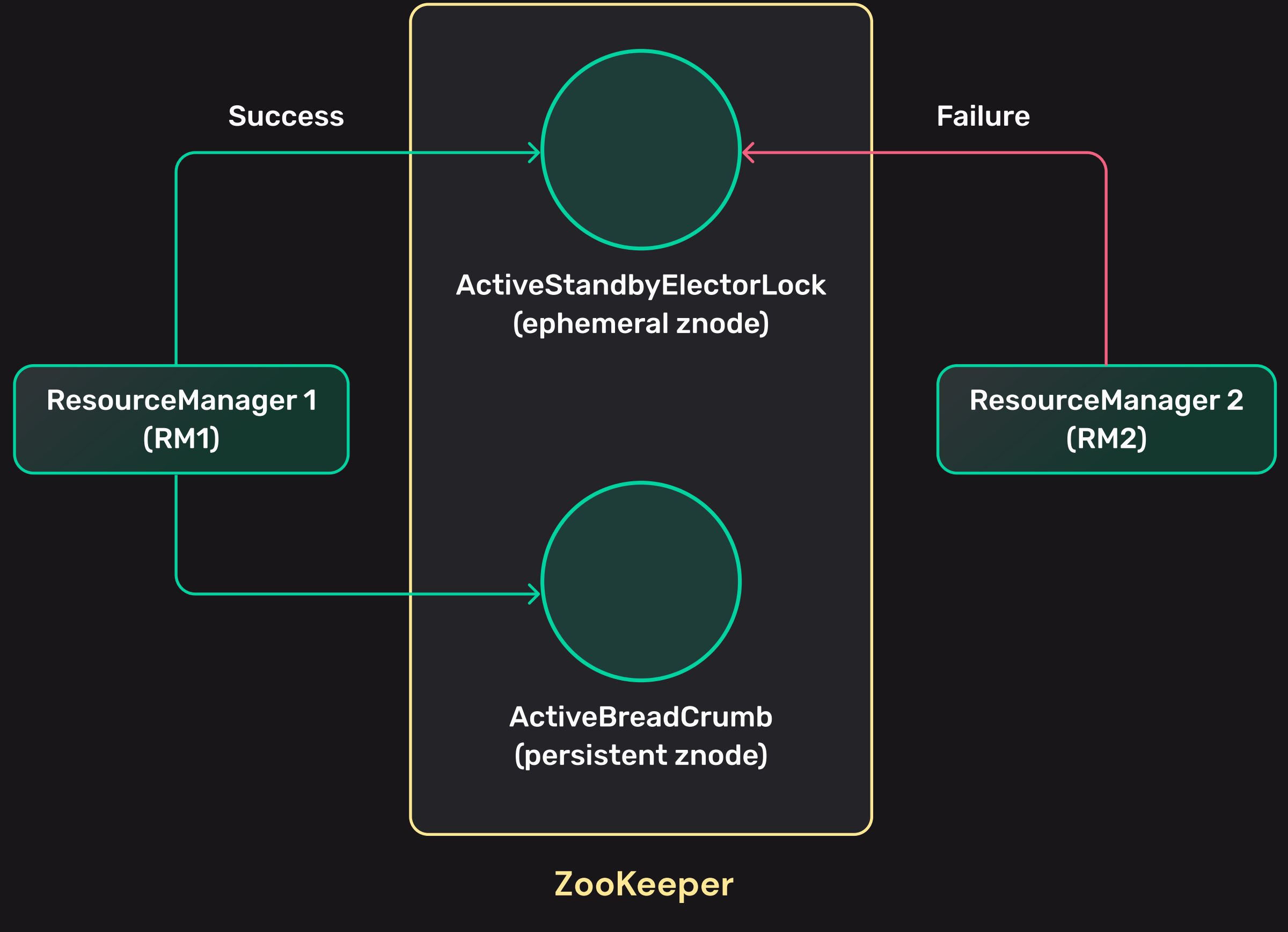
Клиент ZooKeeper (пользователь, сервис или приложение) может создавать Znodes двух типов:

- **Эфемерный (ephemeral)**. Эфемерный Znode удаляется, если сессия, в которой был создан Znode, отключилась. Например, если RM1 создал эфемерный Znode с именем `eph_znode`, этот Znode будет удален, если сессия взаимодействия RM1 с ZooKeeper по какой-либо причине завершится.
- **Постоянный (persistent)**. Постоянный Znode существует в ZooKeeper до тех пор, пока не будет удален. В отличие от эфемерных Znode, постоянный Znode не зависит от существования процесса, который его создал. Например, если RM1 создает постоянный Znode, то этот Znode продолжает существовать и после завершения работы RM1. Такой Znode можно удалить только явным образом.

Более подробная информация об архитектуре и модели данных ZooKeeper доступна в разделе Архитектура ZooKeeper.

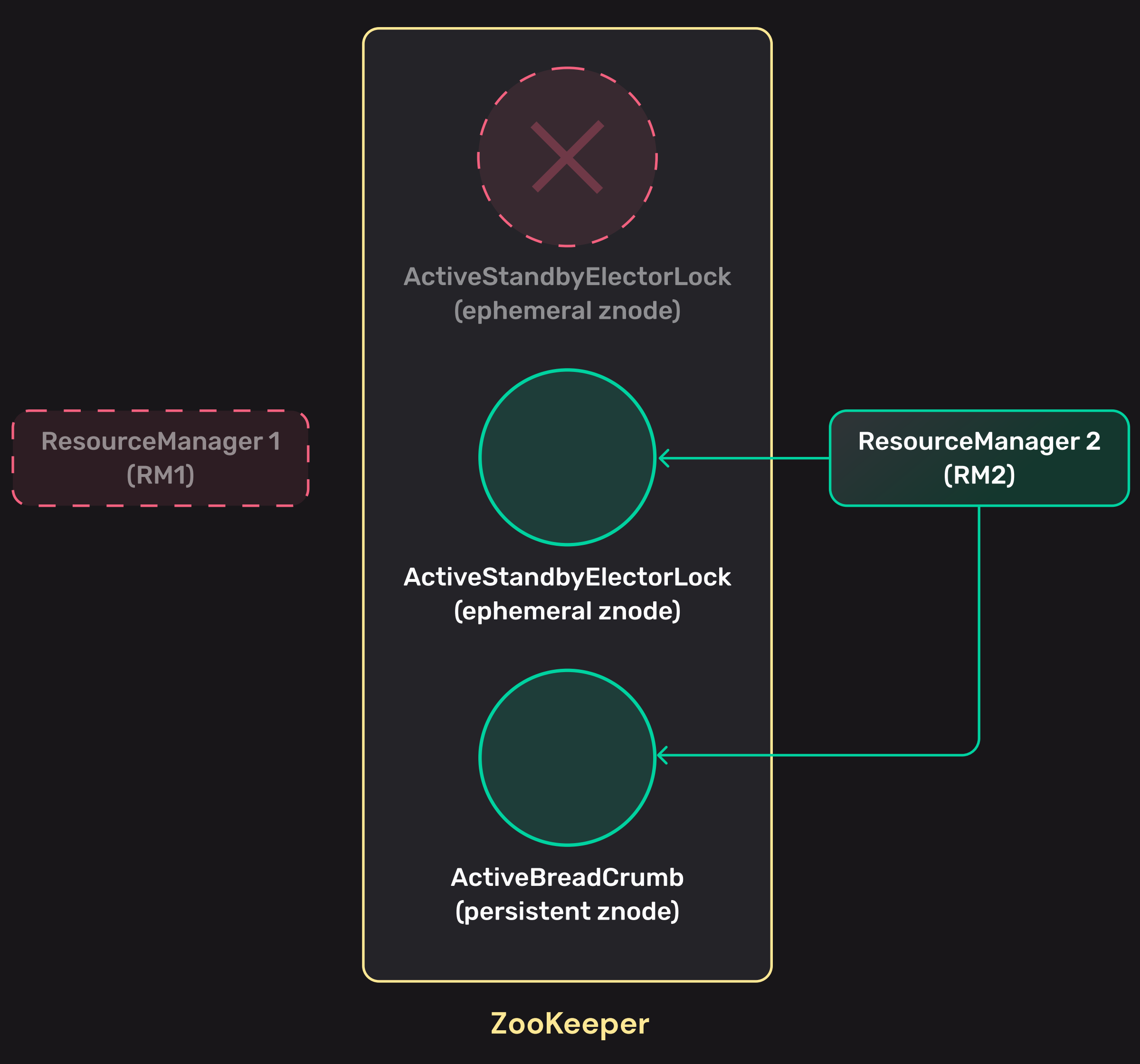
Алгоритм выбора

Допустим, имеется кластер с двумя RM (RM1 и RM2), которые претендуют на роль активного RM. Выбор RM выполняется следующим образом: RM, который первым смог создать эфемерный Znode с именем `ActiveStandbyElectorLock`, получает активную роль. Важно, что только один RM может создать Znode с именем `ActiveStandby`.


Success
Failure
ActiveStandbyElectorLock
(ephemeral znode)
ResourceManager 1
(RM1)
ResourceManager 2
(RM2)
ActiveBreadCrumb
(persistent znode)
ZooKeeper

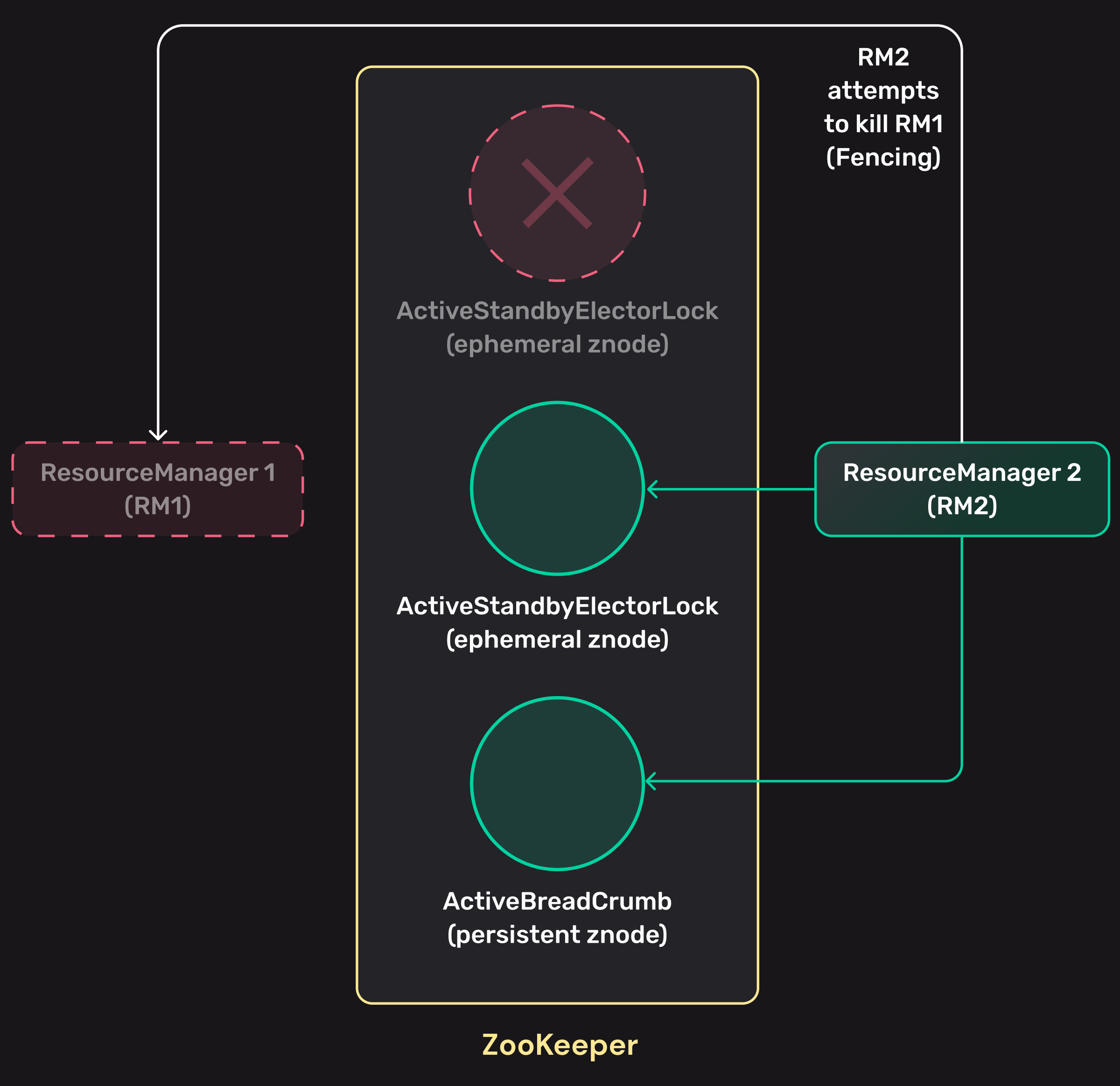
Состояние системы до отказа

Если RM1 выйдет из строя, Znode `ActiveStandbyElectorLock` будет автоматически удален, поскольку он является эфемерным. В таком случае резервный RM2 станет активным.


ActiveStandbyElectorLock
(ephemeral znode)
ResourceManager 1
(RM1)
ResourceManager 2
(RM2)
ActiveStandbyElectorLock
(ephemeral znode)
ActiveBreadCrumb
(persistent znode)
ZooKeeper

Состояние системы после восстановления

Иногда RM может отключаться на короткое время, а затем снова подключаться к ZooKeeper. Однако ZooKeeper не может различать такие краткосрочные отключения, поэтому он удаляет эфемерные Znodes, созданные этим RM. Это может привести к сценарию split-brain, когда оба RM действуют как активные. Чтобы избежать этого, RM2 считывает информацию об активном RM из persistent Znode `ActiveBreadCrumb`, в который RM1 ранее записал информацию о себе (имя хоста и TCP-порт). Как только RM2 понимает, что RM1 активный, он пытается изолировать RM1, создавая "ограждение" (fencing). После этого RM2 перезаписывает данные в persistent Znode `ActiveBreadCrumb`, сохраняя в нем информацию о себе (имя хоста и TCP-порт RM2), поскольку RM2 теперь является активным RM.


RM2
attempts
to kill RM1
(Fencing)
ActiveStandbyElectorLock
(ephemeral znode)
ResourceManager 1
(RM1)
ResourceManager 2
(RM2)
ActiveStandbyElectorLock
(ephemeral znode)
ActiveBreadCrumb
(persistent znode)
ZooKeeper

Изолирование конкурирующего менеджера ресурсов

Действия клиентов, ApplicationMaster и NodeManager при переключении RM

При наличии нескольких RM файл конфигурации *yarn-site.xml*, используемый клиентами и узлами, должен содержать список всех RM.

Клиенты, мастеры приложений (ApplicationMasters) и менеджеры узлов (NodeManagers) пытаются подключиться к RM, перебирая этот список по кругу, пока не обнаружат активный RM. Если используемый ими RM перестает быть активным, они проходятся далее по списку, пока не обнаружат "новый" активный RM.

Политика переключения на новый активный RM определяется с помощью одного из следующих параметров настройки:

- Если используется режим HA, логику попыток определяет параметр `yarn.client.failover-proxy-provider`, по умолчанию указывающий на класс `org.apache.hadoop.yarn.client.ConfiguredRMFailoverProxyProvider`. Эту логику можно изменить, назначив другой класс, расширяющий `org.apache.hadoop.yarn.client.RMFailoverProxyProvider`.
- Если режим HA не используется, логику попыток определяет параметр `yarn.client.failover-no-ha-proxy-provider`.

Примеры настройки YARN HA

Альберт Багдасарян

Содержание

Назад к содержанию

Параметры настройки

Пример конфигурации

Команды администратора

Параметры настройки

Отказоустойчивость настраивают с помощью различных параметров конфигурации, необходимые и наиболее важные из которых приведены в следующей таблице.

Файл *yarn-default.xml* содержит полный список параметров, включая значения по умолчанию. Инструкции по настройке хранилища состояний (state-store) приведены в документе ResourceManager Restart.

Параметр настройки	Описание
hadoop.zk.address	Адреса серверов ZooKeeper в виде списка пар хост:порт. Используется как хранилищем состояний (state store) HDFS, так и встроенным в ZooKeeper инструментом выбора лидера
yarn.resourcemanager.ha.enabled	Включение автоматической отказоустойчивости RM HA
yarn.resourcemanager.ha.rm-ids	Список логических идентификаторов ResourceManagers (RM), например `rm1,rm2`
yarn.resourcemanager.hostname.rm-id	Для каждого `rm-id` указывают hostname узла, на котором установлен RM. В качестве альтернативы можно задавать IP-адреса этих узлов
yarn.resourcemanager.address.rm-id	Для каждого `rm-id` указывают host:port для отправки задач клиентами. Если параметр задан, то значение hostname, указанное в `yarn.resourcemanager.hostname.rm-id`, игнорируется для указанной цели
yarn.resourcemanager.scheduler.address.rm-id	Для каждого `rm-id` указывают хост:порт планировщика для получения ресурсов ApplicationMaster. Если параметр задан, то значение hostname, указанное в `yarn.resourcemanager.hostname.rm-id`, игнорируется для указанной цели
yarn.resourcemanager.resource-tracker.address.rm-id	Для каждого `rm-id` указывают host:port для подключения NodeManagers. Если параметр задан, то значение hostname, указанное в `yarn.resourcemanager.hostname.rm-id`, игнорируется для указанной цели
yarn.resourcemanager.admin.address.rm-id	Для каждого `rm-id` указывают host:port для административных команд. Если параметр задан, то значение hostname, указанное в `yarn.resourcemanager.hostname.rm-id`, игнорируется для указанной цели
yarn.resourcemanager.webapp.address.rm-id	Для каждого `rm-id` указывают host:port web-приложения RM. Это не нужно, если параметр `yarn.http.policy` имеет значение HTTPS_ONLY. Если параметр задан, то значение hostname, указанное в `yarn.resourcemanager.hostname.rm-id`, игнорируется для указанной цели
yarn.resourcemanager.webapp.https.address.rm-id	Для каждого `rm-id` указывают host:port web-приложения RM для доступа по HTTPS. Это не нужно, если параметр `yarn.http.policy` имеет значение HTTP_ONLY. Если параметр задан, то значение hostname, указанное в `yarn.resourcemanager.hostname.rm-id`, игнорируется для указанной цели
yarn.resourcemanager.ha.id	Идентифицирует данный RM среди других RM, входящих в "ансамбль". Параметр опционален, однако, если его значение задано, то администратор должен убедиться, что собственный идентификатор указан в настройках каждого RM
yarn.resourcemanager.ha.automatic-failover.enabled	Включение автоматической отказоустойчивости; по умолчанию включено, но сервис действует только при включенном HA
yarn.resourcemanager.ha.automatic-failover.embedded	Использование встроенного метода выборов лидера для назначения активного RM. По умолчанию параметр включен, но метод действует только тогда, когда включены одновременно HA и `automatic-failover`
yarn.resourcemanager.cluster-id	Идентификатор кластера. Используется при выборе лидера для того, чтобы RM не получил активную роль в другом кластере
yarn.client.failover-proxy-provider	Класс, который будет использован клиентами, Application Masters (AM) и NodeManagers (NM) для аварийного переключения на активный RM
yarn.client.failover-no-ha-proxy-provider	Класс, который будет использован клиентами, AM и NM для аварийного переключения на активный RM, когда кластер не работает в режиме HA
yarn.client.failover-max-attempts	Максимально допустимое количество попыток сервиса FailoverProxyProvider восстановить работоспособность кластера
yarn.client.failover-sleep-base-ms	База для вычисления задержки (в миллисекундах), используемая для расчета экспоненциальной задержки между выборами активного RM при обеспечении отказоустойчивости
yarn.client.failover-sleep-max-ms	Максимально допустимое время задержки (в миллисекундах) между выборами активного RM при обеспечении отказоустойчивости
yarn.client.failover-retries	Максимально допустимое количество повторных попыток подключения к RM при каждой попытке восстановления работоспособности кластера
yarn.client.failover-retries-on-socket-timeouts	Максимально допустимое количество повторных попыток подключения к активному RM при тайм-ауте сокета

Пример конфигурации

Пример минимальной настройки для отказоустойчивости RM.

```xml
<property>
  <name>yarn.resourcemanager.ha.enabled</name>
  <value>true</value>
</property>
<property>
  <name>yarn.resourcemanager.cluster-id</name>
  <value>cluster1</value>
</property>
<property>
  <name>yarn.resourcemanager.ha.rm-ids</name>
  <value>rm1,rm2</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname.rm1</name>
  <value>master1</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname.rm2</name>
  <value>master2</value>
</property>
<property>
  <name>yarn.resourcemanager.webapp.address.rm1</name>
  <value>master1:8088</value>
</property>
<property>
  <name>yarn.resourcemanager.webapp.address.rm2</name>
  <value>master2:8088</value>
</property>
<property>
  <name>hadoop.zk.address</name>
  <value>zk1:2181,zk2:2181,zk3:2181</value>
</property>
```

Команды администратора

Команда `yarn rmadmin` имеет несколько специфических параметров для проверки состояния RM и перехода в состояние active или standby. Она принимают в качестве аргумента идентификатор RM, заданный параметром `yarn.resourcemanager.ha.rm-ids` в настройках YARN.

Запрос состояния RM1:

```
$ yarn rmadmin -getServiceState rm1
```

Вывод на экран:

```
active
```

Аналогично запрос состояния RM2:

```
$ yarn rmadmin -getServiceState rm2
standby
```

Если включена автоматическая отказоустойчивость, переход в состояние active или standby в ручном режиме будет недоступен.

> **ВНИМАНИЕ**
>
> Данное ограничение можно обойти с помощью флага `-forcemanual`.

Пример запроса на переключение:

```
$ yarn rmadmin -transitionToStandby rm1
```

Пример отказа:

```
Automatic failover is enabled for org.apache.hadoop.yarn.client.RMHAServiceTarget@1d8299fd
Refusing to manually manage HA state, since it may cause
a split-brain scenario or other incorrect state.
If you are very sure you know what you are doing, please
specify the forcemanual flag.
```

Смотрите также справочное руководство YARN CLI для часто используемых команд YARN.

Логирование в YARN

Елена Костюченко

Содержание

Назад к содержанию

Обзор

Поиск по логам

Уровни логирования

Логи контейнеров

Агрегирование логов

Обзор

YARN записывает выводы команд и события процессов в текстовые логи. Эта информация может быть полезна при диагностике технических проблем.

Каждый компонент YARN хранит логи локально, на хосте, на котором он установлен. Логи имеют расширение *.log* и расположены в директории *var/log/hadoop-yarn/*. Здесь же находятся файлы *.out*, содержащие информацию о запуске компонентов.

Просмотреть логи YARN в браузере можно через веб-интерфейсы YARN.

Имена файлов с логами имеют следующий формат: `hadoop-yarn-<компонент>-<хост>.log`.

Здесь:

- `<компонент>` — имя компонента, например, ResourceManager;
- `<хост>` — FQDN хоста компонента.

Конфигурация логов YARN настраивается в файле log4j.properties вместе с HDFS.

Поиск по логам

Чтобы найти в логах интересующую вас информацию, например, сообщения об ошибках, подключитесь к хосту с компонентом, логи которого вы хотите просмотреть, и воспользуйтесь командой `grep`.

Например:

```
$ cat /var/log/hadoop-yarn/hadoop-yarn-nodemanager-elenas-adh3.ru-central1.internal.log | grep -i -A3 -B1 error
```

Эта команда просматривает лог компонента NodeManager, расположенного на хосте `elenas-adh3.ru-central1.internal`, и ищет сообщения, содержащие слово `error`. Опция `-i` позволяет игнорировать различия в регистре, а опции `-A3 -B1` добавляют к найденной строке с ошибкой одну строчку до нее и три строчки после.

Пример вывода команды:

```
2024-04-08 13:26:24,910 INFO org.apache.hadoop.yarn.server.nodemanager.containermanager.ContainerManagerImpl: ContainerManager
bound to 0.0.0.0/0.0.0.0:8041
2024-04-08 13:26:24,912 ERROR org.apache.hadoop.yarn.server.nodemanager.containermanager.scheduler.ContainerScheduler: Unknown
event arrived at ContainerScheduler: EventType: RECOVERY_COMPLETED
2024-04-08 13:26:24,915 INFO org.apache.hadoop.yarn.server.nodemanager.webapp.WebServer: Instantiating NMWebApp at 0.0.0.0:8042
2024-04-08 13:26:24,935 INFO org.eclipse.jetty.util.log: Logging initialized @56117ms
2024-04-08 13:26:25,004 INFO org.apache.hadoop.security.authentication.server.AuthenticationFilter: Unable to initialize
FileSignerSecretProvider, falling back to use random secrets.
```

Уровни логирования

YARN использует для создания логов библиотеку Log4j2, которая поддерживает следующие уровни логирования (от наименее информативного к наиболее информативному):

1. `FATAL` — сообщает, что операция не может быть выполнена и будет завершена.
2. `ERROR` — уведомляет, что программа работает неправильно или прекратила работу.
3. `WARN` — предупреждает о потенциальных проблемах. Это означает, что программа работает не по стандартному сценарию и в будущем могут возникнуть проблемы.
4. `INFO` — передает информацию о жизненном цикле или состоянии программы.
5. `DEBUG` — выводит отладочную информацию о внутренних состояниях программы.
6. `TRACE` — выводит сообщения, отслеживающие ход выполнения программы.

Logger Log4j также принимает значения уровня логирования: `OFF` — для отключения логирования, и `ALL` — для разрешения всех типов сообщений.

Включение одного уровня логирования включит этот уровень и все уровни до него. Например, если установлен уровень логирования `WARN`, то в логи будут попадать только `WARN`, `ERROR` и `FATAL`, но не `INFO`, `DEBUG` и `TRACE`.

Логи контейнеров

Приложения, запущенные в YARN, генерируют собственные логи. Чтобы посмотреть логи приложения, используйте команду logs.

Агрегирование логов

Агрегирование логов — это функция YARN, которая копирует логи приложений в один файл в HDFS после завершения работы приложения.

По умолчанию агрегированные логи сохраняются в директорию *hdfs://<адрес NameNode>:8020/logs/yarn/logs/*, где `<адрес NameNode>` — это IP-адрес или FQDN активной NameNode.

Агрегирование логов можно настроить с помощью параметров логирования в файле конфигурации *yarn-site.xml*.

Параметры агрегирования логов YARN

Параметр	Описание	Значение по умолчанию
yarn.log.server.url	URL, по которому отображаются логи приложений. Если задан сервер JobHistory, логи будут доступны в веб-интерфейсе Resource Manager	—
yarn.log-aggregation-enable	Включает агрегирование логов	true
yarn.log-aggregation.retain-seconds	Время в секундах, по истечении которого логи будут удалены. Значение `-1` или ниже отключит удаление логов. Использование слишком маленького отрезка времени создаст большую нагрузку на NameNode	172800
yarn.nodemanager.remote-app-log-dir	HDFS-директория для сохранения логов приложений	/logs

Чтобы изменить параметры логирования через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на YARN.
3. Найдите поле ***yarn-site.xml***.
4. Выберите параметр и внесите необходимые изменения.
5. Подтвердите изменения в конфигурации, нажав ***Save***.
6. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Настройка и использование GPU в YARN

Елена Костюченко

Содержание

Назад к содержанию

Обзор

Конфигурация YARN для GPU

Запуск приложений YARN на GPU

Обзор

Для приложений YARN, требующих интенсивных вычислений, Hadoop поддерживает перенос рассчетов на графический процессор вместо CPU.

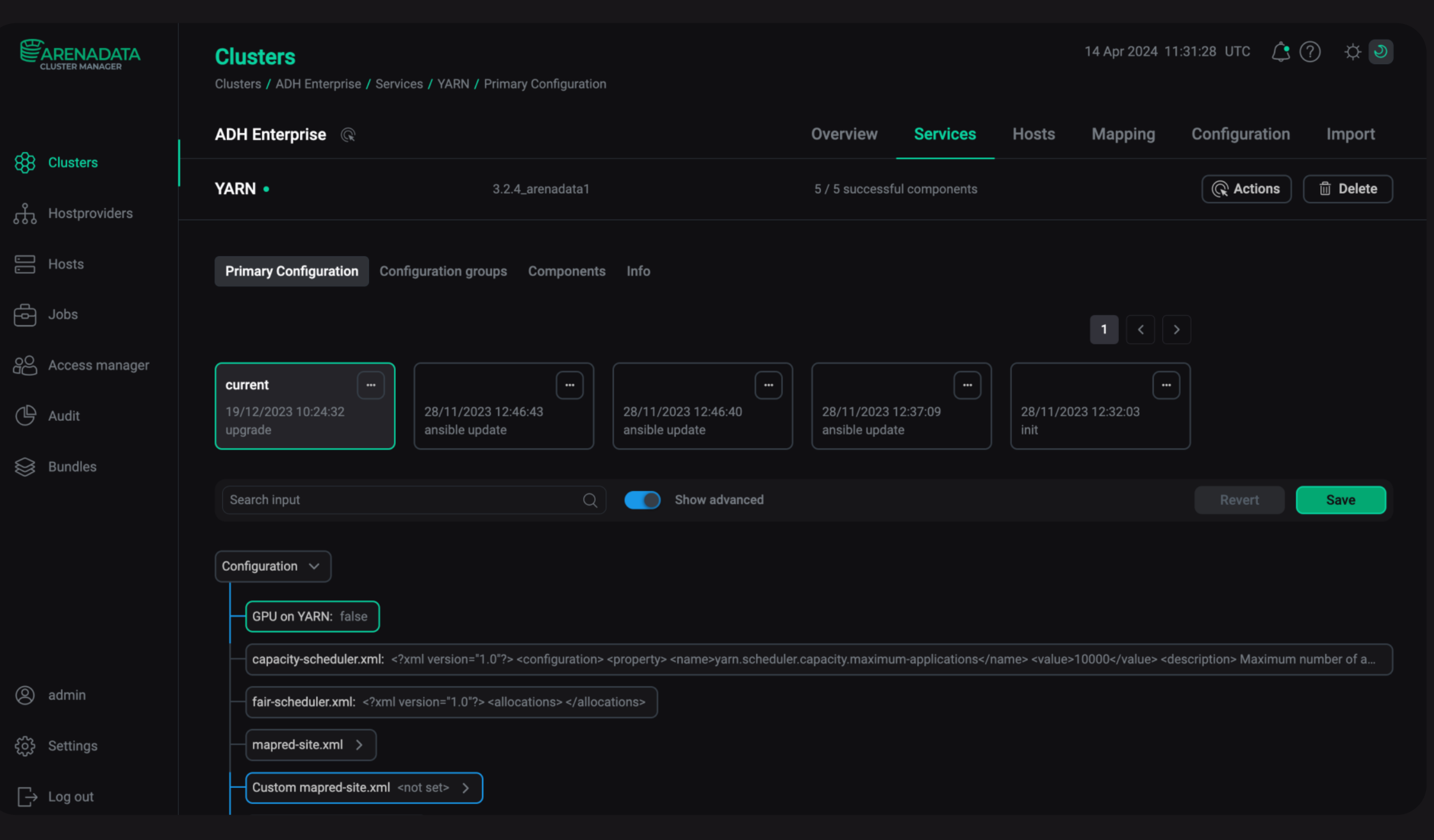
Условия для использования GPU в YARN:

- Графический процессор Nvidia.
- Драйверы Nvidia.
- Если используется Docker, необходимо установить nvidia-docker 1.0.
- YARN должен использовать Capacity scheduler.

Конфигурация YARN для GPU

Чтобы настроить поддержку GPU в YARN через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите в ***Services*** и нажмите на YARN.
3. Включите опцию ***Show Advanced***, найдите параметр ***GPU on YARN*** и установите для него значение `true`.



GPU on YARN: false

capacity-scheduler.xml: <?xml version="1.0"?> <configuration> <property> <name>yarn.scheduler.capacity.maximum-applications</name> <value>10000</value> <description> Maximum number of a...

fair-scheduler.xml: <?xml version="1.0"?> <allocations> </allocations>

mapred-site.xml

Custom mapred-site.xml <not set>

Настройка GPU on YARN в ADCM

4. Найдите и откройте свойство ***capacity-scheduler.xml***.
5. В открывшемся окне измените значение параметра `yarn.scheduler.capacity.resource-calculator` на `org.apache.hadoop.yarn.util.resource.DominantResourceCalculator`:

```
<configuration>
  <property>
    <name>yarn.scheduler.capacity.resource-calculator</name>
    <value>org.apache.hadoop.yarn.util.resource.DominantResourceCalculator</value>
  </property>
</configuration>
```

6. В группе параметров ***yarn-site.xml*** отредактируйте параметры так, чтобы создать желаемую конфигурацию для вашего кластера, которая будет учитывать использование графического процессора. Например:

```
<configuration>
  <property>
    <name>yarn.nodemanager.resource-plugins</name>
    <value>yarn.io/gpu</value>
  </property>
  <property>
     <name>yarn.resource-types</name>
     <value>yarn.io/gpu</value>
  </property>
  <property>
     <name>yarn.nodemanager.resource-plugins.gpu.allowed-gpu-devices</name>
     <value>auto</value>
  </property>
  <property>
     <name>yarn.nodemanager.resource-plugins.gpu.path-to-discovery-executables</name>
     <value>/usr/bin</value>
  </property>
  <property>
     <name>yarn.nodemanager.linux-container-executor.cgroups.mount</name>
     <value>true</value>
  </property>
  <property>
     <name>yarn.nodemanager.linux-container-executor.cgroups.mount-path</name>
     <value>/sys/fs/cgroup</value>
  </property>
  <property>
     <name>yarn.nodemanager.linux-container-executor.cgroups.hierarchy</name>
     <value>yarn</value>
  </property>
  <property>
    <name>yarn.nodemanager.container-executor.class</name>
    <value>org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor</value>
  </property>
  <property>
    <name>yarn.nodemanager.linux-container-executor.group</name>
    <value>yarn</value>
  </property>
</configuration>
```

7. Найдите параметр ***container-executor.cfg template*** и добавьте к нему следующие значения:

```
yarn.nodemanager.linux-container-executor.group=yarn

#--Original container-exectuor.cfg Content--

[gpu]
module.enabled=true
[cgroups]
root=/sys/fs/cgroup
yarn-hierarchy=yarn
```

Если для запуска приложений будет использоваться Docker, добавьте дополнительные свойства в ***container-executor.cfg template***:

```
[docker]
docker.allowed.devices=/dev/nvidiactl,/dev/nvidia-uvm,/dev/nvidia-uvm-tools,/dev/nvidia1,/dev/nvidia0 ①
docker.allowed.volume-drivers ②
docker.allowed.ro-mounts=nvidia_driver_375.66 ③
docker.allowed.runtimes=nvidia ④
```

① Добавляет устройства с графическим процессором в раздел Docker. Введите список значений, разделенных запятой. Список устройств можно получить с помощью команды `ls /dev/nvidia*`.

② Добавляет `nvidia-docker` в белый список volume-driver.

③ Добавляет `nvidia_driver_<version>` в белый список точек монтирования, доступных только для чтения.

④ Чтобы использовать *nvidia-docker-v2* в качестве плагина Docker для GPU, добавьте `nvidia` в белый список среды выполнения.

8. Нажмите ***Apply*** и подтвердите изменения в конфигурации YARN, нажав ***Save***.
9. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***. Это автоматически обновит все необходимые файлы конфигурации на всех хостах.

ПРИМЕЧАНИЕ

В некоторых случаях может потребоваться предоставить доступ всем пользователям к локальному каталогу NodeManager, чтобы избежать ошибок, связанных с разрешениями.

В зависимости от конфигурации вашего кластера, для некоторых параметров конфигурации YARN для GPU можно использовать значения по умолчанию.

GPU в YARN: опциональные параметры

Параметр	Описание	Значение по умолчанию
yarn.nodemanager.resource-plugins	Значение `yarn.io/gpu` этого параметра позволяет включить модуль изоляции GPU на стороне NodeManager. По умолчанию YARN автоматически обнаруживает и настраивает графические процессоры	—
yarn.nodemanager.resource-plugins.gpu.allowed-gpu-devices	Если установлено значение по умолчанию, YARN автоматически обнаружит ресурсы GPU в системе. Если этого не случилось или при обнаружении возникла ошибка, вы можете указать список GPU-устройств в этом параметре вручную в следующем формате: `GPU1_index:GPU1_minor_number, GPU2_index:GPU2_minor_number`. Например: `0:0,1:1,2:2`	auto
yarn.nodemanager.resource-plugins.gpu.path-to-discovery-executables	Путь к бинарному файлу для обнаружения GPU, который использует NodeManager для сбора информации о графических процессорах, если для параметра `yarn.nodemanager.resource.gpu.allowed-gpu-devices` установлено значение `auto`. Например: `/usr/local/bin/nvidia-smi`. Если путь не задан, NodeManager попытается найти бинарный файл самостоятельно	—
yarn.nodemanager.resource-plugins.gpu.docker-plugin	Плагин команд Docker для GPU. По умолчанию используется Nvidia Docker v1.0.	nvidia-docker-v1
yarn.nodemanager.resource-plugins.gpu.docker-plugin.nvidia-docker-v1.endpoint	Эндпойнт плагина Nvidia Docker	http://localhost:3476/v1.0/docker/cli
yarn.nodemanager.linux-container-executor.cgroups.mount	Использование контроллера устройств CGroup для изоляции GPU	true

Запуск приложений YARN на GPU

Ниже приведены примеры запуска приложений YARN с использованием команды yarn jar в контейнере YARN или Docker.

YARN-контейнер

Чтобы запустить приложение в контейнере YARN, выполните:

```
$ yarn jar <path/to/hadoop-yarn-applications-distributedshell.jar> \
  -jar <path/to/hadoop-yarn-applications-distributedshell.jar> \
  -shell_command /usr/local/nvidia/bin/nvidia-smi \
  -container_resources memory-mb=3072,vcores=1,yarn.io/gpu=2 \
  -num_containers 2
```

Эта команда запрашивает 2 контейнера по 3 ГБ каждый, 1 виртуальное ядро и 2 графических процессора.

Docker-контейнер

Чтобы запустить приложение в контейнере Docker, выполните:

```
$ yarn jar <path/to/hadoop-yarn-applications-distributedshell.jar> \
       -jar <path/to/hadoop-yarn-applications-distributedshell.jar> \
       -shell_env YARN_CONTAINER_RUNTIME_TYPE=docker \
       -shell_env YARN_CONTAINER_RUNTIME_DOCKER_IMAGE=<docker-image-name> \
       -shell_command nvidia-smi \
       -container_resources memory-mb=3072,vcores=1,yarn.io/gpu=2 \
       -num_containers 2
```

Эта команда запрашивает 2 контейнера по 3 ГБ каждый, 1 виртуальное ядро и 2 графических процессора.

Управление сервисом YARN через ADCM

Елена Костюченко

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом YARN. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса YARN, перечислены в таблице ниже.

Подробное описание компонентов YARN приведено в статье Архитектура YARN.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты YARN
NodeManager Decommission/Recommission	Позволяет выводить из эксплуатации и восстанавливать компоненты NodeManager. Данная функция может использоваться для временного вывода NodeManager из эксплуатации или для его безопасного удаления. При выполнении этого действия необходимо указать хосты с компонентами NodeManager, которые нужно вывести из эксплуатации, в списке ***Hosts in decommissioned mode***. Для каждого элемента в списке введите полное доменное имя (FQDN) хоста (поле ***Enter field name***), а также время старта вывода из эксплуатации в секундах (поле ***Enter field value***). По умолчанию процесс вывода из эксплуатации стартует моментально. Когда все хосты указаны, проверьте конфигурацию и запустите действие. Чтобы применить Graceful Decommission, воспользуйтесь командой `refreshNodes` с параметром `-g`
Manage Ranger plugin	Включает или отключает плагин Ranger для YARN
Add/modify node labels	Добавляет или редактирует метки узлов. Метки узлов можно использовать для группирования узлов по типам в зависимости от их назначения или характеристик, например, для распределения ресурсов узлов между очередями. При выборе этого действия введите метки, которые будут использоваться в кластере, и их эксклюзивность (`True` или `False`). Эксклюзивность по умолчанию — `True`
Add/modify node-to-labels mapping	Назначает хостам метки узлов, добавленные с помощью действия ***Add/modify node labels***. При выполнении этого действия необходимо указать FQDN хостов с компонентом NodeManager и метки, которые следует для них назначить
Reload Scheduler config	Перезагружает списки управления доступом, состояния и свойства планировщика. Используйте это действие, чтобы применить изменения, внесенные в файл конфигурации CapacityScheduler. FairScheduler не требует перезагрузки после изменения конфигурации
Remove node label	Удаляет метки узлов из кластера. При использовании этого действия введите список меток, которые хотите удалить, через запятую
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются. Сервис поддерживает функцию ***Rolling restart***, которая позволяет перезапускать компоненты NodeManager по одному (или партиями), а не все сразу. При использовании режима высокой доступности данная опция позволяет выполнять рестарт сервиса YARN без потери активных YARN-контейнеров. Опция ***Rolling restart*** имеет следующие параметры: ▪ ***batch_size*** — количество компонентов NodeManager, которые будут перезапущены за одну итерацию. ▪ ***batch_delay*** — задержка в секундах между перезапусками партий компонентов. ▪ ***health_checks*** — указывает, нужно ли выполнять проверку компонентов после рестарта. ▪ ***max_failed_batches_number*** — максимально допустимое количество партий компонентов NodeManager с ошибками при рестарте. При достижении этого значения ADCM-действие ***Restart*** завершается с ошибкой.

Действия с компонентами YARN

Действие	Описание
Check	Проверяет, правильно ли работают все экземпляры компонента в кластере
Restart	Перезапускает все экземпляры компонента в кластере
Start	Запускает все экземпляры компонента в кластере
Stop	Останавливает все экземпляры компонента в кластере

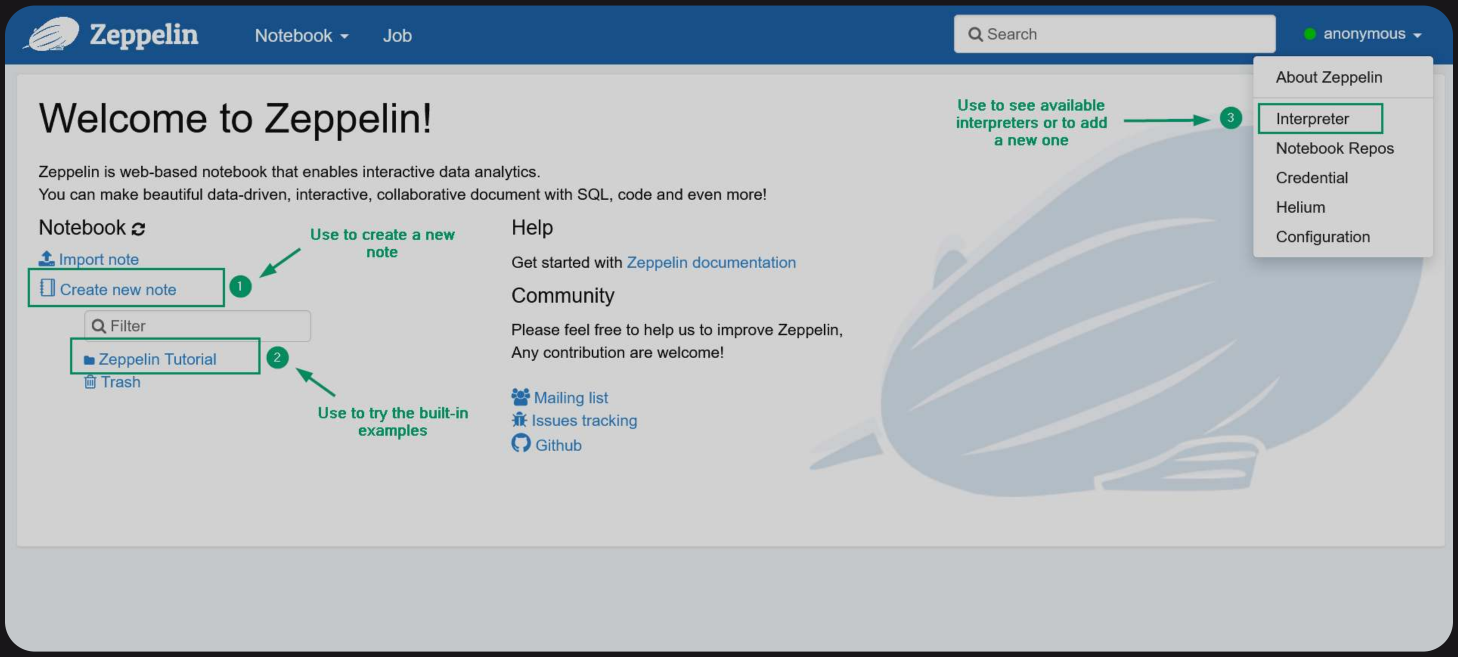
Обзор Zeppelin UI

Константин Алпашкин

Назад к содержанию

Zeppelin предоставляет веб-интерфейс, позволяющий работать с интерпретаторами и заметками. Веб-интерфейс доступен по следующему URL: *http://<IP>:8180/*, где *<IP>* — это IP-адрес вашего Zeppelin-сервера.

Ниже показана домашняя страница веб-интерфейса Zeppelin:



Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Get started with Zeppelin documentation

Please feel free to help us to improve Zeppelin,
Any contribution are welcome!

Zeppelin
Notebook
Job
Search
anonymous
About Zeppelin
Interpreter
Notebook Repos
Credential
Helium
Configuration
Use to see available interpreters or to add a new one
3
Notebook
Import note
Create new note
1
Use to create a new note
Filter
Zeppelin Tutorial
Trash
2
Use to try the built-in examples
Help
Community
Mailing list
Issues tracking
Github

Домашняя страница Zeppelin

Для того, чтобы начать работать с Zeppelin, используйте один из способов ниже:

1. **Создание новой заметки**. Чтобы создать новую заметку, кликните элемент списка ***Create new note*** на домашней странице слева. Эта опция также доступна, если вы развернете элемент главного меню ***Notebook***. Примеры создания заметок, основанные на различных интерпретаторах, доступны в секции Примеры создания заметок в Zeppelin.
2. **Просмотр обучающих статей**. Кликнув на ***Zeppelin Tutorial*** на домашней странице слева, вы увидите список примеров. Вы можете изучить примеры и попробовать выполнить код из примеров самостоятельно.
3. **Просмотр существующих интерпретаторов или добавление нового**. Для начала работы с интерпретаторами, выберите пункт ***Interpreter*** в меню аккаунта, расположенном в верхнем правом углу домашней страницы. Пример добавления кастомного интерпретатора описан в статье Добавление интерпретатора в группу.

ПРИМЕЧАНИЕ

Подробное описание веб-интерфейса доступно в документации Zeppelin. Мы рекомендуем ознакомиться с документацией перед началом работы.

Добавление интерпретатора в группу

Константин Алпашкин

Содержание

Назад к содержанию

Подготовка

Метод 1. Использование префиксов

Метод 2. Добавление нового интерпретатора

Список встроенных интерпретаторов Zeppelin доступен в веб-интерфейсе. Для этого выберите ***Interpreter*** в меню аккаунта. Также вы можете добавить свои интерпретаторы в существующие группы. Данная статья описывает два способа создания интерпретаторов, в качестве примера рассматривается добавление Phoenix-интерпретатора в группу `jdbc`.

ПРИМЕЧАНИЕ

Вы можете найти дополнительную информацию в документации Zeppelin. Zeppelin также позволяет создавать свои интерпретаторы, которые не используют существующие группы, а также устанавливать сторонние интерпретаторы.

Подготовка

Как описано в статье Подключение к Phoenix, вы можете работать с базами данных Phoenix через JDBC. В примерах, используемых в этой статье, используется Phoenix Thin Driver. Перед тем, как использовать любой из двух методов, выполните следующие шаги по подготовке:

1. Определите необходимый вам JAR-файл, название которого содержит версии HBase и Phoenix и имеет вид *phoenix-<phoenix_version>-HBase-<hbase_version>-thin-client.jar*. Например, для версий HBase 2.0 и Phoenix 5.0.0 следует использовать *phoenix-5.0.0-HBase-2.0-thin-client.jar*. Его можно скопировать из узла кластера HBase (из каталога */usr/lib/phoenix*) или загрузить из репозитория Phoenix.

2. Скопируйте подготовленный JAR-файл в директорию интерпретатора */usr/lib/zeppelin/interpreter/jdbc/*:

```
$ sudo cp /usr/lib/phoenix/phoenix-5.0.0-HBase-2.0-thin-client.jar /usr/lib/zeppelin/interpreter/jdbc/
```

3. Убедитесь, что файл успешно скопирован, используя следующую команду:

```
$ ls -la /usr/lib/zeppelin/interpreter/jdbc/
```

Результат выполнения команды выглядит следующим образом:

```
total 35412
drwxr-xr-x.  3 root root     4096 Dec  7 17:46 .
drwxr-xr-x. 28 root root     4096 Dec  7 08:36 ..
-rw-r--r--.  1 root root     2314 Apr 15  2021 ansi.sql.keywords
-rw-r--r--.  1 root root   167962 Apr 15  2021 commons-dbcp2-2.0.1.jar
-rw-r--r--.  1 root root    60686 Apr 15  2021 commons-logging-1.1.1.jar
-rw-r--r--.  1 root root   109568 Apr 15  2021 commons-pool2-2.3.jar
-rw-r--r--.  1 root root     4800 Apr 15  2021 interpreter-setting.json
-rw-r--r--.  1 root root   489884 Apr 15  2021 log4j-1.2.17.jar
drwxr-xr-x.  2 root root       55 Dec  6 09:10 META-INF
-rw-r--r--.  1 root root 34651233 Dec  7 17:46 phoenix-5.0.0-HBase-2.0-thin-client.jar
-rw-r--r--.  1 root root   648487 Apr 15  2021 postgresql-9.4-1201-jdbc41.jar
-rw-r--r--.  1 root root     6573 Apr 15  2021 postgresql-native-driver-sql.keywords
-rw-r--r--.  1 root root    32119 Apr 15  2021 slf4j-api-1.7.10.jar
-rw-r--r--.  1 root root     8866 Apr 15  2021 slf4j-log4j12-1.7.10.jar
-rw-r--r--.  1 root root    46253 Apr 15  2021 zeppelin-jdbc-0.8.1.jar
```

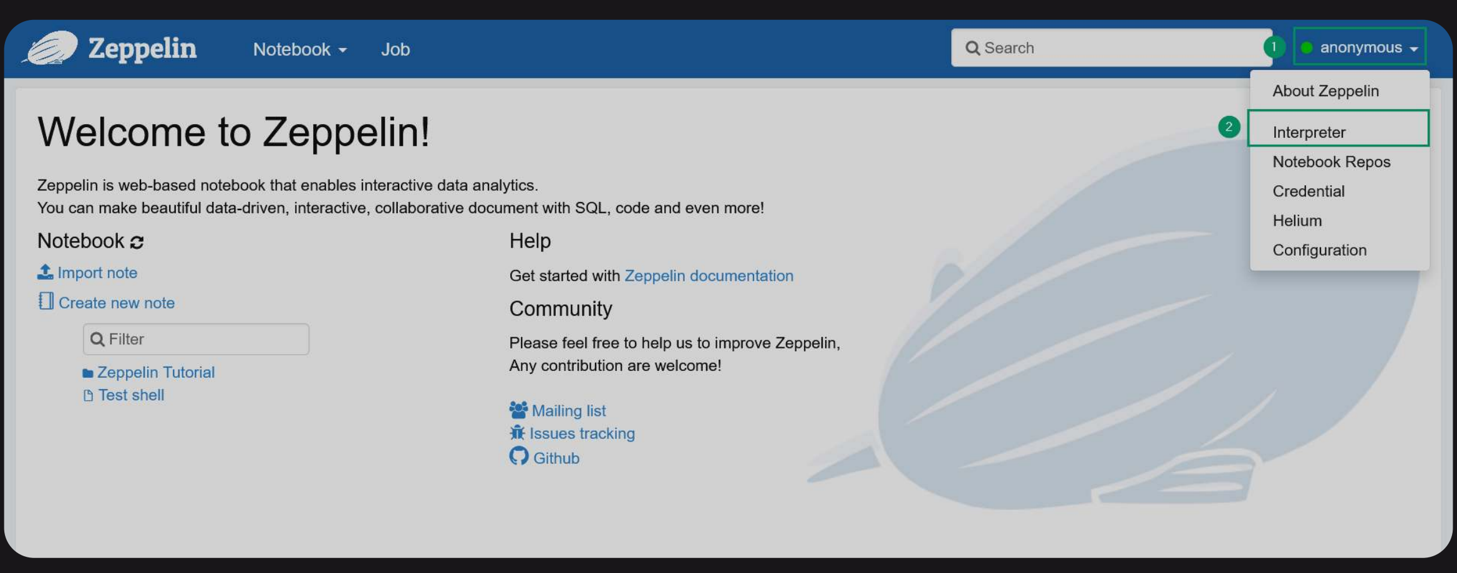
РЕКОМЕНДАЦИЯ

Начиная с версии Zeppelin 0.9.0, вы можете пропустить шаги подготовки и использовать Apache Phoenix guide для создания нового интерпретатора Phoenix через web UI. В более старых версиях могут наблюдаться ошибки при доступе к Maven-репозиториям с зависимостями. По этой причине рекомендуется подкладывать JAR-файлы вручную.

Метод 1. Использование префиксов

Данный метод подразумевает добавление определенного префикса к существующему интерпретатору. Таким образом, при работе с заметками вы можете использовать исходный интерпретатор не со стандартными настройками, а с настройками, которые имеют специальный префикс. Для вызова такого интерпретатора из заметки необходимо добавить префикс к имени интерпретатора, например `%jdbc(phoenix)`.

1. Выберите ***Interpreter*** в меню аккаунта, расположенном в правом верхнем углу домашней страницы Zeppelin.



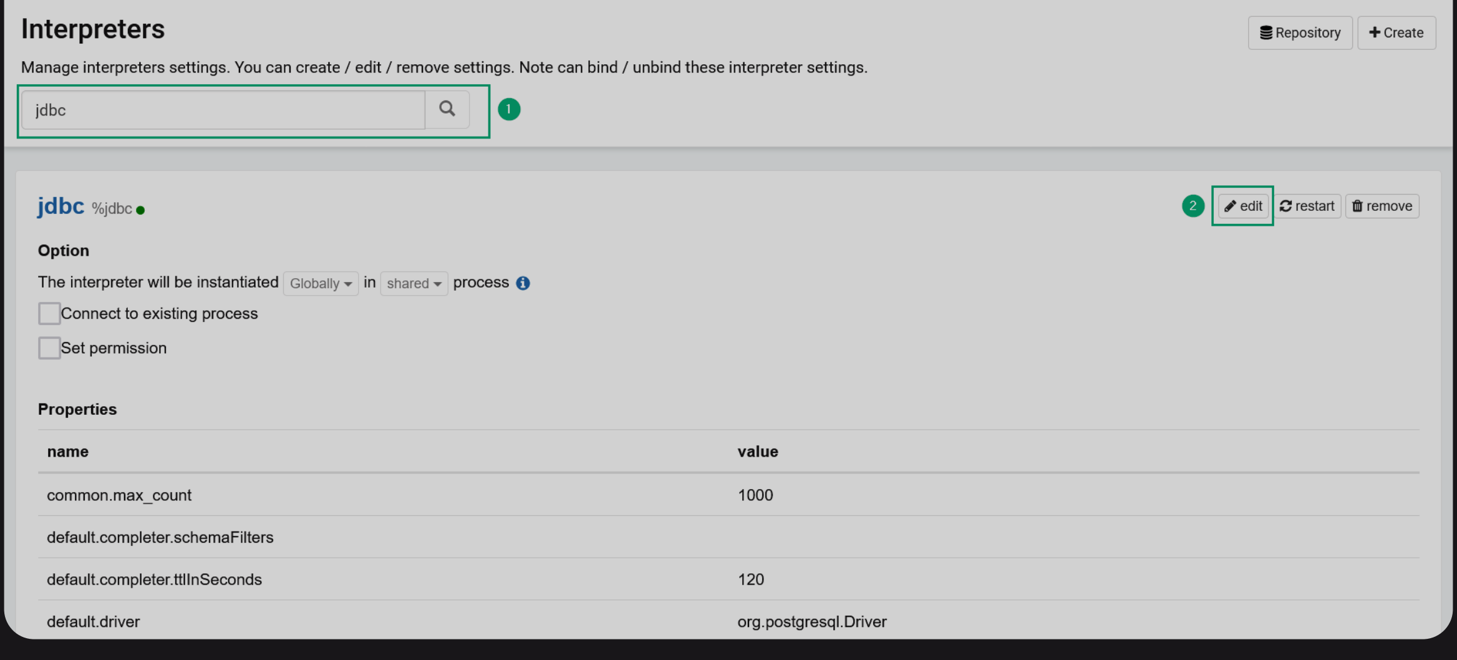
Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Get started with Zeppelin documentation

Please feel free to help us to improve Zeppelin,
Any contribution are welcome!

Список интерпретаторов

2. Используя поиск, найдите интерпретатор `jdbc` и кликните ***edit***.



Manage interpreters settings. You can create / edit / remove settings. Note can bind / unbind these interpreter settings.

jdbc

jdbc %jdbc

Option

The interpreter will be instantiated Globally in shared process

☐ Connect to existing process

☐ Set permission

Properties

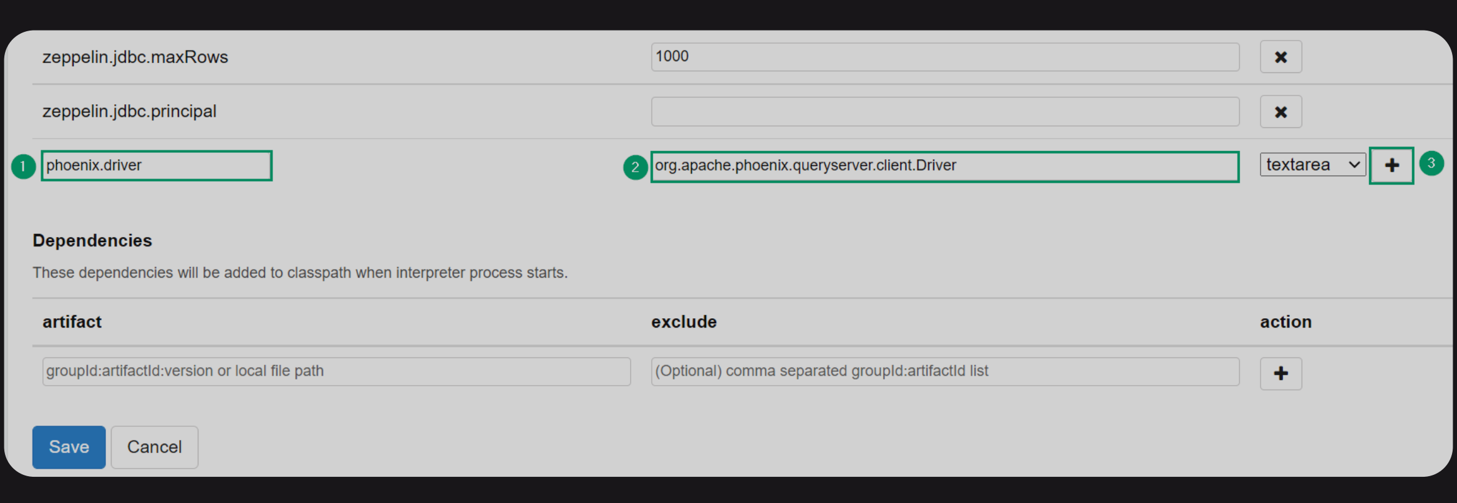
name	value
common.max_count	1000
default.completer.schemaFilters	
default.completer.ttlInSeconds	120
default.driver	org.postgresql.Driver

Редактирование JDBC-интерпретатора

3. Добавьте новые свойства в список настроек интерпретатора с префиксом `phoenix`.

Имя	Значение	Описание
phoenix.driver	org.apache.phoenix.queryserver.client.Driver	Драйвер для соединения с Phoenix
phoenix.url	jdbc:phoenix:thin:url=http://<IP>:8765;serialization=PROTOBUF	URL, используемый для соединения с Phoenix, где `<IP>` — это IP-адрес вашего Phoenix Query Server
phoenix.user	—	Логин для соединения с Phoenix
phoenix.password	—	Пароль для соединения с Phoenix

Чтобы добавить поле, введите его имя, значение, а затем кликните **+**, чтобы добавить следующее поле.



zeppelin.jdbc.maxRows 1000

zeppelin.jdbc.principal

phoenix.driver org.apache.phoenix.queryserver.client.Driver

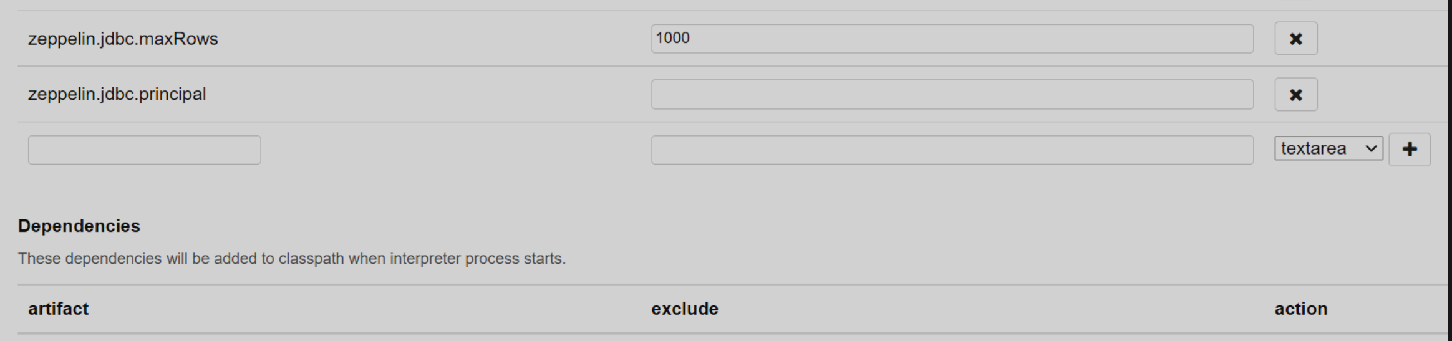
Dependencies

These dependencies will be added to classpath when interpreter process starts.

artifact	exclude	action
groupId:artifactId:version or local file path	(Optional) comma separated groupId:artifactId list	+

Добавление новых полей

4. После добавления всех необходимых полей, кликните ***Save***.



zeppelin.jdbc.maxConnLifetime -1

zeppelin.jdbc.maxRows 1000

zeppelin.jdbc.principal

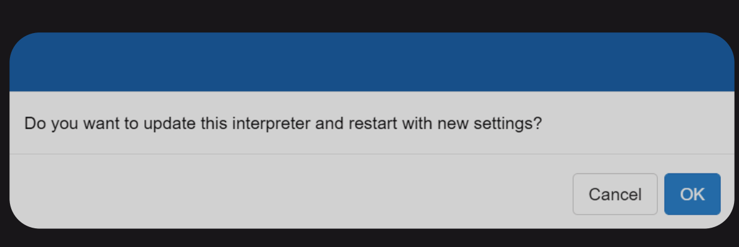
Dependencies

These dependencies will be added to classpath when interpreter process starts.

artifact	exclude	action
groupId:artifactId:version or local file path	(Optional) comma separated groupId:artifactId list	+

Сохранение JDBC-интерпретатора

5. Подтвердите операцию в открывшемся диалоге. Новый интерпретатор готов.



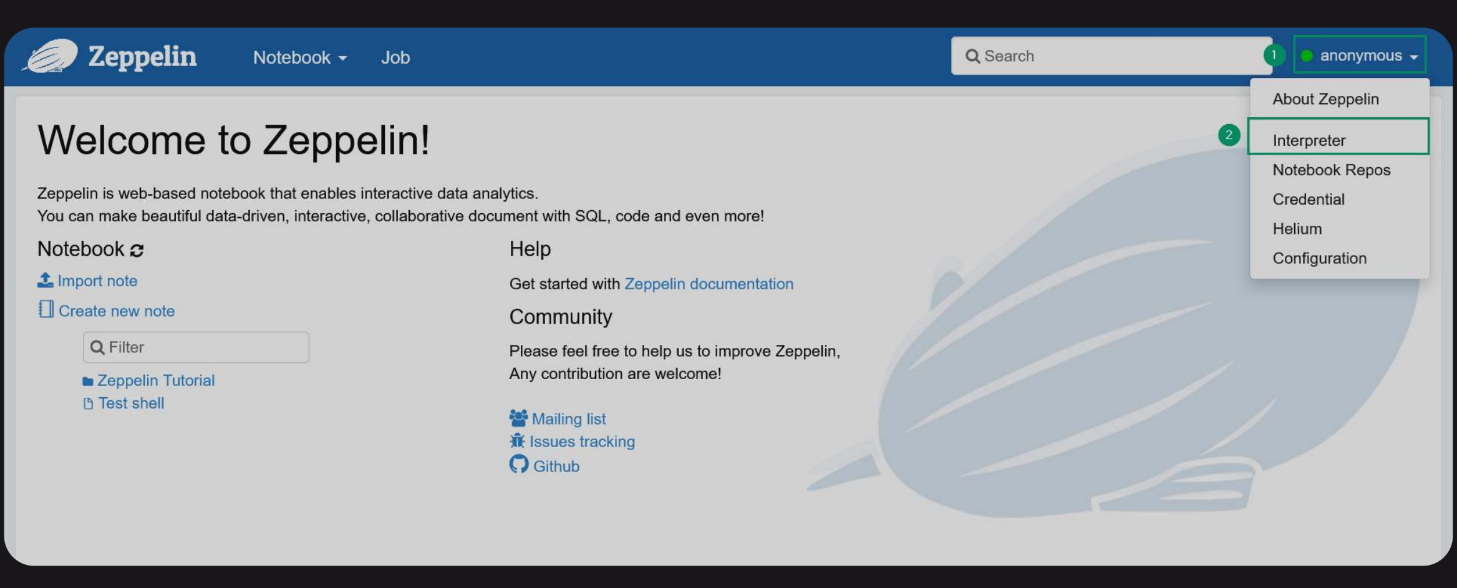
Do you want to update this interpreter and restart with new settings?

Подтверждение сохранения

Метод 2. Добавление нового интерпретатора

Данный метод подразумевает создание нового интерпретатора на основе уже существующего. Для этого выберите существующий интерпретатор для группы (в данном примере — `jdbc`) и замените его стандартные свойства на свои. Все библиотеки, которые были доступны для исходного интерпретатора, остаются доступными и для вновь созданного. Для вызова такого интерпретатор из заметки используйте новое имя интерпретатора: `%phoenix`.

1. Выберите ***Interpreter*** в меню аккаунта, расположенном в правом верхнем углу домашней страницы Zeppelin.



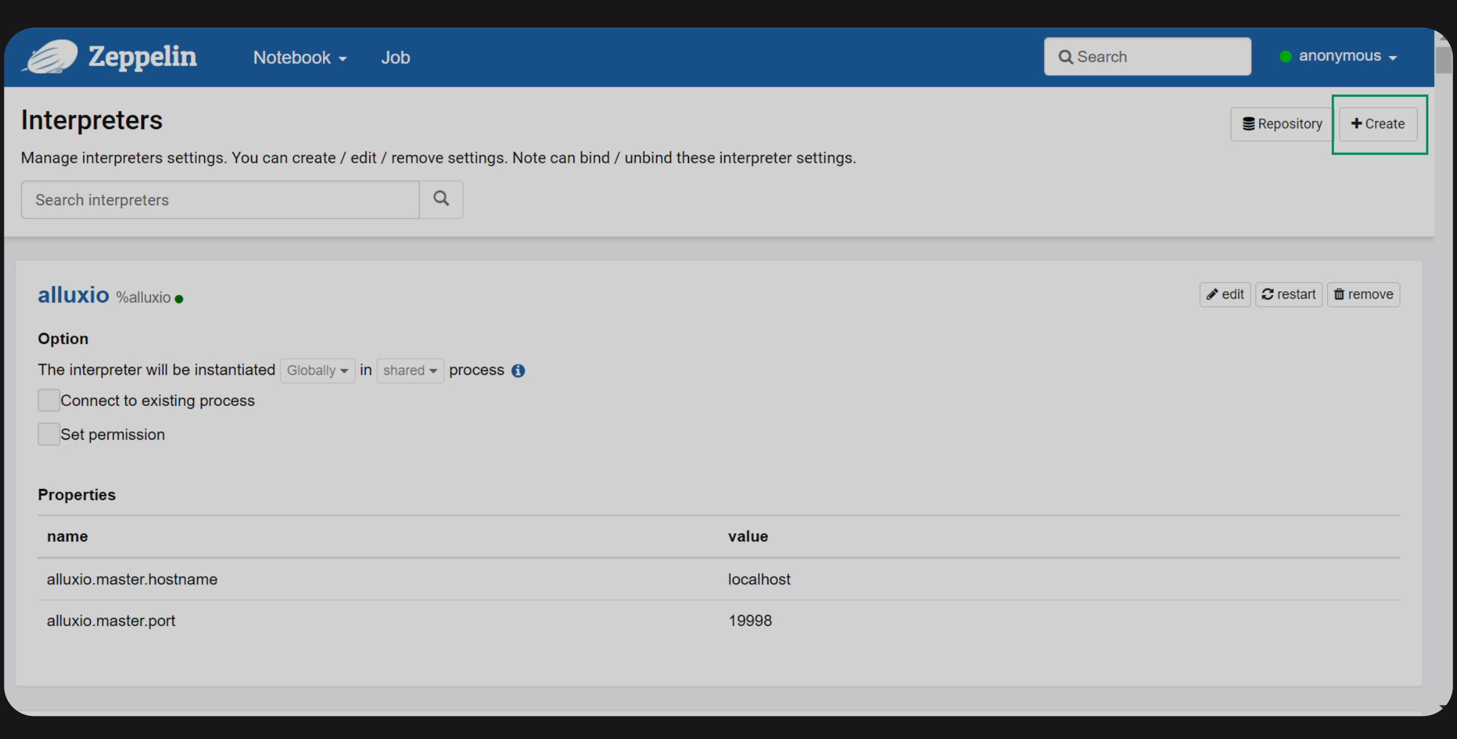
Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Get started with Zeppelin documentation

Please feel free to help us to improve Zeppelin,
Any contribution are welcome!

Список интерпретаторов

2. Кликните ***Create***.



Manage interpreters settings. You can create / edit / remove settings. Note can bind / unbind these interpreter settings.

alluxio %alluxio

Option

The interpreter will be instantiated Globally in shared process

☐ Connect to existing process

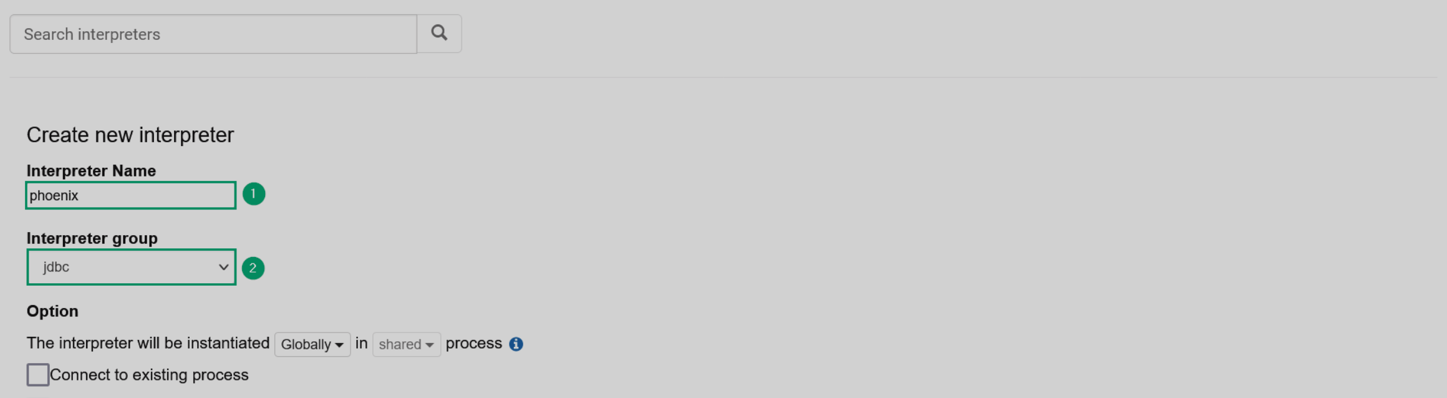
☐ Set permission

Properties

name	value
alluxio.master.hostname	localhost
alluxio.master.port	19998

Создание нового интерпретатора

3. Введите `phoenix` в поле ***Interpreter Name*** и выберите `jdbc` в поле ***Interpreter group***.



Manage interpreters settings. You can create / edit / remove settings. Note can bind / unbind these interpreter settings.

Create new interpreter

Interpreter Name: phoenix

Interpreter group: jdbc

Option

The interpreter will be instantiated Globally in shared process

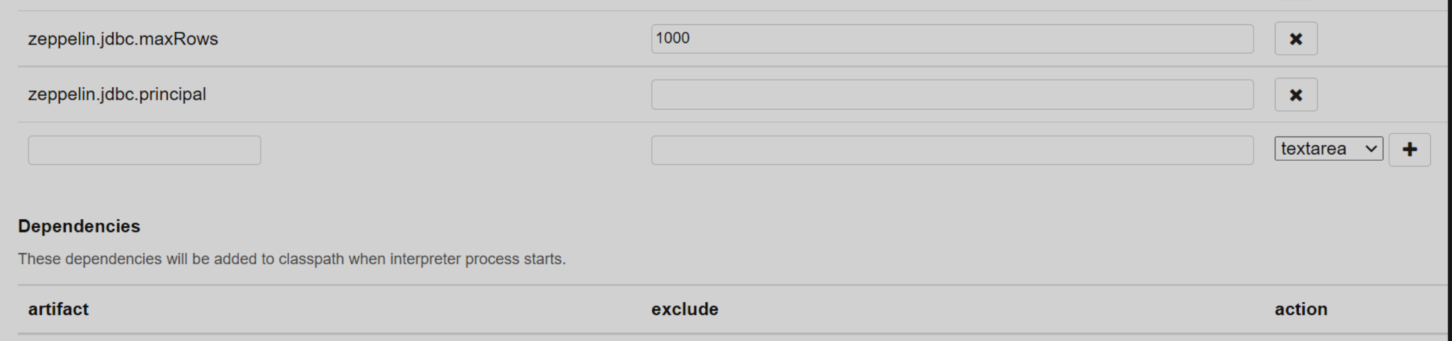
☐ Connect to existing process

☐ Set permission

Заполнение имени и группы

4. Очистите значения следующих полей и укажите свои: `default.driver`, `default.url`, `default.user`, `default.password`. Используйте таблицу из предыдущей секции.

5. Кликните ***Save***. Новый интерпретатор готов.



zeppelin.jdbc.maxConnLifetime -1

zeppelin.jdbc.maxRows 1000

zeppelin.jdbc.principal

Dependencies

These dependencies will be added to classpath when interpreter process starts.

artifact	exclude	action
groupId:artifactId:version or local file path	(Optional) comma separated groupId:artifactId list	+

Сохранение Phoenix-интерпретатора

ПРИМЕЧАНИЕ

Пример создания заметки с использованием интерпретатора, описанного в этой статье, доступен в секции Создание заметки Phoenix.

Установка стороннего интерпретатора

Константин Алпашкин

Содержание

Назад к содержанию

Шаг 1. Загрузка библиотек интерпретатора

Шаг 2. Добавление интерпретатора через веб-интерфейс

Zeppelin работает не только с community-интерпретаторами, но также позволяет установить сторонние интерпретаторы из Maven-репозиториев. В данной статье представлен пример установки интерпретатора для Apache Solr. Убедитесь, что сервис Solr доступен в вашем ADH-кластере прежде чем выполнять шаги, описанные ниже.

ПРИМЕЧАНИЕ

Больше информации об установке интерпретаторов вы можете найти в документации Zeppelin.

Шаг 1. Загрузка библиотек интерпретатора

1. Установите соединение с Zeppelin Server (например, по SSH).
2. Выполните следующую команду, чтобы загрузить библиотеки для нового интерпретатора `solr`:

```
$ sudo ZEPPELIN_INTERPRETER_DEP_MVNREPO=https://repo1.maven.org/maven2 /usr/lib/zeppelin/bin/install-interpreter.sh --name solr --
artifact com.lucidworks.zeppelin:zeppelin-solr:0.1.6
```

Результат выполнения имеет следующий вид:

```
OpenJDK 64-Bit Server VM warning: ignoring option MaxPermSize=512m; support was removed in 8.0
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/zeppelin/lib/interpreter/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.
class]
SLF4J: Found binding in [jar:file:/usr/lib/zeppelin/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Can't find interpreter list /etc/zeppelin/conf/interpreter-list
Install solr(com.lucidworks.zeppelin:zeppelin-solr:0.1.6) to /usr/lib/zeppelin/interpreter/solr ...
Interpreter solr installed under /usr/lib/zeppelin/interpreter/solr.

1. Restart Zeppelin
2. Create interpreter setting in 'Interpreter' menu on Zeppelin GUI
3. Then you can bind the interpreter on your note
```

РЕКОМЕНДАЦИЯ

Переменная окружения `ZEPPELIN_INTERPRETER_DEP_MVNREPO` позволяет избежать ошибок с разрешением зависимостей, которые могут встречаться в Zepplin 0.9.0 и более ранних версиях. Вы можете присвоить переменной `ZEPPELIN_INTERPRETER_DEP_MVNREPO` значение `https://repo1.maven.org/maven2` в конфигурационном файле */etc/zeppelin/conf/zeppelin-env.sh*, а затем перезапустить Zeppelin Server.

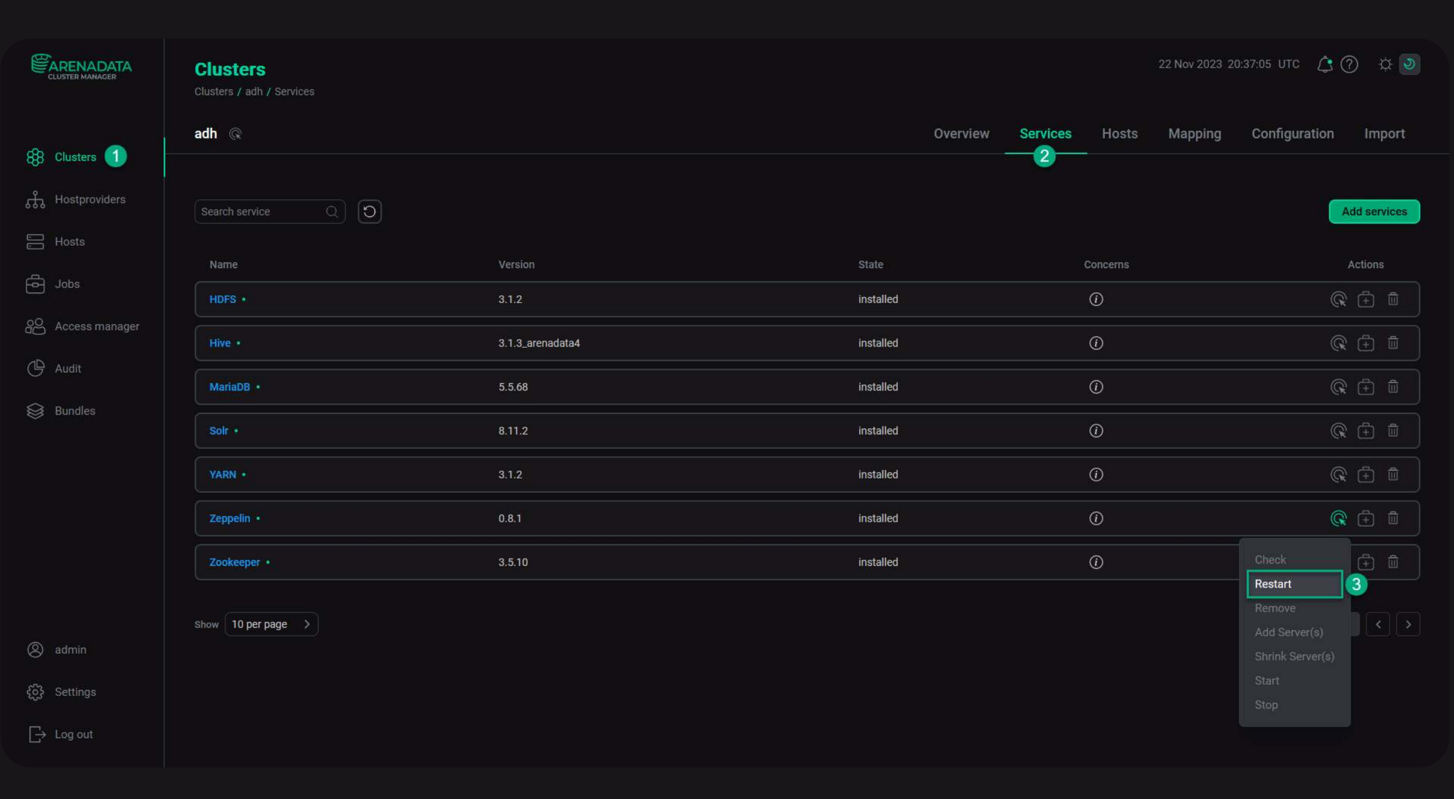
3. Убедитесь, что директория, соответствующая новому интерпретатору `solr`, добавлена:

```
$ ls -la /usr/lib/zeppelin/interpreter/solr
```

Пример вывода

```
total 39440
drwxr-xr-x.  2 root root     4096 Dec 10 09:12 .
drwxr-xr-x. 31 root root     4096 Dec 10 09:12 ..
-rw-r--r--.  1 root root    89616 Dec  6 15:58 aether-api-1.12.jar
-rw-r--r--.  1 root root    24449 Dec  6 15:58 aether-connector-file-1.12.jar
-rw-r--r--.  1 root root    29646 Dec  6 15:58 aether-connector-wagon-1.12.jar
-rw-r--r--.  1 root root   129254 Dec  6 15:58 aether-impl-1.12.jar
-rw-r--r--.  1 root root    15430 Dec  6 15:58 aether-spi-1.12.jar
-rw-r--r--.  1 root root   126629 Dec  6 15:58 aether-util-1.12.jar
-rw-r--r--.  1 root root   622849 Dec  6 15:58 bcpkix-jdk15on-1.52.jar
-rw-r--r--.  1 root root  2902942 Dec  6 15:58 bcprov-jdk15on-1.52.jar
-rw-r--r--.  1 root root    30085 Dec  6 15:58 commons-codec-1.2.jar
-rw-r--r--.  1 root root   358048 Dec  6 15:58 commons-configuration-1.9.jar
-rw-r--r--.  1 root root    54423 Dec  6 15:58 commons-exec-1.3.jar
-rw-r--r--.  1 root root   305001 Dec  6 15:58 commons-httpclient-3.1.jar
-rw-r--r--.  1 root root   208700 Dec  6 15:58 commons-io-2.5.jar
-rw-r--r--.  1 root root   279193 Dec  6 15:58 commons-lang-2.5.jar
-rw-r--r--.  1 root root    60686 Dec  6 15:58 commons-logging-1.1.1.jar
-rw-r--r--.  1 root root  2213560 Dec  6 15:58 commons-math3-3.6.1.jar
-rw-r--r--.  1 root root   109568 Dec  6 15:58 commons-pool2-2.3.jar
-rw-r--r--.  1 root root   258075 Dec  6 15:58 gson-2.8.9.jar
-rw-r--r--.  1 root root    26742 Dec  6 15:58 gson-extras-0.2.1.jar
-rw-r--r--.  1 root root  2172168 Dec  6 15:58 guava-15.0.jar
-rw-r--r--.  1 root root   747794 Dec  6 15:58 httpclient-4.5.3.jar
-rw-r--r--.  1 root root   323824 Dec  6 15:58 httpcore-4.4.6.jar
-rw-r--r--.  1 root root    41029 Dec  6 15:58 httpmime-4.5.3.jar
-rw-r--r--.  1 root root    16516 Dec  6 15:58 jcl-over-slf4j-1.7.24.jar
-rw-r--r--.  1 root root   213911 Dec  6 15:58 jline-2.12.1.jar
-rw-r--r--.  1 root root   281579 Dec  6 15:58 jsoup-1.6.1.jar
-rw-r--r--.  1 root root   227712 Dec  6 15:58 libthrift-0.9.2.jar
-rw-r--r--.  1 root root   489884 Dec  6 15:58 log4j-1.2.17.jar
-rw-r--r--.  1 root root    55639 Dec  6 15:58 maven-aether-provider-3.0.3.jar
-rw-r--r--.  1 root root    51908 Dec  6 15:58 maven-artifact-3.0.jar
-rw-r--r--.  1 root root   163439 Dec  6 15:58 maven-model-3.0.3.jar
-rw-r--r--.  1 root root   149093 Dec  6 15:58 maven-model-builder-3.0.3.jar
-rw-r--r--.  1 root root    48920 Dec  6 15:58 maven-plugin-api-3.0.jar
-rw-r--r--.  1 root root    30088 Dec  6 15:58 maven-repository-metadata-3.0.3.jar
-rw-r--r--.  1 root root    27948 Dec  6 15:58 noggit-0.8.jar
-rw-r--r--.  1 root root    47058 Dec  6 15:58 plexus-classworlds-2.4.jar
-rw-r--r--.  1 root root     4211 Dec  6 15:58 plexus-component-annotations-1.5.5.jar
-rw-r--r--.  1 root root    61072 Dec  6 15:58 plexus-interpolation-1.14.jar
-rw-r--r--.  1 root root   223943 Dec  6 15:58 plexus-utils-2.0.7.jar
-rw-r--r--.  1 root root 15612191 Dec  6 15:58 scala-compiler-2.11.12.jar
-rw-r--r--.  1 root root  5749423 Dec  6 15:58 scala-library-2.11.12.jar
-rw-r--r--.  1 root root   423753 Dec  6 15:58 scala-parser-combinators_2.11-1.0.4.jar
-rw-r--r--.  1 root root   671138 Dec  6 15:58 scala-xml_2.11-1.0.5.jar
-rw-r--r--.  1 root root   489453 Dec  6 15:58 sisu-guice-3.0.2-no_aop.jar
-rw-r--r--.  1 root root   247361 Dec  6 15:58 sisu-inject-bean-2.2.2.jar
-rw-r--r--.  1 root root   203536 Dec  6 15:58 sisu-inject-plexus-2.2.2.jar
-rw-r--r--.  1 root root    32119 Dec  6 15:58 slf4j-api-1.7.10.jar
-rw-r--r--.  1 root root     8866 Dec  6 15:58 slf4j-log4j12-1.7.10.jar
-rw-r--r--.  1 root root  1870892 Dec  6 15:58 solr-solrj-7.5.0.jar
-rw-r--r--.  1 root root   161867 Dec  6 15:58 stax2-api-3.1.4.jar
-rw-r--r--.  1 root root    10494 Dec  6 15:58 wagon-http-1.0.jar
-rw-r--r--.  1 root root    14945 Dec  6 15:58 wagon-http-lightweight-1.0.jar
-rw-r--r--.  1 root root    24599 Dec  6 15:58 wagon-http-shared-1.0.jar
-rw-r--r--.  1 root root    53186 Dec  6 15:58 wagon-provider-api-1.0.jar
-rw-r--r--.  1 root root   486013 Dec  6 15:58 woodstox-core-asl-4.4.1.jar
-rw-r--r--.  1 root root   100679 Dec  6 15:58 zeppelin-solr-0.1.6.jar
-rw-r--r--.  1 root root   887738 Dec  6 15:58 zookeeper-3.4.11.jar
```

4. Перезапустите Zeppelin Server. Вы можете сделать это, используя ADCM:
   - Откройте страницу ***Clusters***.
   - Выберите ваш кластер из списка.
   - Перейдите на вкладку ***Services***.
   - Кликните ***Restart*** в выпадающем списке сервиса ***Zeppelin***.

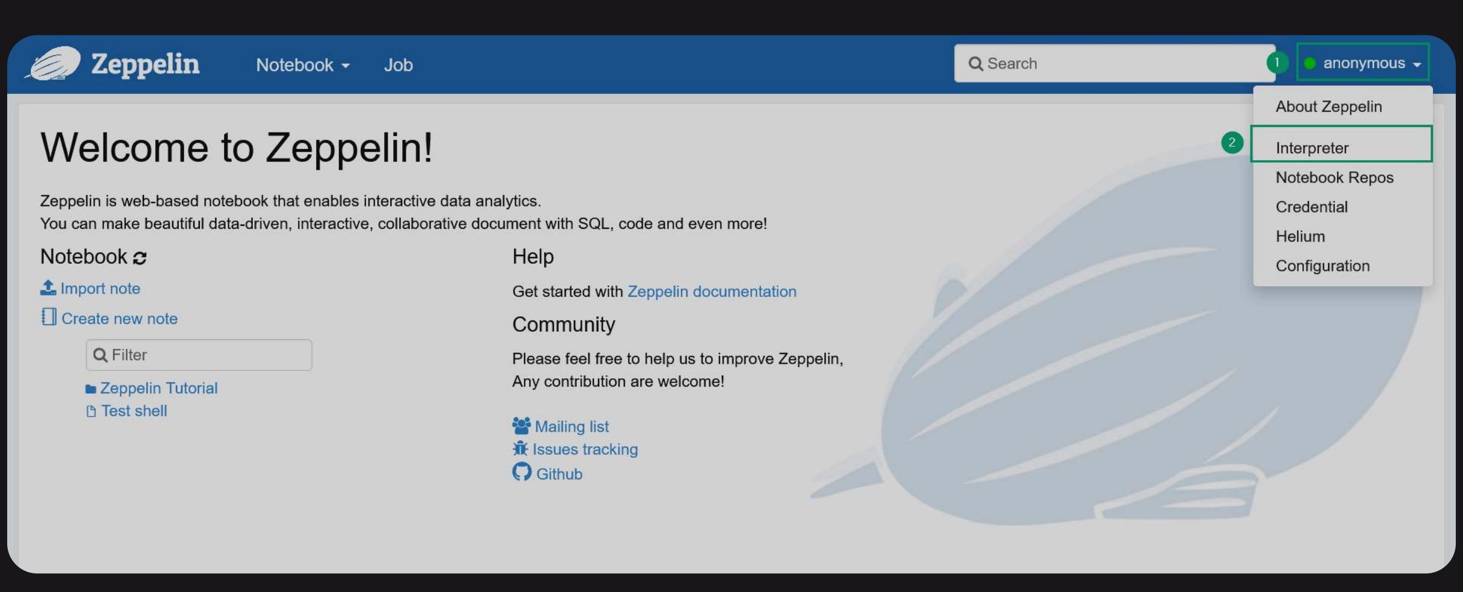


Name	Version	State
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
MariaDB	5.5.68	installed
Solr	8.11.2	installed
YARN	3.1.2	installed
Zeppelin	0.8.1	installed
Zookeeper	3.5.10	installed

Перезапуск сервиса Zepplin через ADCM

Шаг 2. Добавление интерпретатора через веб-интерфейс

1. Выберите ***Interpreter*** в меню аккаунта, расположенном в правом верхнем углу домашней страницы Zeppelin.



Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

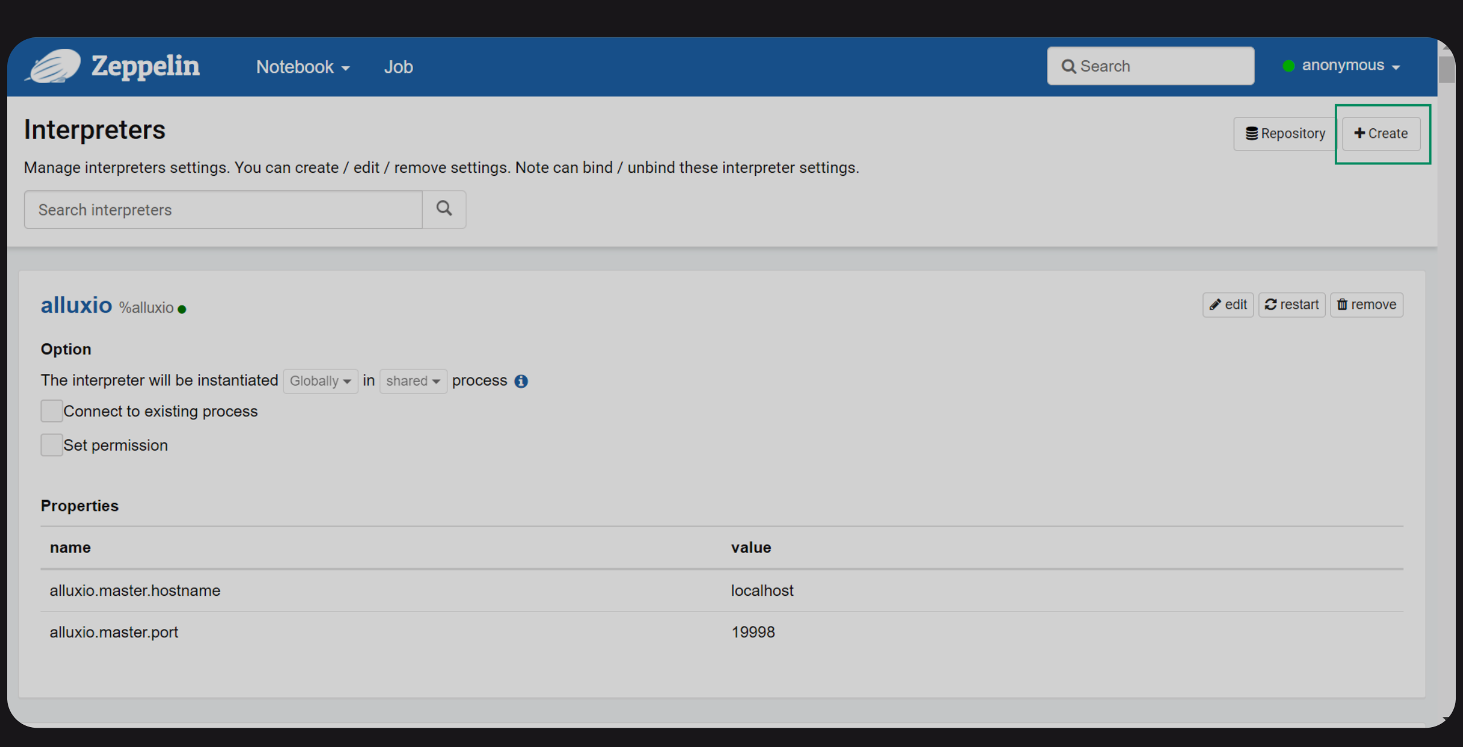
Notebook
Import note
Create new note
Filter
Zeppelin Tutorial
Test shell

Help
Get started with Zeppelin documentation

Community
Please feel free to help us to improve Zeppelin,
Any contribution are welcome!
Mailing list
Issues tracking
Github

Список интерпретаторов

2. Кликните ***Create***.



Interpreters

Manage interpreters settings. You can create / edit / remove settings. Note can bind / unbind these interpreter settings.

alluxio %alluxio

Option

The interpreter will be instantiated Globally in shared process

☐ Connect to existing process

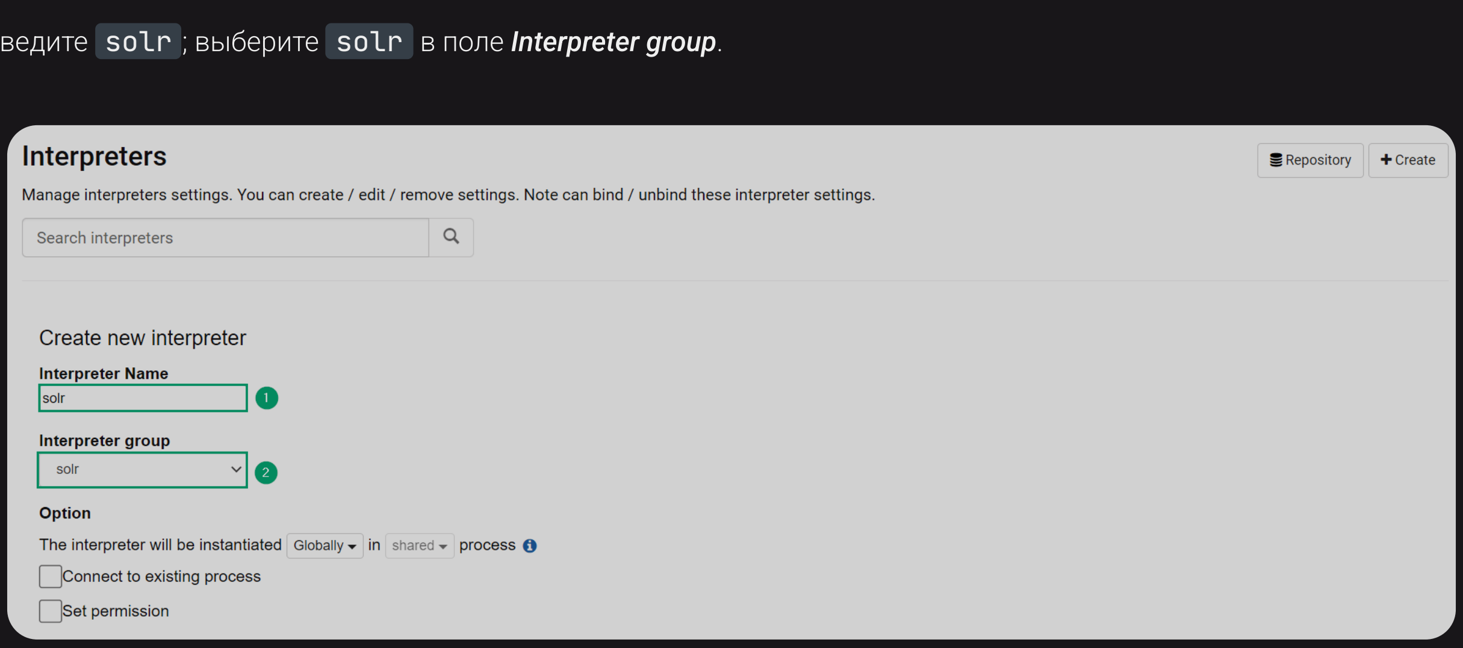
☐ Set permission

Properties

name	value
alluxio.master.hostname	localhost
alluxio.master.port	19998

Создание нового интерпретатора

3. В поле ***Interpreter Name*** введите `solr`; выберите `solr` в поле ***Interpreter group***.



Interpreters

Manage interpreters settings. You can create / edit / remove settings. Note can bind / unbind these interpreter settings.

Create new interpreter

Interpreter Name: solr

Interpreter group: solr

Option

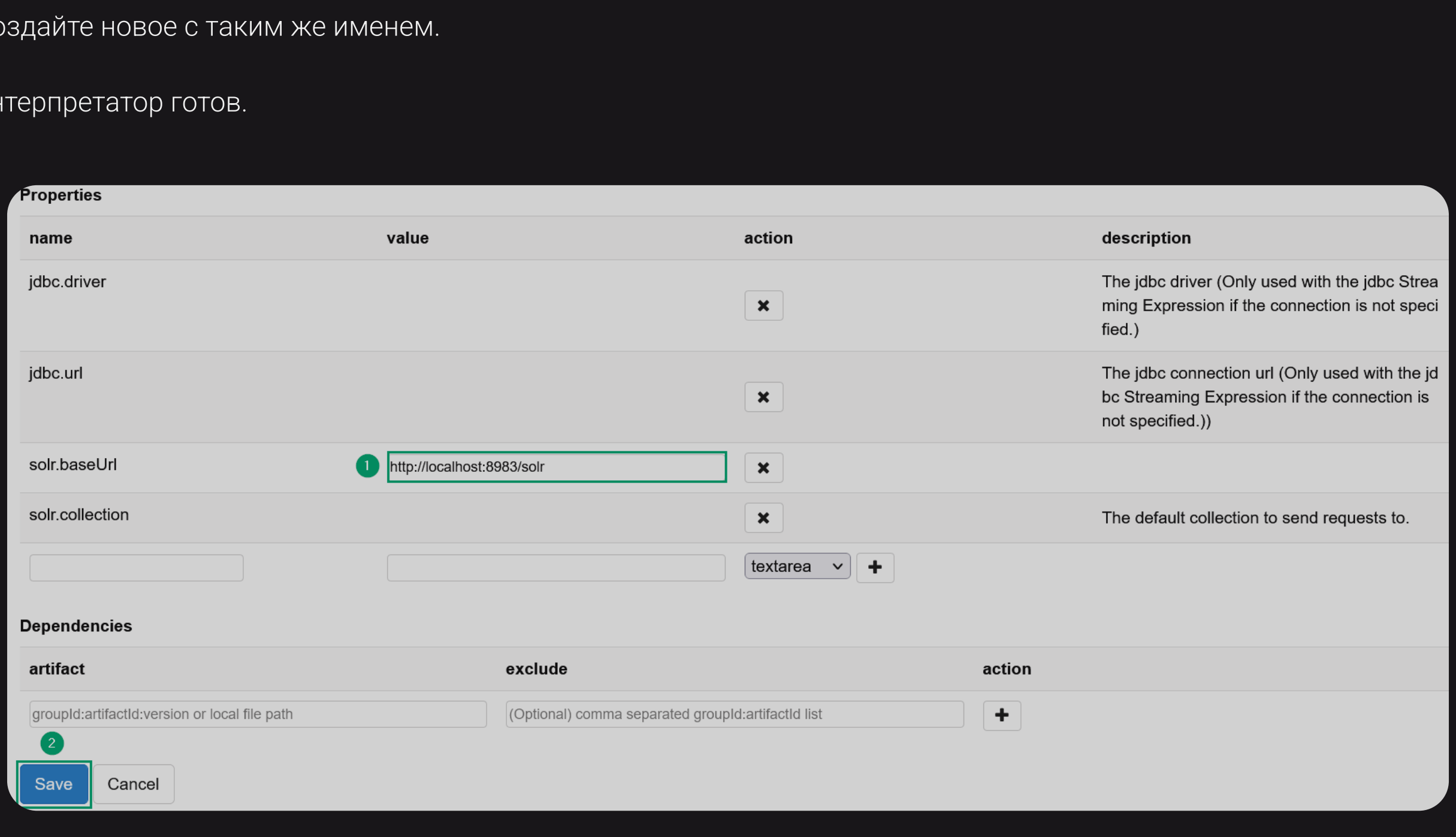
The interpreter will be instantiated Globally in shared process

☐ Connect to existing process

☐ Set permission

Заполнение имени и группы

4. Установите в поле ***solr.baseUrl*** значение, равное `http://<IP>:8983/solr`, где `<IP>` — это IP-адрес вашего Solr-сервиса. Если значение поля не редактируется, удалите данное поле и создайте новое с таким же именем.
5. Кликните ***Save***. Новый интерпретатор готов.



textarea
groupId:artifactId:version or local file path
(Optional) comma separated groupId:artifactId list
Save
Cancel
1
2

Properties

name	value	action	description
jdbc.driver		✖	The jdbc driver (Only used with the jdbc Streaming Expression if the connection is not specified.)
jdbc.url		✖	The jdbc connection url (Only used with the jdbc Streaming Expression if the connection is not specified.))
solr.baseUrl	http://localhost:8983/solr	✖	
solr.collection		✖	The default collection to send requests to.

Dependencies

artifact	exclude	action

Заполнение полей и сохранение интерпретатора

ПРИМЕЧАНИЕ

- Пример создания заметки, использующей интерпретатор, который описан в данной статье, можно увидеть на странице Создание заметки Solr.
- Вы также можете использовать базовый интерпретатор JDBC для соединения с Solr. Для этого следуйте инструкциям на странице Добавление интерпретатора в группу, а также смотрите документацию Solr.

User-managed интерпретаторы

Константин Алпашкин

Содержание

Назад к содержанию

- Обзор
- Примеры использования
  - Кейс: защита системного интерпретатора от изменений при рестарте Zeppelin
  - Кейс: обновление настроек интерпретатора после включения/отключения Kerberos/SSL/HA
  - Кейс: передача конфигурации интерпретатора в ADCM в виде файла
  - Кейс: сброс настроек для всех кастомных интерпретаторов

Обзор

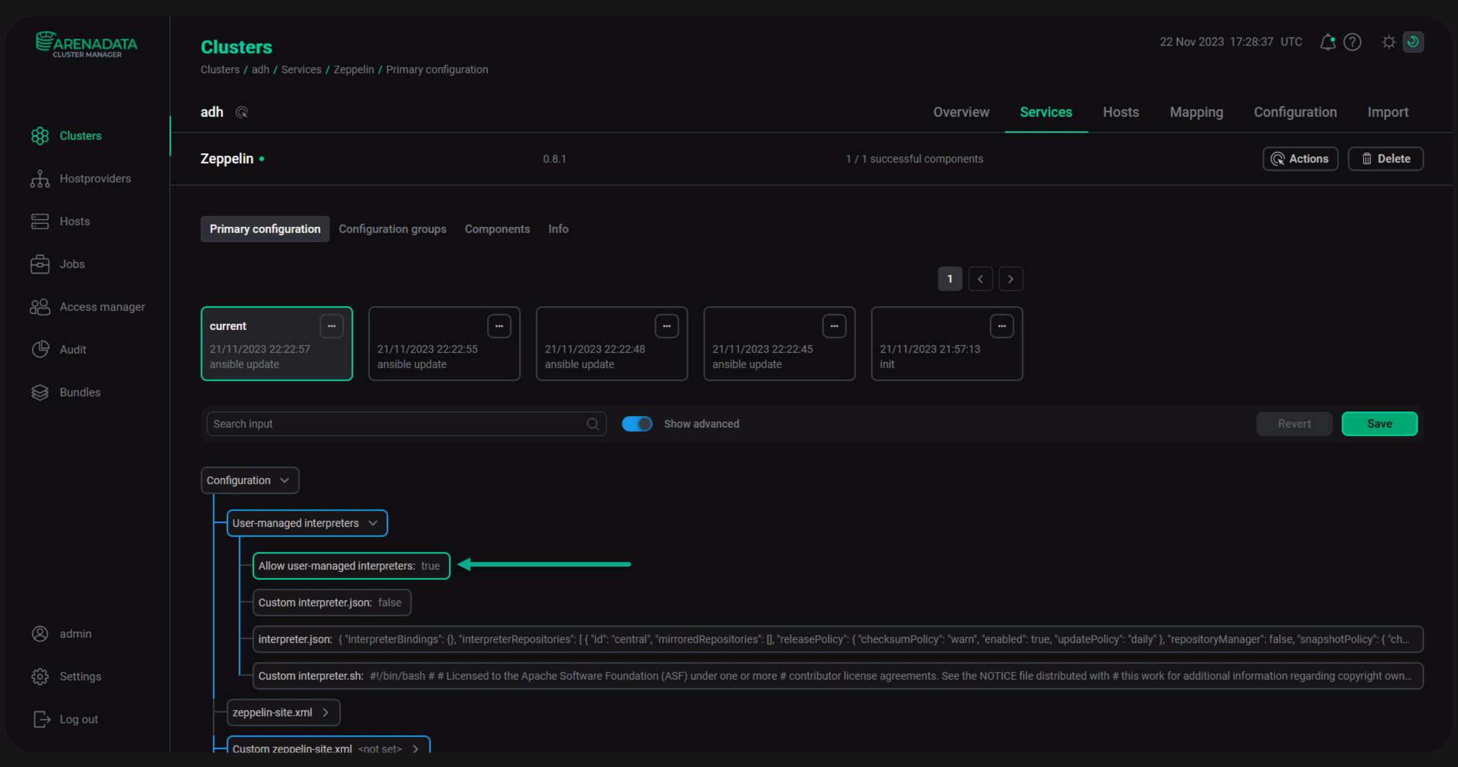
В данном разделе описаны особенности работы с кастомными Zeppelin-интерпретаторами, для которых задан параметр `user-managed=true`, а также представлены типичные сценарии их использования.

> По умолчанию кастомные параметры интерпретаторов, заданные пользователем, теряются, если рестарт Zeppelin-сервиса выполняется с помощью ADCM. Чтобы сохранить кастомные свойства при рестарте Zeppelin, должны быть выполнено следующее:
>
> - Опция ***Allow user-managed interpreters*** на странице конфигурации Zeppelin-сервиса в ADCM включена (значение по умолчанию).
> - У интерпретаторов, чьи свойства необходимо сохранять, должно быть объявлено свойство `user-managed=true`. Сделать это можно в Zeppelin UI.
>
> Подробности использования этих настроек описаны далее в статье.

Интерпретаторы Zeppelin делятся на 2 категории:

- **Системные**. Это те интерпретаторы, которые включены в бандл ADH и доступны в интерфейсе ADCM по умолчанию.
- **Кастомные**. Это интерпретаторы, добавленные пользователями вручную.

В веб-интерфейсе Zeppelin для каждого интерпретатора можно добавить кастомные свойства типа ключ/значение для настройки поведения интерпретатора. Однако при рестарте Zeppelin-сервиса с помощью ADCM такие свойства теряются. Чтобы кастомные свойства сохранялись после рестарта, используйте опцию ***Allow user-managed interpreters*** в ADCM, как изображено на рисунке.


ARENADATA CLUSTER MANAGER
Clusters
Clusters / adh / Services / Zeppelin / Primary configuration
22 Nov 2023 17:28:37 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
adh
Overview
Services
Hosts
Mapping
Configuration
Import
Zeppelin
0.8.1
1 / 1 successful components
Actions
Delete
Primary configuration
Configuration groups
Components
Info
current 21/11/2023 22:22:57 ansible update
21/11/2023 22:22:55 ansible update
21/11/2023 22:22:48 ansible update
21/11/2023 22:22:45 ansible update
21/11/2023 21:57:13 init
Search input
Show advanced
Revert
Save
Configuration
User-managed interpreters
zeppelin-site.xml
Custom zeppelin-site.xml <not set>

Allow user-managed interpreters: true

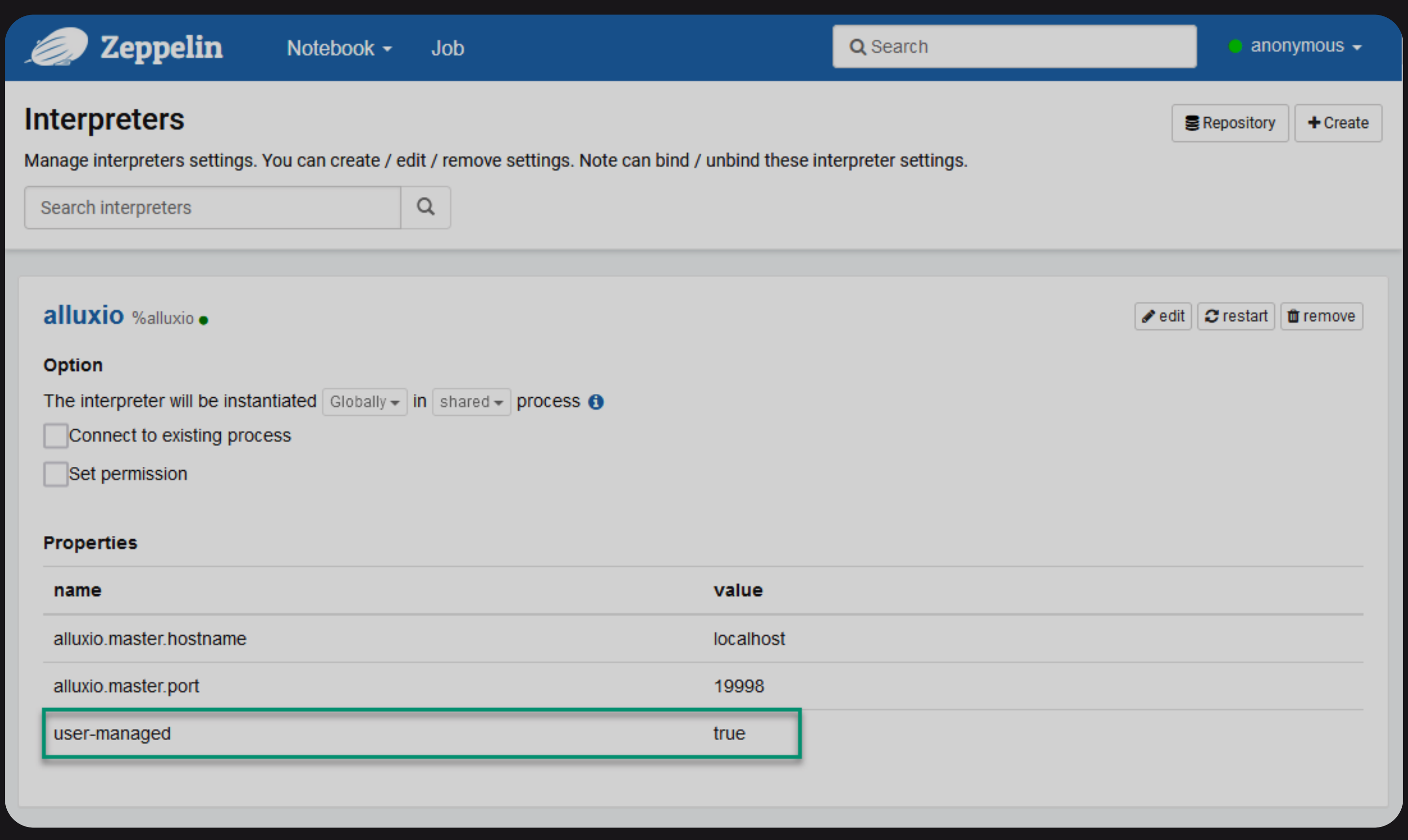
Custom interpreter.json: false

interpreter.json: { "interpreterBindings": {}, "interpreterRepositories": [{ "id": "central", "mirroredRepositories": [], "releasePolicy": { "checksumPolicy": "warn", "enabled": true, "updatePolicy": "daily" }, "repositoryManager": false, "snapshotPolicy": { "ch...

Custom interpreter.sh: #!/bin/bash # # Licensed to the Apache Software Foundation (ASF) under one or more # contributor license agreements. See the NOTICE file distributed with # this work for additional information regarding copyright own...

Опция Allow user-managed interpreters

Если опция ***Allow user-managed interpreters*** включена, при рестарте Zeppelin-сервиса ADCM подтянет с Zeppelin-хоста конфигурации всех интерпретаторов, у которых есть свойство `user-managed=true`.


Zeppelin
Notebook
Job
Search
anonymous
Repository
Create
Search interpreters
edit
restart
remove

Interpreters

Manage interpreters settings. You can create / edit / remove settings. Note can bind / unbind these interpreter settings.

alluxio %alluxio

Option

The interpreter will be instantiated Globally in shared process

☐ Connect to existing process

☐ Set permission

Properties

name	value
alluxio.master.hostname	localhost
alluxio.master.port	19998
user-managed	true

Опция user-managed в Zeppelin UI

Если отключить ***Allow user-managed interpreters***, при рестарте Zeppelin-сервиса ADCM будет создавать дефолтный список системных интерпретаторов.

Примеры использования

Ниже приведены примеры решения типовых задач с использованием user-managed интерпретаторов.

Кейс: защита системного интерпретатора от изменений при рестарте Zeppelin

Если вы не хотите, чтобы настройки интерпретатора изменялись при рестарте Zeppelin из ADCM, установите свойство `user-managed=true` для нужного интерпретатора в веб-интерфейсе Zeppelin. Если опция ***Allow user-managed interpreters*** включена в ADCM, настройки кастомных интерпретаторов не будут изменяться после рестарта Zeppelin.

Кейс: обновление настроек интерпретатора после включения/отключения Kerberos/SSL/HA

Допустим, вы только что включили/отключили Kerberos/SSL/HA для ADH-сервиса. Обычно подобные операции приводят к изменению JDBC-строки подключения, поэтому если такая JDBC-строка была задана на интерпретаторе в виде пользовательского свойства, его необходимо обновить. Чтобы быстро обновить конфигурацию интерпретатора:

1. Отключите опцию ***Allow user-managed interpreters*** на странице конфигурации Zeppelin-сервиса в ADCM либо установите `user-managed=false` (в Zeppelin UI) для тех интерпретаторов, чьи параметры необходимо обновить.
2. Выполните рестарт Zeppelin.

После рестарта Zeppelin-сервиса ADCM сгенерирует список стандартных интерпретаторов по шаблону, используя корректное значение JDBC-строки.

Кейс: передача конфигурации интерпретатора в ADCM в виде файла

Вы можете определить содержимое файлов *interpreter.json*/*interpreter.sh* через ADCM. Для этого вставьте содержимое файлов на странице настройки Zeppelin-сервиса и сохраните конфигурацию. После рестарта Zeppelin инициализирует интерпретаторы, используя кастомные *interpreter.json*/*interpreter.sh*.

Кейс: сброс настроек для всех кастомных интерпретаторов

Чтобы сбросить настройки всех интерпретаторов, выполните следующее:

1. Отключите опцию ***Allow user-managed interpreters*** (по умолчанию включена) на странице конфигурации Zeppelin-сервиса в ADCM.
2. Отключите опции ***Custom interpreter.json***/***interpreter.sh***.
3. Выполните рестарт Zeppelin.

После этого при рестарте Zeppelin-сервиса все интерпретаторы, у которых были объявлены кастомные свойства, будут созданы заново с настройками по умолчанию.

Создание нового интерпретатора в Zeppelin

Константин Алпашкин

Содержание

Назад к содержанию

Обзор сценария

Шаг 1. Клонирование проекта Zeppelin

Шаг 2. Создание модуля интерпретатора

Шаг 3. Сборка нового интерпретатора

Шаг 4. Импорт JAR интерпретатора в Zeppelin

В данной статье описан процесс создания Zeppelin-интерпретатора "с нуля". Информация о добавлении интерпретаторов Zeppelin в существующую группу доступна в статье Добавление интерпретатора в группу.

Обзор сценария

Для создания нового интерпретатора необходим класс, наследующий абстрактный класс Interpreter. Данный класс в виде JAR необходимо поместить на хост ADH с компонентом Zeppelin Server. Разработка полноценного интерпретатора, способного корректно обрабатывать пользовательский код из заметок, подразумевает сложную логику и тестирование, что выходит за рамки данной статьи. В приведенном ниже примере представлена базовая реализация интерпретатора с методами-заглушками для демонстрации подключения и работоспособности тестового интерпретатора. Шаги в сценарии подходят для интерпретаторов любой сложности и являются лишь одним из возможных подходов к созданию интерпретатора.

Сценарий, описанный в статье, включает следующие шаги:

1. Клонирование проекта Zeppelin.
2. Создание интерпретатора на базе существующего модуля.
3. Реализация класса `org.apache.zeppelin.interpreter.Interpreter`, упаковка в JAR.
4. Добавление интерпретатора на хост ADH.

Шаг 1. Клонирование проекта Zeppelin

В этом примере предполагается использование проекта Zeppelin в качестве базы для создания нового интерпретатора. Такой подход облегчает управление зависимостями и структурой.

1. Склонируйте репозиторий Zeppelin.

```
$ git clone https://github.com/apache/zeppelin/tree/master
```

2. Используйте ветку, соответствующую сборке Zeppelin-сервиса. Например:

```
$ cd zeppelin
$ git checkout 0.11.2
```

Шаг 2. Создание модуля интерпретатора

В этом примере новый интерпретатор создается путем копирования существующего модуля интерпретатора и изменения его содержимого.

1. Скопируйте существующий модуль интерпретатора, например `file`.

```
$ cp -r file ./mycustominterpreter
```

2. В новом модуле удалите все классы и создайте тестовый Java-класс `CustomInterpreter`, наследующий `org.apache.zeppelin.interpreter.Interpreter`. В данном сценарии `CustomInterpreter` представляет собой класс-заглушку, который не парсит пользовательский код из заметок, однако создает записи в логе при вызове методов.

Пример CustomInterpreter.java

```
package io.arenadata.zeppelin.mycustominterpreter;

import org.apache.zeppelin.interpreter.Interpreter;
import org.apache.zeppelin.interpreter.InterpreterContext;
import org.apache.zeppelin.interpreter.InterpreterException;
import org.apache.zeppelin.interpreter.InterpreterResult;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import java.util.Properties;

public class CustomInterpreter extends Interpreter {
    public CustomInterpreter(Properties properties) {
        super(properties);
        Logger logger = LoggerFactory.getLogger(CustomInterpreter.class);
    }

    @Override
    public void open() throws InterpreterException {
        logger.info("Stub method called: open()");
    }

    @Override
    public void close() throws InterpreterException {
        logger.info("Stub method called: close()");
    }

    @Override
    public InterpreterResult interpret(String st, InterpreterContext context) throws InterpreterException {
        logger.info("Stub method called: interpret()");
        return null;
    }

    @Override
    public void cancel(InterpreterContext context) throws InterpreterException {
        logger.info("Stub method called: cancel()");
    }

    @Override
    public FormType getFormType() throws InterpreterException {
        logger.info("Stub method called: getFormType()");
        return null;
    }

    @Override
    public int getProgress(InterpreterContext context) throws InterpreterException {
        logger.info("Stub method called: getProgress()");
        return 0;
    }
}
```

РЕКОМЕНДАЦИЯ

Код интерпретаторов, доступных в Zeppelin "из коробки", можно найти в GitHub-репозитории.

Теперь проект имеет следующую структуру:

```
zeppelin/
+---alluxio
+---angular
+---bigquery
+---bin
+---build-tools
+---cassandra
+---conf
+---custominterpreter
|   |   pom.xml
|   |
|   +---src
|   |   +---main
|   |   |   +---java
|   |   |   |   \---io
|   |   |   |       \---arenadata
|   |   |   |           \---zeppelin
|   |   |   |               \---mycustominterpreter
|   |   |   |                       CustomInterpreter.java <-- Interpreter logic here
|   |   |   |
|   |   |   \---resources
|   |   |           interpreter-setting.json
|   |   |
|   |   \---test
|   |
|   \---target
+---flink
+---flink-cmd
+---groovy
...
// other modules
```

3. Так как модуль был полностью скопированной директорией, необходимо немного рефакторинга, а именно:

   - Добавить новый интерпретатор в список модулей в родительском POM (*zeppelin/pom.xml*). Например:

```
<module>mycustominterpreter</module>
```

   - Обновить поля `artifactId` и `interpreter.name` в POM-файле нового модуля (*zeppelin/mycustominterpreter/pom.xml*).
   - В *zeppelin/mycustominterpreter/src/main/resources/interpreter-setting.json* в поле `className` указать имя нового класса интерпретатора (`CustomInterpreter`). Также необходимо задать новые значения для `name` и `group`. Они будут использоваться для вызова интерпретатора в коде заметок, используя `%<interpreter.group>.<interpreter.name>`. Также в этом файле можно задать список параметров интерпретатора, которые будут доступны в Zeppelin UI. Подробная информация о таких параметрах, как `editOnDblClick` и `completionSupport` доступна в документации Zeppelin.
   - Исправить имена пакетов.
   - Удалить тест-классы.

Шаг 3. Сборка нового интерпретатора

Соберите проект Zeppelin с помощью Maven:

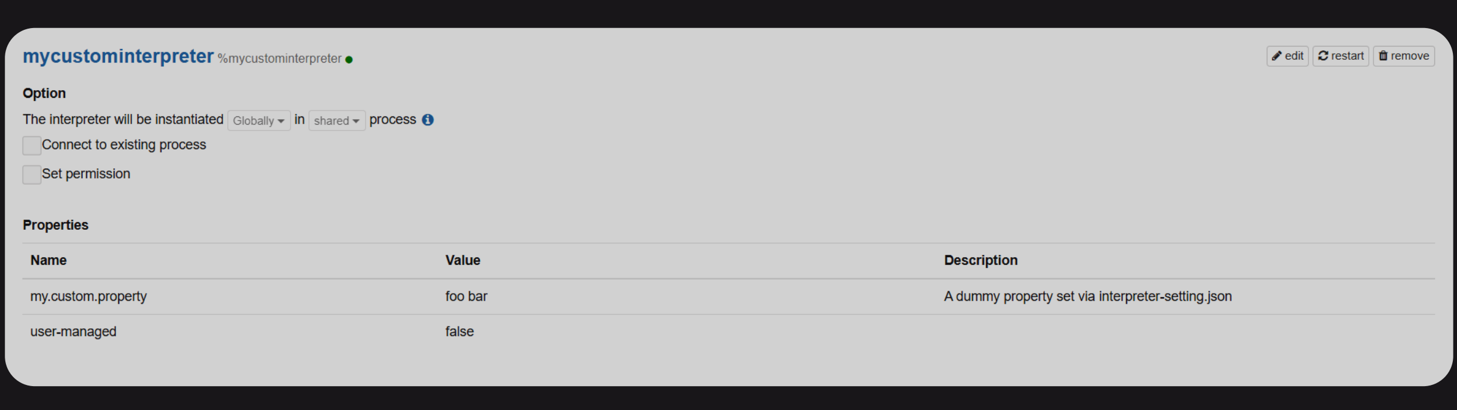
```
$ mvn clean package -DskipTests
```

В директории *target* нового модуля будет доступен JAR интерпретатора. Необходим Uber-JAR, включающий все зависимости и имеющий следующую структуру:

```
mycustominterpreter.jar
|   interpreter-setting.json
|
+---com
|   \---google
|       \--- ...
+---javassist
|   \--- ...
+---javax
|   \--- ...
+---jersey
|   \--- ...
+---META-INF
|   \--- ...
+---io
        +---arenadata
                \---zeppelin
                        \---mycustominterpreter
                                \---CustomInterpreter.class
    +---aopalliance
    +---apache
    +---slf4j
...
```

Шаг 4. Импорт JAR интерпретатора в Zeppelin

1. Скопируйте JAR интерпретатора в директорию */usr/lib/zeppelin/interpreter/* на хосте ADH, где установлен компонент Zeppelin Server.
2. Перезапустите сервис Zeppelin. После рестарта Zeppelin новый интерпретатор станет доступен в списке интерпретаторов в UI.



mycustominterpreter %mycustominterpreter

Option

The interpreter will be instantiated Globally in shared process

☐ Connect to existing process

☐ Set permission

Properties

Name	Value	Description
my.custom.property	foo bar	A dummy property set via interpreter-setting.json
user-managed	false	

Новый интерпретатор

3. Создайте тестовую заметку с новым интерпретатором. При создании заметки выберите новый интерпретатор по умолчанию либо используйте `%mycustominterpreter` в каждом параграфе. В данном сценарии отправка любых данных интерпретатору не вернет результатов, но создаст записи в логах. Используйте логи в */var/log/zeppelin/*, чтобы проверить активность тестового интерпретатора, например:

```
$ grep -r "Stub method called:*" /var/log/zeppelin
```

Примеры создания заметок в Zeppelin

Константин Алпашкин

Назад к содержанию

В данном разделе рассмотрены примеры создания заметок, которые основаны на следующих интерпретаторах:

- Phoenix
- Solr
- Spark
- Python

РЕКОМЕНДАЦИЯ

Пример с Phoenix — наиболее детальный и рекомендуется к рассмотрению первым. Другие полезные примеры вы можете найти в разделе Zeppelin Tutorial интерфейса Zeppelin.

Создание заметки Phoenix

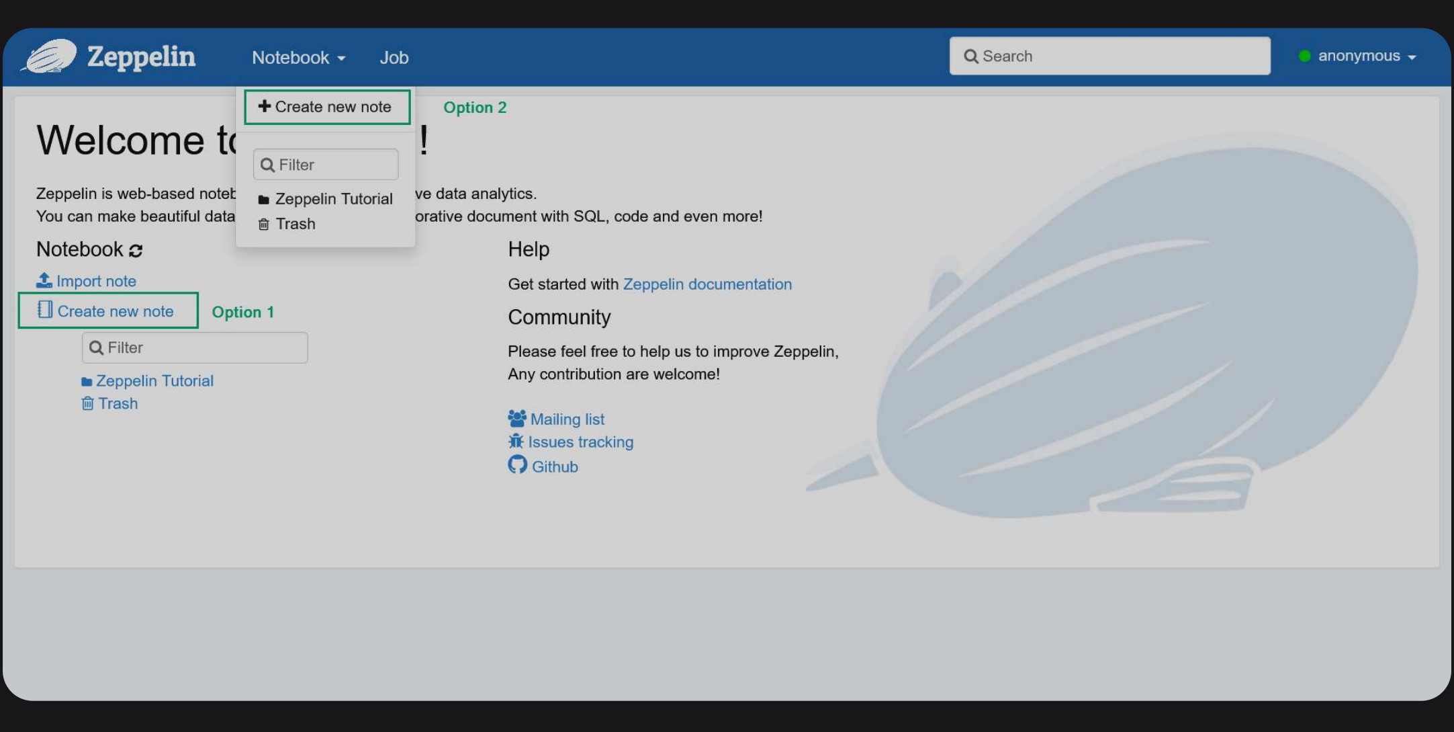
Константин Алпашкин

Назад к содержанию

ПРИМЕЧАНИЕ

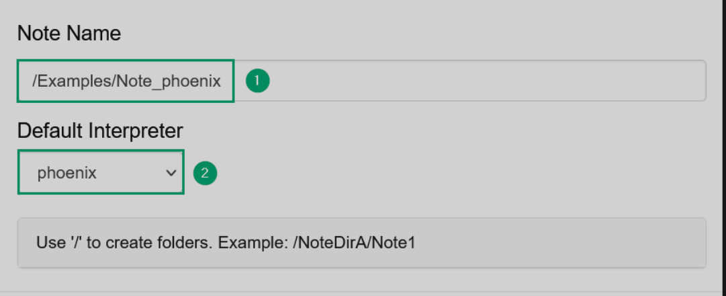
Перед созданием заметки убедитесь, что интерпретатор Phoenix настроен, как описано в статье Добавление интерпретатора в группу.

1. В веб-интерфейсе Zeppelin кликните ***Create new note*** или выберите пункт основного меню ***Notebook*** → ***Create new note***.


Zeppelin
Notebook
Job
Search
anonymous
Create new note
Option 2
Filter
Zeppelin Tutorial
Trash
Welcome t
!
Zeppelin is web-based noteb
ve data analytics.
You can make beautiful data
orative document with SQL, code and even more!
Notebook
Import note
Create new note
Option 1
Filter
Zeppelin Tutorial
Trash
Help
Get started with Zeppelin documentation
Community
Please feel free to help us to improve Zeppelin,
Any contribution are welcome!
Mailing list
Issues tracking
Github

Добавление новой заметки

2. В открывшемся диалоге введите имя и укажите интерпретатор по умолчанию, который будет использован в случаях, когда явно не указан тип интерпретатора. Обратите внимание, что вы можете создавать папки для ваших заметок. Затем кликните ***Create***.


Create New Note

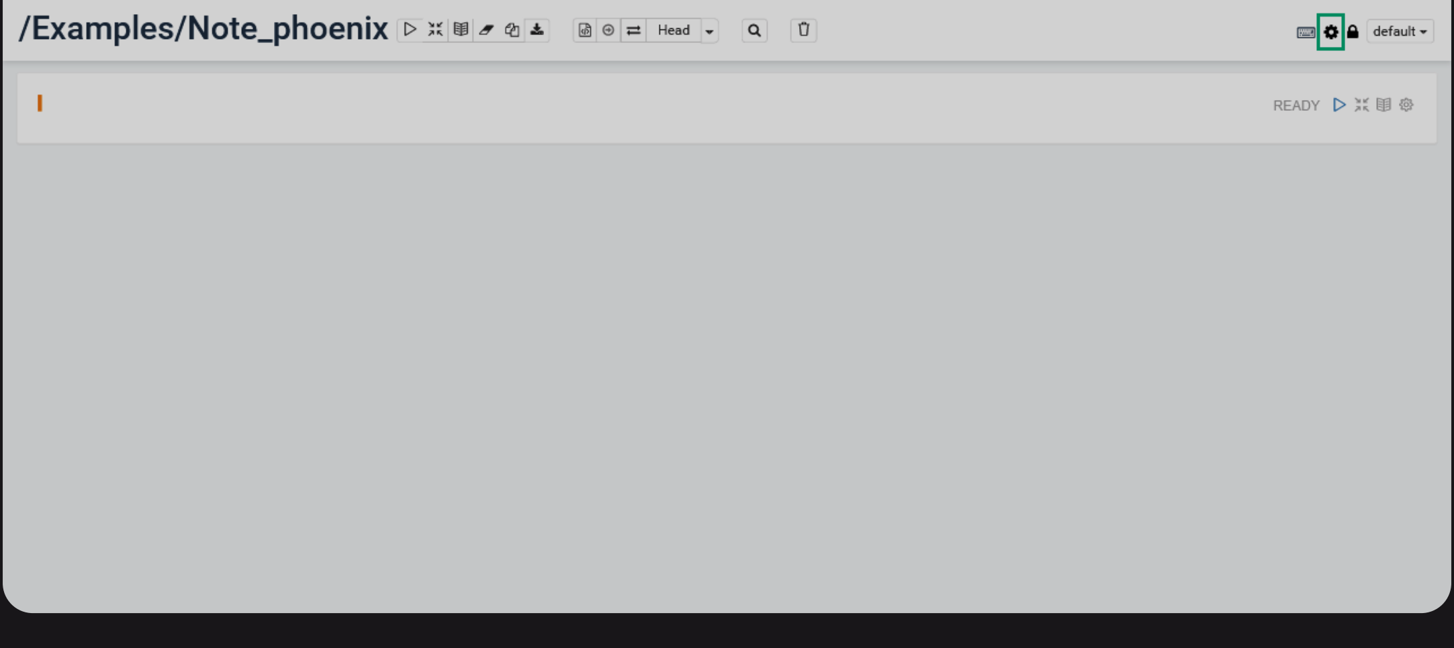
Note Name: /Examples/Note_phoenix

Default Interpreter: phoenix

Use '/' to create folders. Example: /NoteDirA/Note1

Описание новой заметки

3. В открывшемся окне кликните ***Interpreter binding***, чтобы увидеть доступные интерпретаторы.

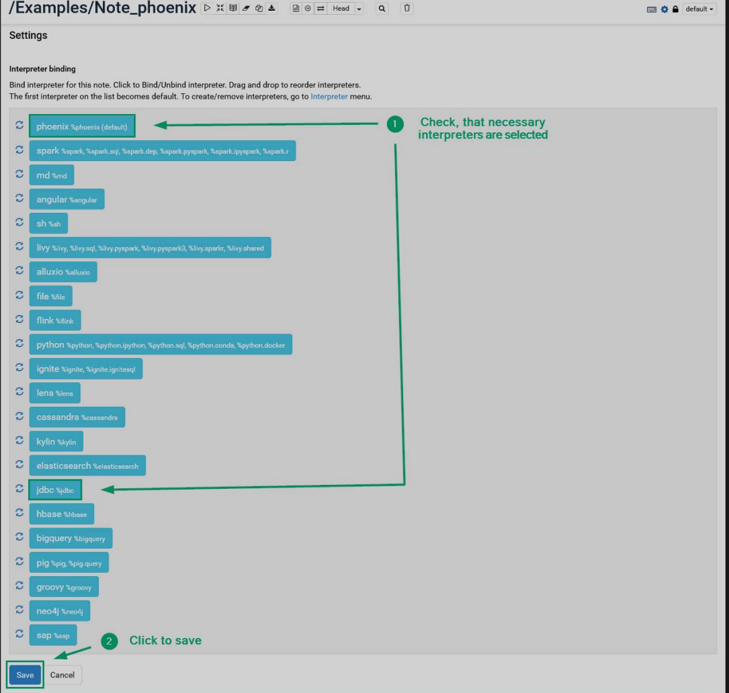

Zeppelin
Notebook
Job
Search
anonymous
/Examples/Note_phoenix
Head
default
READY

Переход к сопоставлению интерпретаторов

4. Убедитесь, что интерпретаторы `%jdbc` и `%phoenix` выбраны в открывшемся списке; в противном случае — выберите их вручную. Затем кликните ***Save***.

ПРИМЕЧАНИЕ

Вы можете использовать только один интерпретатор, однако данный раздел описывает работу с обоими интерпретаторами.


Settings

Interpreter binding

Bind interpreter for this note. Click to Bind/Unbind interpreter. Drag and drop to reorder interpreters.
The first interpreter on the list becomes default. To create/remove interpreters, go to Interpreter menu.

/Examples/Note_phoenix
phoenix %phoenix (default)
spark
md %md
angular %angular
sh %sh
livy
alluxio %alluxio
file %file
flink %flink
python
ignite
lens %lens
cassandra %cassandra
kylin %kylin
elasticsearch %elasticsearch
jdbc %jdbc
hbase %hbase
bigquery %bigquery
pig %pig, %pig.query
groovy %groovy
neo4j %neo4j
sap %sap
1 Check, that necessary interpreters are selected
2 Click to save
Save
Cancel
READY

Сопоставление интерпретаторов

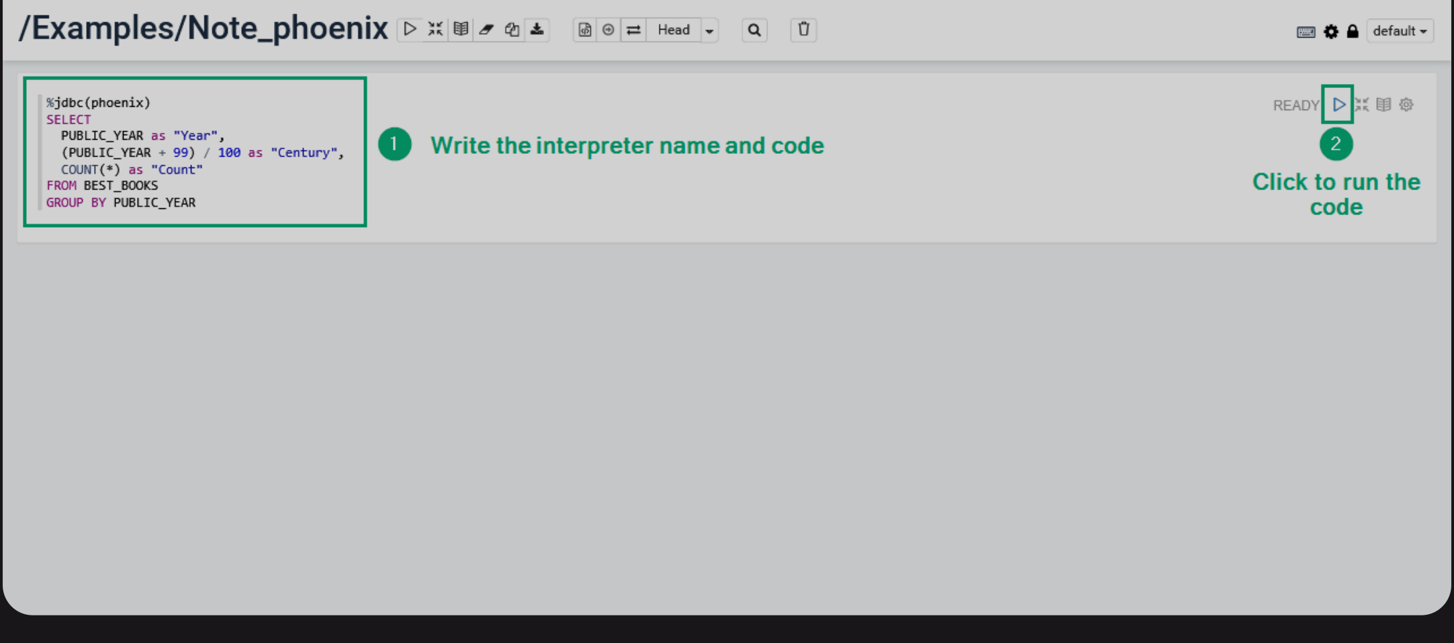
5. Для начала используйте интерпретатор %jdbc с настройками, отмеченными префиксом `phoenix_`. Введите следующий код в первом параграфе вашей заметки. Данный запрос возвращает количество книг, сгруппированных по дате их первой публикации:

```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

ПРИМЕЧАНИЕ

Запрос использует таблицу `BEST_BOOKS`. Создание этой таблицы описано в статье Массовая загрузка в Phoenix. Если вы не создавали эту таблицу, вы можете выполнить любой другой запрос к вашей Phoenix-таблице.

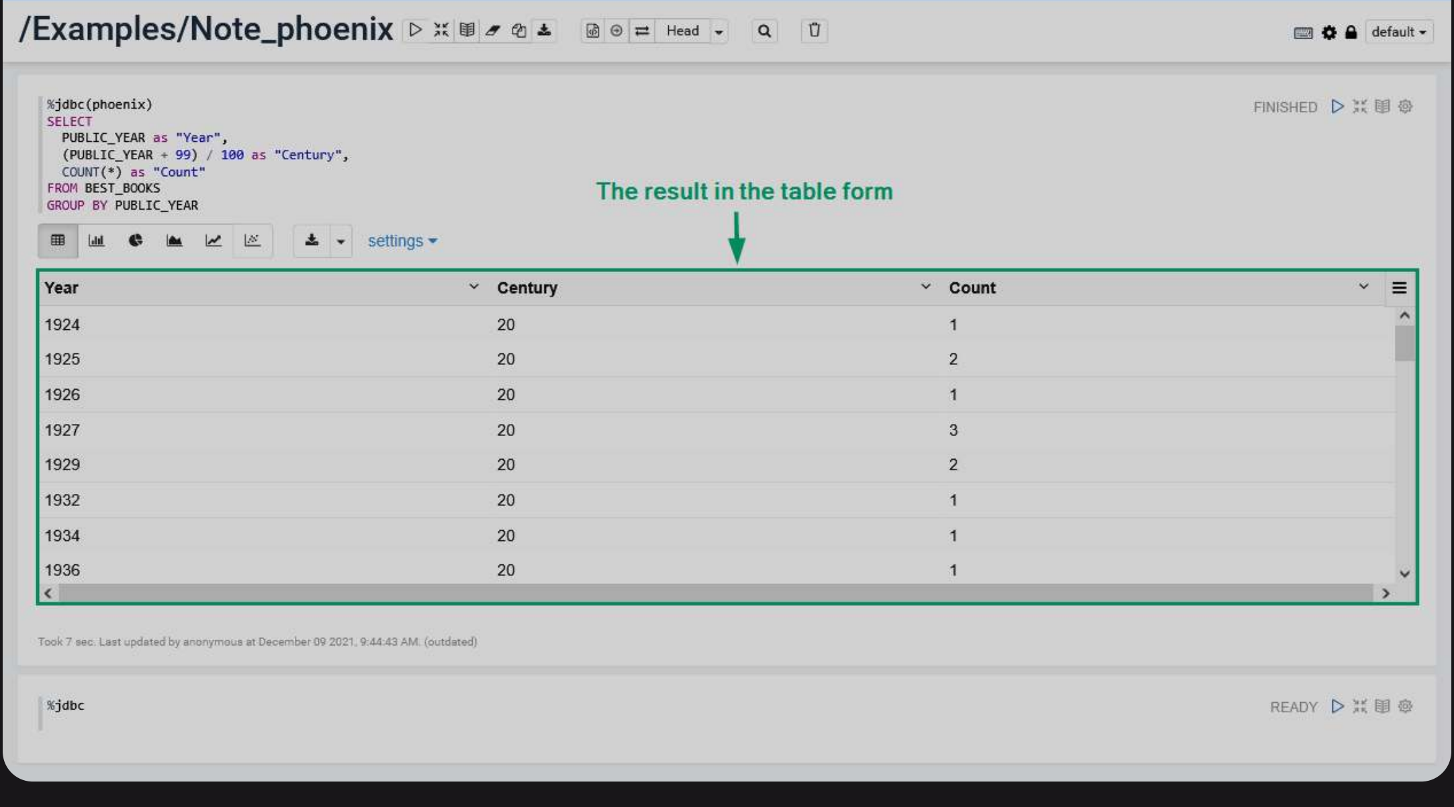
6. Чтобы выполнить запрос, кликните ***Run this paragraph***.


```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

Zeppelin
Notebook
Job
Search
anonymous
/Examples/Note_phoenix
Head
default
1 Write the interpreter name and code
READY
2 Click to run the code

Выполнение кода

7. Результаты запроса доступны в виде таблицы.


```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

Year	Century	Count
1924	20	1
1925	20	2
1926	20	1
1927	20	3
1929	20	2
1932	20	1
1934	20	1
1936	20	1

Took 7 sec. Last updated by anonymous at December 09 2021, 9:44:43 AM. (outdated)

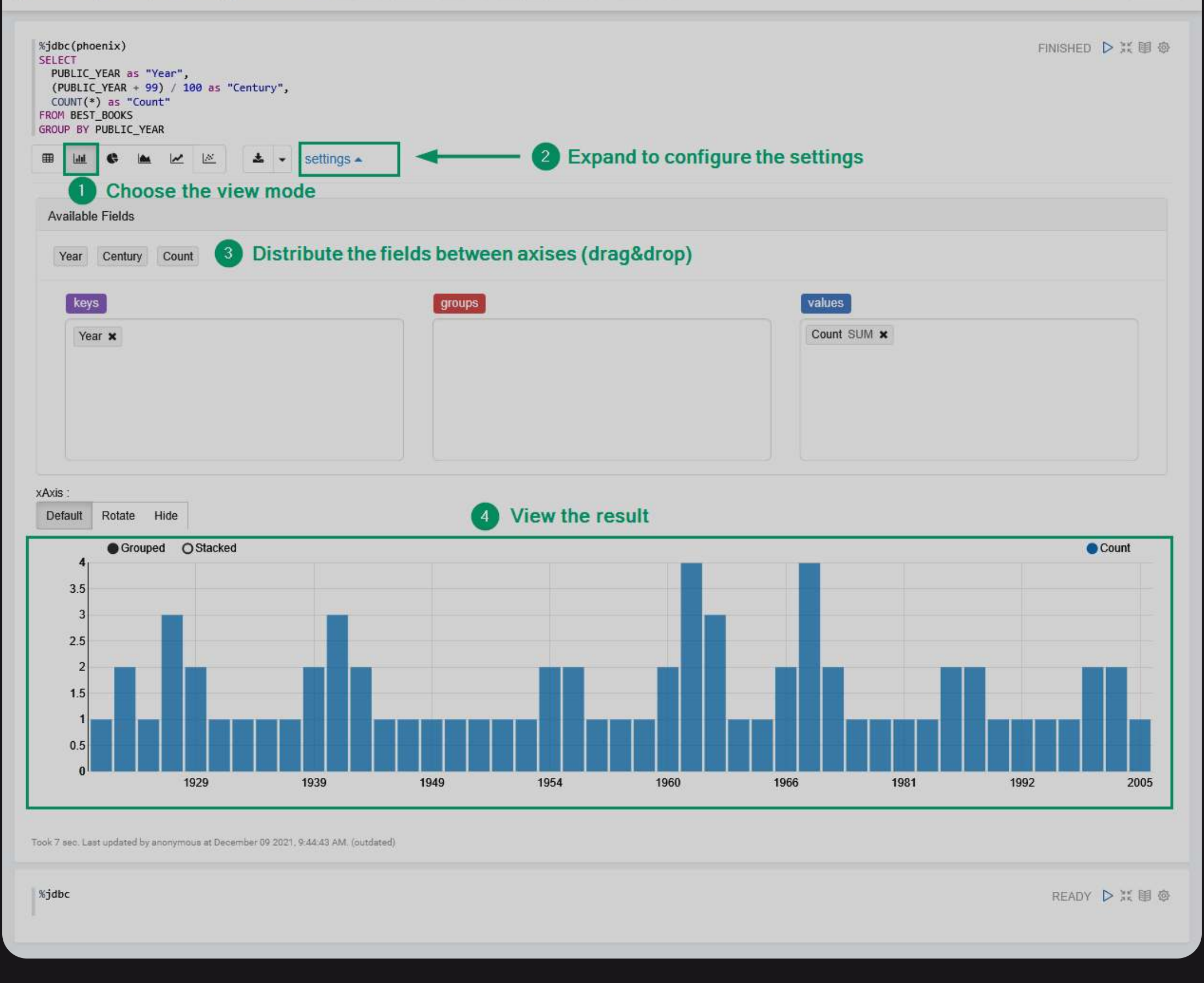
```
%jdbc
```

Zeppelin
Notebook
Job
Search
anonymous
/Examples/Note_phoenix
Head
default
FINISHED
settings
The result in the table form
READY

Таблица с результатами

РЕКОМЕНДАЦИЯ

Вы можете использовать другие способы визуализации результатов. Для этого кликните по иконке, которая соответствуют текущей визуализации, разверните панель ***settings*** и распределите поля результатов вашего запроса, используя drag-and-drop. Ниже представлены доступные формы визуализации.


```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

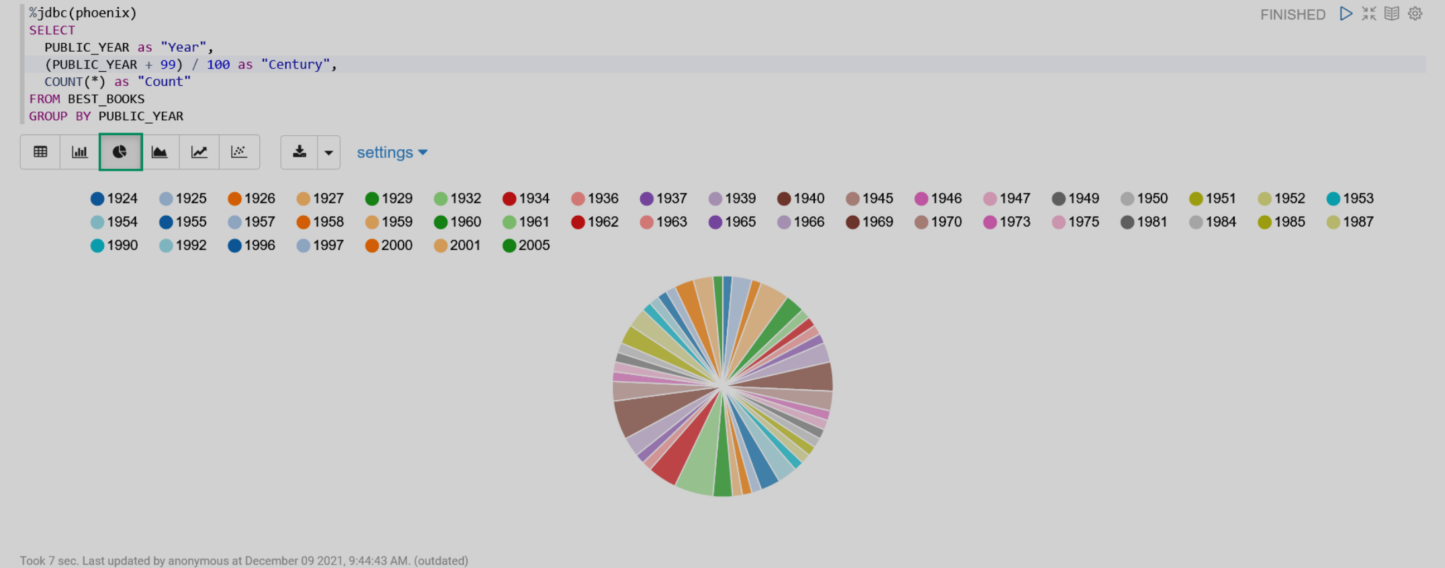
Took 7 sec. Last updated by anonymous at December 09 2021, 9:44:43 AM. (outdated)

```
%jdbc
```

Zeppelin
Notebook
Job
Search
anonymous
/Examples/Note_phoenix
Head
default
FINISHED
settings
2 Expand to configure the settings
1 Choose the view mode
Available Fields
Year
Century
Count
3 Distribute the fields between axises (drag&drop)
keys
Year
groups
values
Count SUM
xAxis :
Default
Rotate
Hide
4 View the result
Grouped
Stacked
Count
4
3.5
3
2.5
2
1.5
1
0.5
0
1929
1939
1949
1954
1960
1966
1981
1992
2005
READY

Столбчатая диаграмма с результатами

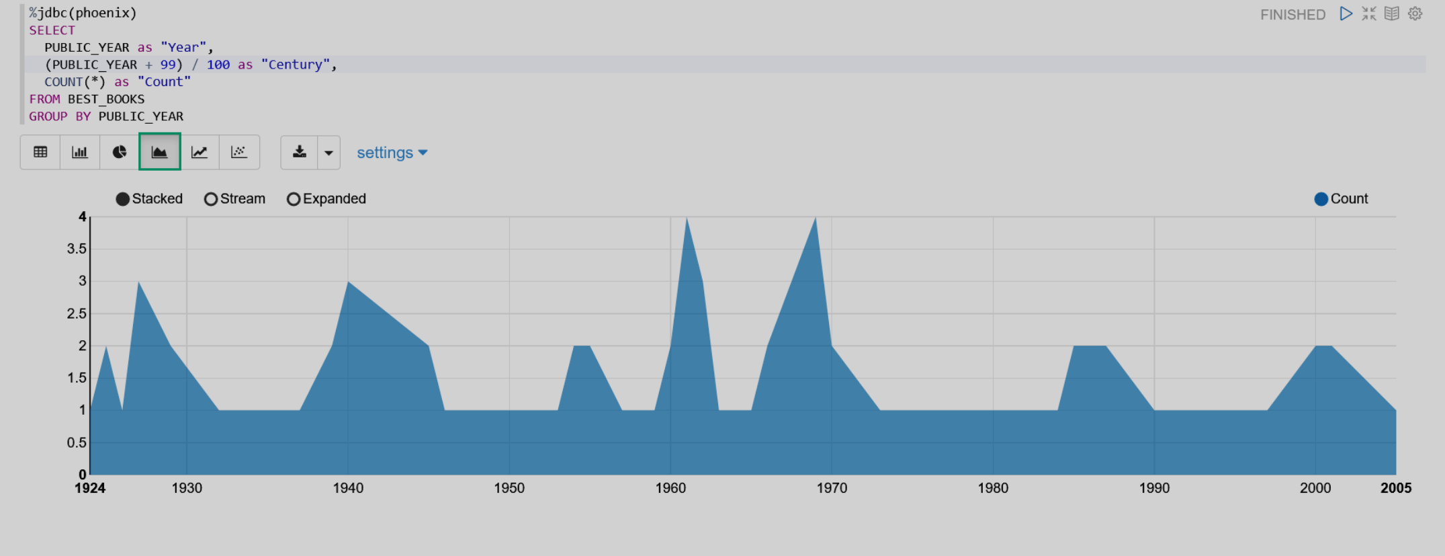
Доступные формы визуализации результатов


```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

Took 7 sec. Last updated by anonymous at December 09 2021, 9:44:43 AM. (outdated)

FINISHED
settings
1924
1925
1926
1927
1929
1932
1934
1936
1937
1939
1940
1945
1946
1947
1949
1950
1951
1952
1953
1954
1955
1957
1958
1959
1960
1961
1962
1963
1965
1966
1969
1970
1973
1975
1981
1984
1985
1987
1990
1992
1996
1997
2000
2001
2005

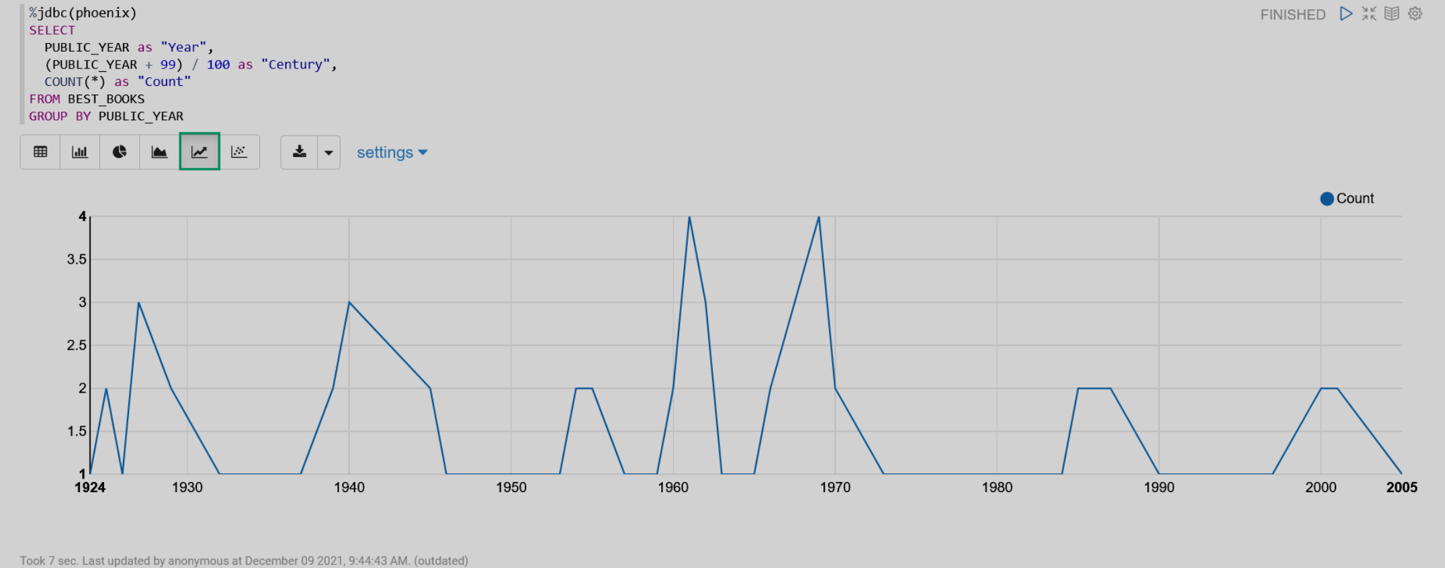
Круговая диаграмма


```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

Took 7 sec. Last updated by anonymous at December 09 2021, 9:44:43 AM. (outdated)

FINISHED
settings
Stacked
Stream
Expanded
Count
4
3.5
3
2.5
2
1.5
1
0.5
0
1924
1930
1940
1950
1960
1970
1980
1990
2000
2005

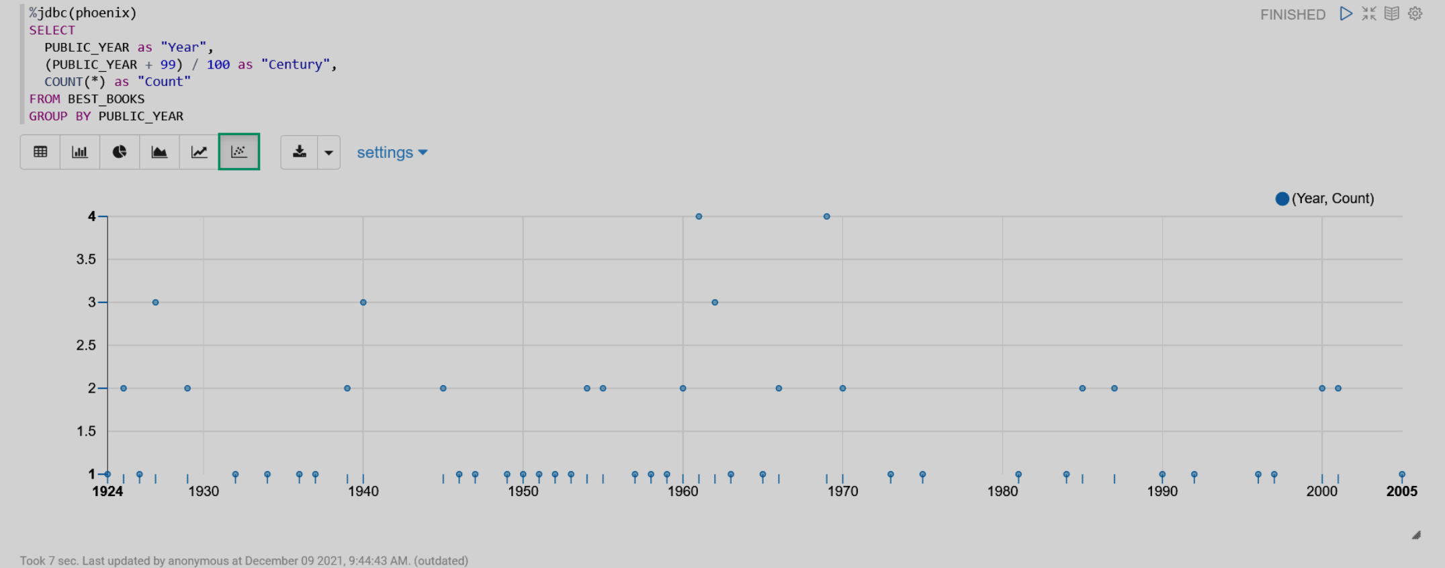
Диаграмма областей


```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

Took 7 sec. Last updated by anonymous at December 09 2021, 9:44:43 AM. (outdated)

FINISHED
settings
Count
4
3.5
3
2.5
2
1.5
1
1924
1930
1940
1950
1960
1970
1980
1990
2000
2005

Линейный график

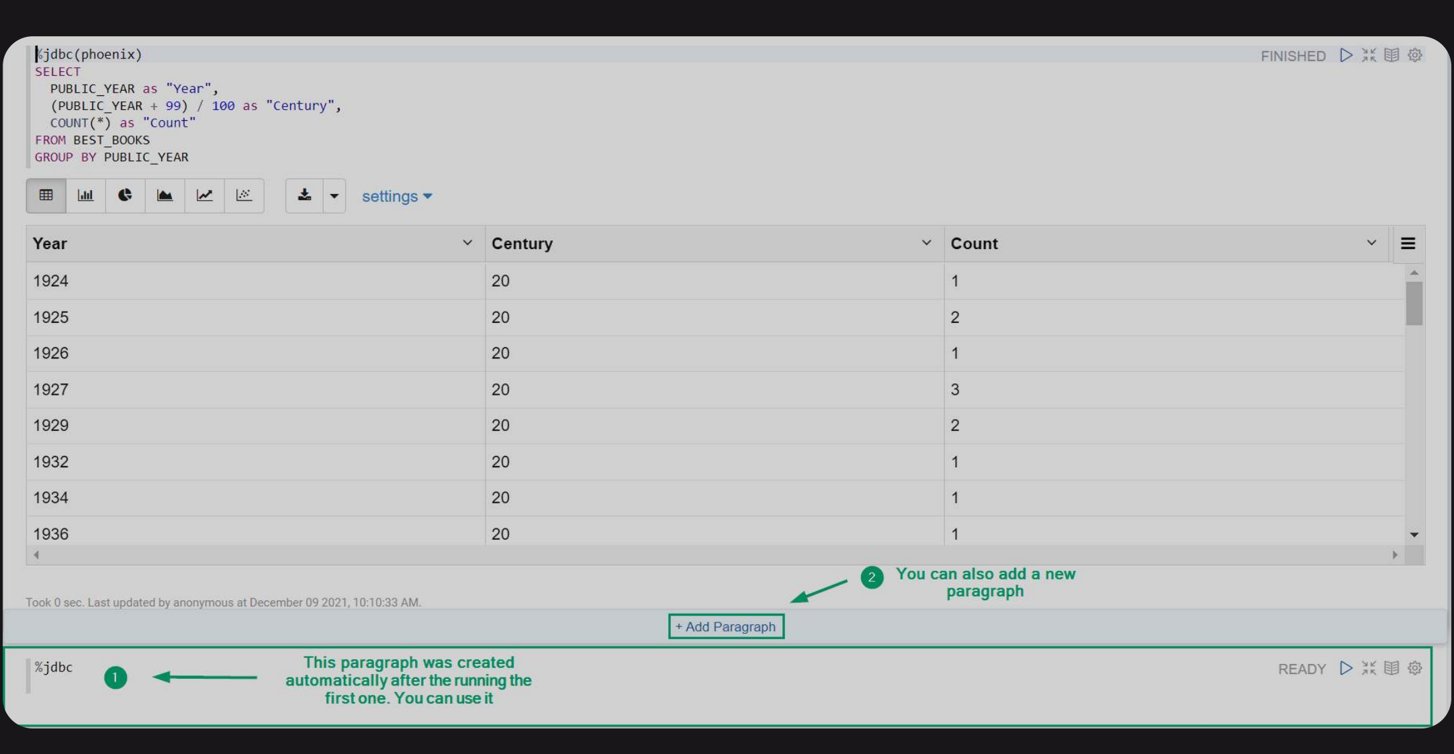

```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

Took 7 sec. Last updated by anonymous at December 09 2021, 9:44:43 AM. (outdated)

FINISHED
settings
(Year, Count)
4
3.5
3
2.5
2
1.5
1
1924
1930
1940
1950
1960
1970
1980
1990
2000
2005

Точечная диаграмма

8. Создайте второй параграф для использования интерпретатора %phoenix. Вы можете использовать параграф, который создается автоматически после успешного выполнения первого параграфа. Также вы можете добавить новый параграф нажатием на кнопку ***Add Paragraph*** (становится видимой при наведении мыши).


```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

Year	Century	Count
1924	20	1
1925	20	2
1926	20	1
1927	20	3
1929	20	2
1932	20	1
1934	20	1
1936	20	1

Took 0 sec. Last updated by anonymous at December 09 2021, 10:10:33 AM.

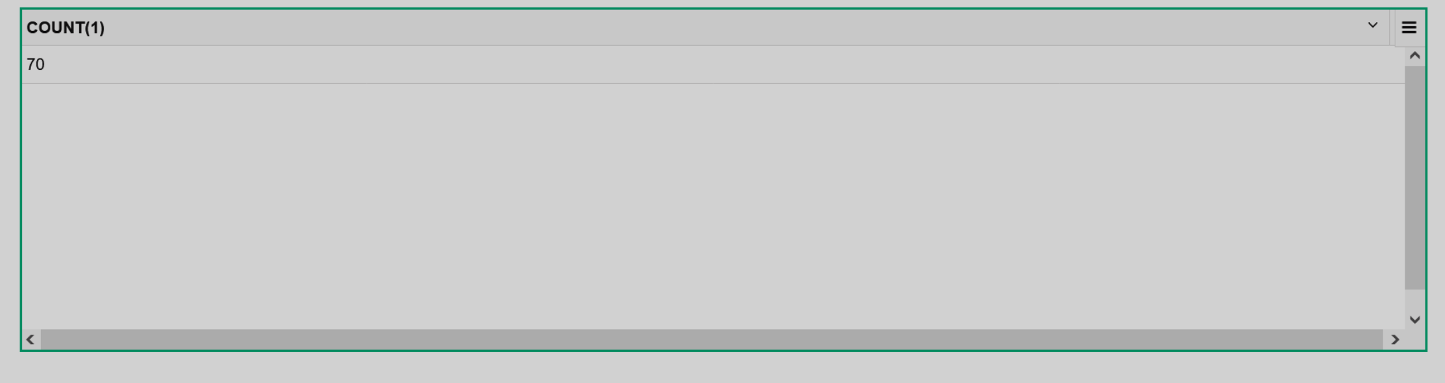
```
%jdbc
```

FINISHED
settings
2 You can also add a new paragraph
+ Add Paragraph
1
This paragraph was created automatically after the running the first one. You can use it
READY

Добавление нового параграфа

9. Введите следующий код во второй параграф и выполните его. Данный запрос возвращает количество строк в таблице `BEST_BOOKS`.

```
%phoenix
SELECT COUNT(*)
FROM BEST_BOOKS
```


```
%phoenix
SELECT COUNT(*)
FROM BEST_BOOKS
```

COUNT(1)
70

Took 3 sec. Last updated by anonymous at December 09 2021, 9:59:55 AM. (outdated)

1 Write the interpreter name and code
FINISHED
Click to run the code 2
settings
3 View the result

Выполнение кода

Создание заметки Solr

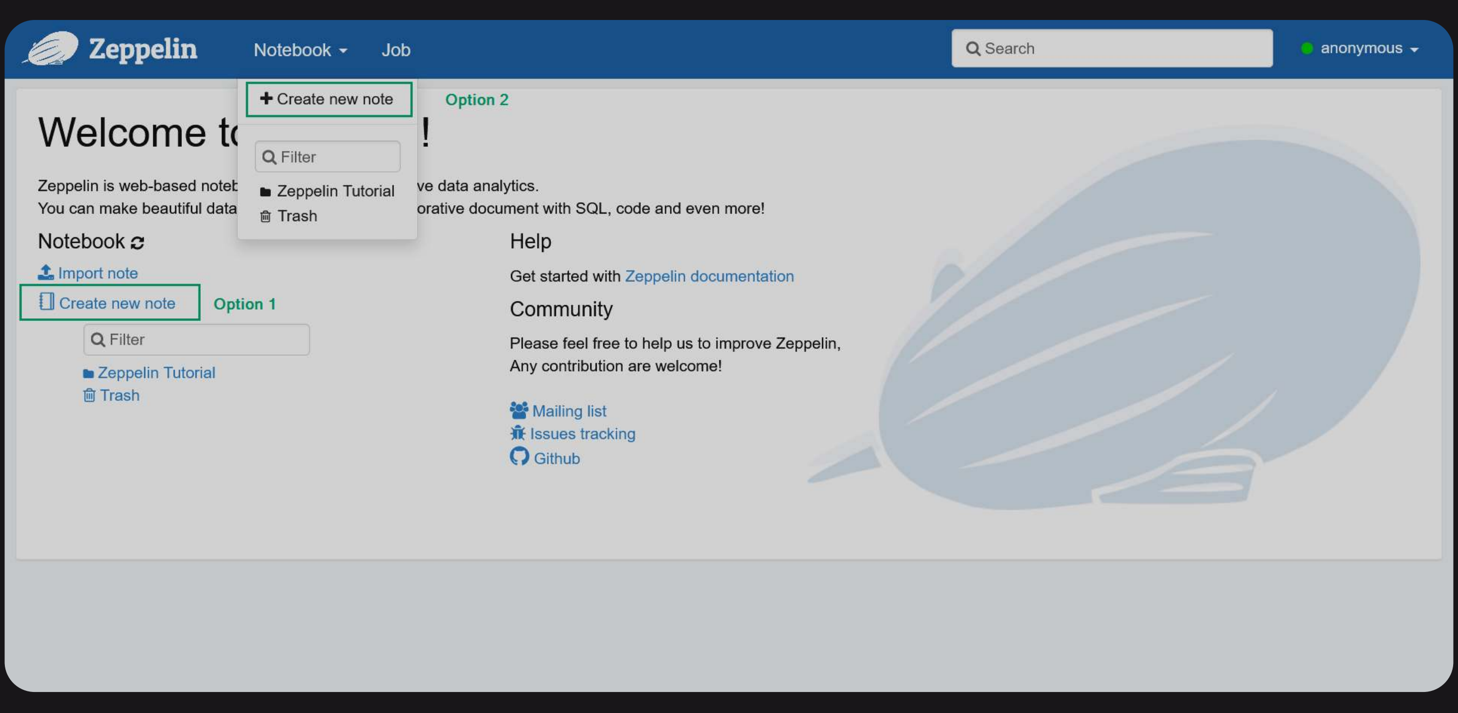
Константин Алпашкин

Назад к содержанию

ПРИМЕЧАНИЕ

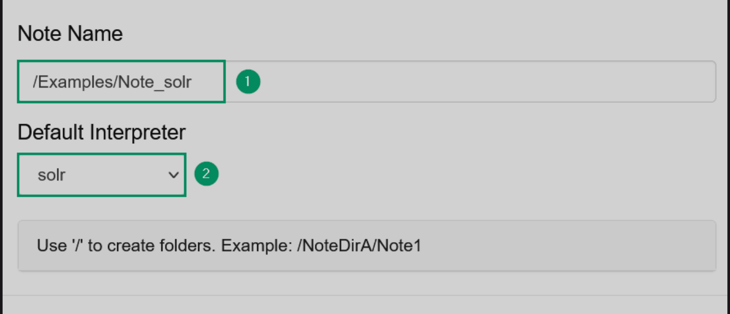
- Перед созданием заметки убедитесь, что интерпретатор Solr настроен, как показано в статье Установка стороннего интерпретатора.
- В этой статье используется Solr-коллекция `people_ages`, которая содержит имена и возраст 1000 человек. Dataset для коллекции доступен по прямой ссылке.

1. В веб-интерфейсе Zeppelin кликните ***Create new note*** или выберите пункт основного меню ***Notebook*** → ***Create new note***.


Zeppelin
Notebook
Job
Search
anonymous
+ Create new note
Option 2
Filter
Zeppelin Tutorial
Trash
Welcome t
!
Zeppelin is web-based noteb
ve data analytics.
You can make beautiful data
orative document with SQL, code and even more!
Notebook
Import note
Create new note
Option 1
Filter
Zeppelin Tutorial
Trash
Help
Get started with Zeppelin documentation
Community
Please feel free to help us to improve Zeppelin,
Any contribution are welcome!
Mailing list
Issues tracking
Github

Добавление новой заметки

2. В открывшемся диалоге введите имя и укажите интерпретатор по умолчанию, который будет использован в случаях, когда явно не указан тип интерпретатора. Обратите внимание, что вы можете создавать папки для ваших заметок. Затем кликните ***Create***.


Create New Note
1
2
Create

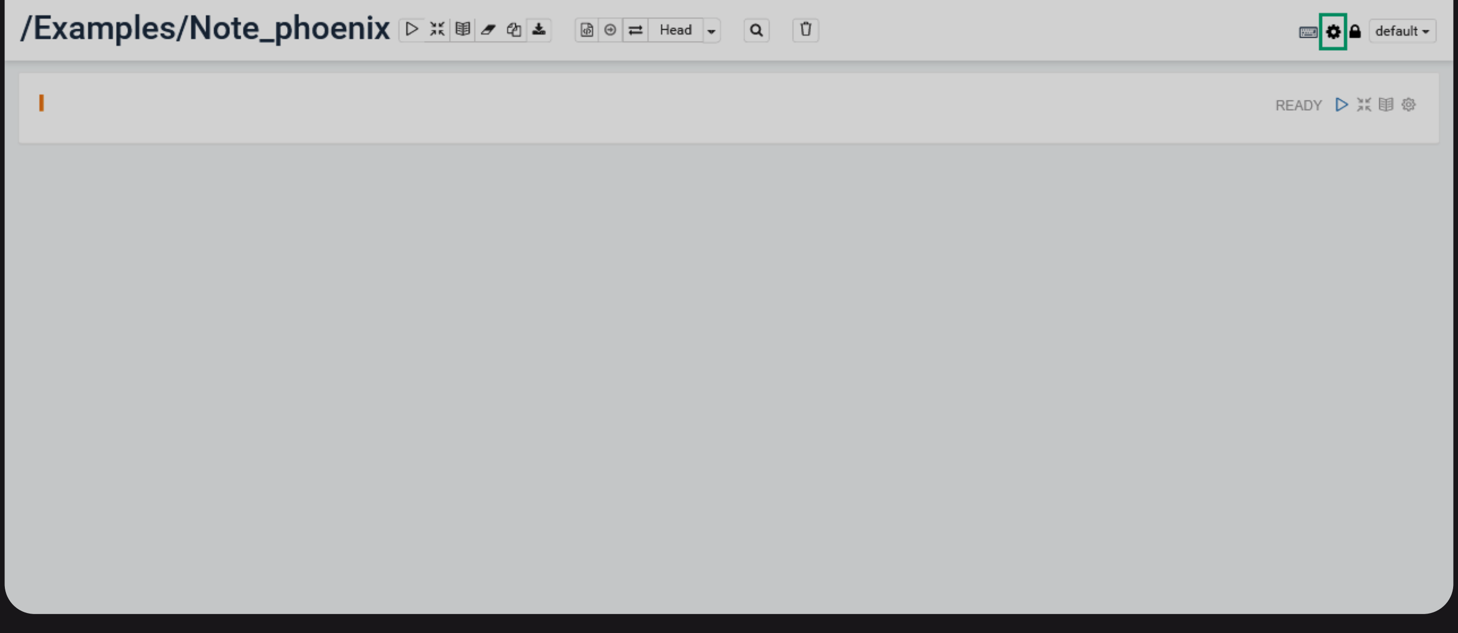
Note Name: /Examples/Note_solr

Default Interpreter: solr

Use '/' to create folders. Example: /NoteDirA/Note1

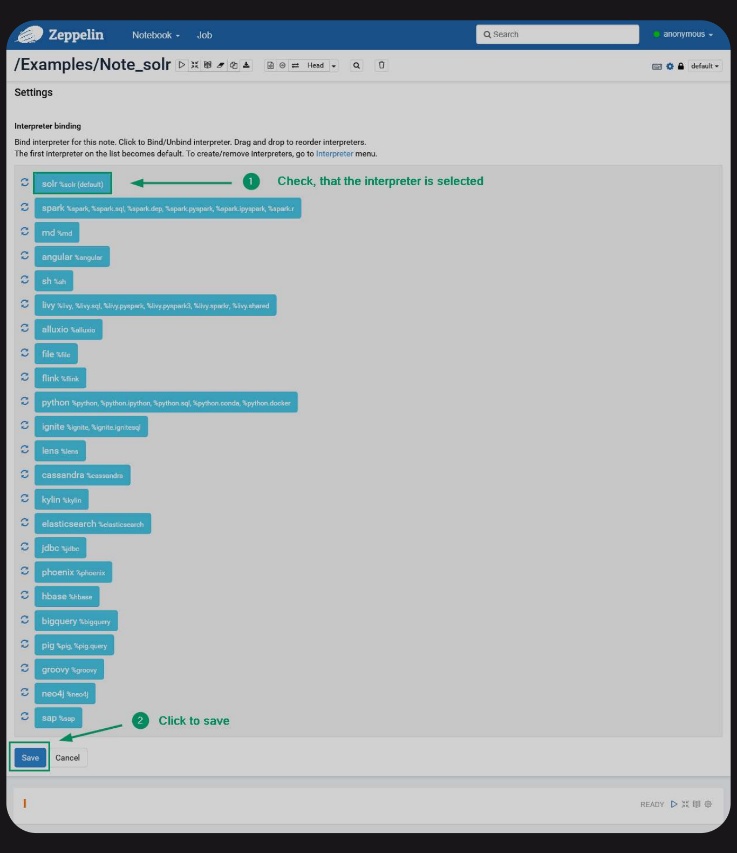
Описание новой заметки

1. В открывшемся окне кликните ***Interpreter binding***, чтобы увидеть доступные интерпретаторы.


Zeppelin
Notebook
Job
Search
anonymous
/Examples/Note_phoenix
Head
default
READY

Переход к сопоставлению интерпретаторов

2. Убедитесь, что интерпретатор `%solr` выбран в открывшемся списке; в противном случае — выберите его. Затем кликните ***Save***.


Zeppelin
Notebook
Job
Search
anonymous
/Examples/Note_solr
Head
default
Settings
1
Check, that the interpreter is selected
solr %solr (default)
spark %spark, %spark.sql, %spark.dep, %spark.pyspark, %spark.ipyspark, %spark.r
md %md
angular %angular
sh %sh
livy %livy, %livy.sql, %livy.pyspark, %livy.pyspark3, %livy.sparkr, %livy.shared
alluxio %alluxio
file %file
flink %flink
python %python, %python.ipython, %python.sql, %python.conda, %python.docker
ignite %ignite, %ignite.ignitesql
lens %lens
cassandra %cassandra
kylin %kylin
elasticsearch %elasticsearch
jdbc %jdbc
phoenix %phoenix
hbase %hbase
bigquery %bigquery
pig %pig, %pig.query
groovy %groovy
neo4j %neo4j
sap %sap
2
Click to save
Save
Cancel
READY

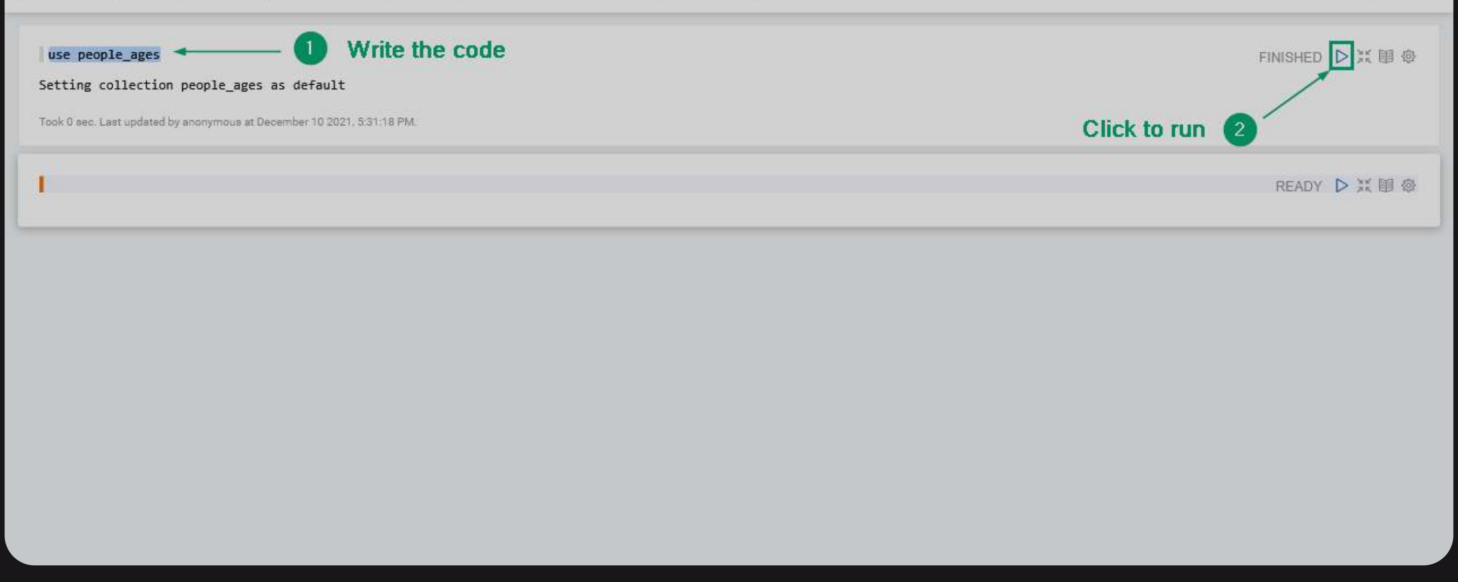
Interpreter binding

Bind interpreter for this note. Click to Bind/Unbind interpreter. Drag and drop to reorder interpreters.
The first interpreter on the list becomes default. To create/remove interpreters, go to Interpreter menu.

Сохранение интерпретатора

3. В первом разделе вашей заметки введите команду, которая выбирает коллекцию по умолчанию для выборки из Solr. Чтобы выполнить код, кликните ***Run this paragraph***.

```
use people_ages
```


Zeppelin
Notebook
Job
Search
anonymous
/Examples/Note_solr
Head
default
1
Write the code
FINISHED
Click to run
2
READY

```
use people_ages
Setting collection people_ages as default
```

Took 0 sec. Last updated by anonymous at December 10 2021, 5:31:18 PM.

Установка Solr-коллекции по умолчанию

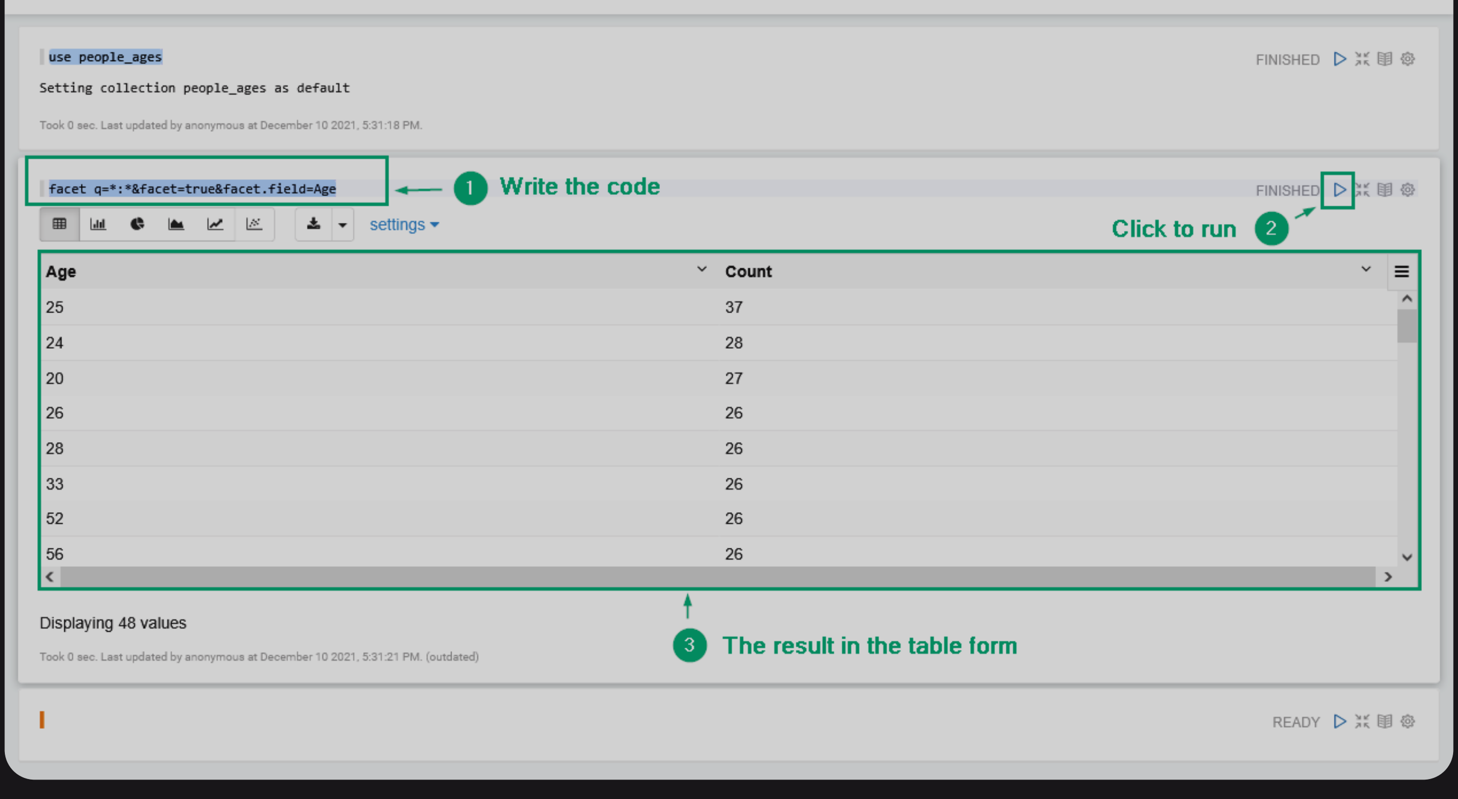
РЕКОМЕНДАЦИЯ

Если не указать имя интерпретатора, по умолчанию используется имя `%solr`.

4. Во втором разделе введите следующий запрос. Он возвращает количество людей, сгруппированных по возрасту.

```
facet q=*:*&facet=true&facet.field=Age
```

Результат приведен ниже.


Zeppelin
Notebook
Job
Search
anonymous
/Examples/Note_solr
Head
default
FINISHED
1
Write the code
settings
Click to run
2
3
The result in the table form
READY

```
use people_ages
Setting collection people_ages as default
```

Took 0 sec. Last updated by anonymous at December 10 2021, 5:31:18 PM.

```
facet q=*:*&facet=true&facet.field=Age
```

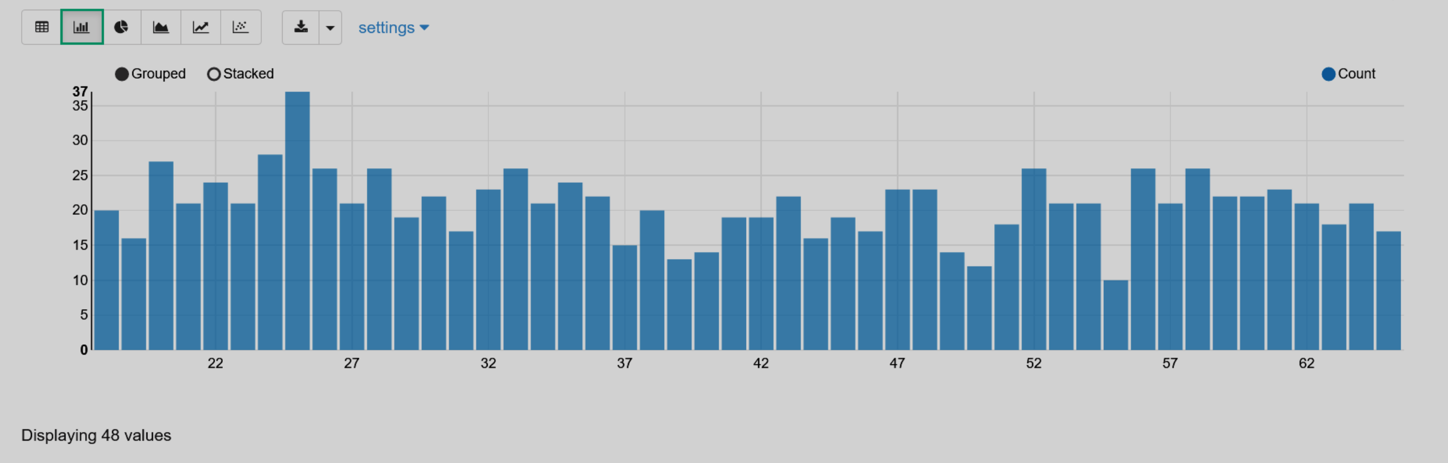
Age	Count
25	37
24	28
20	27
26	26
28	26
33	26
52	26
56	26

Displaying 48 values

Took 0 sec. Last updated by anonymous at December 10 2021, 5:31:21 PM. (outdated)

Таблица с результатами

5. Кликните иконку ***Bar Chart***, чтобы изменить вид результатов.



```
facet q=*:*&facet=true&facet.field=Age
```

Displaying 48 values

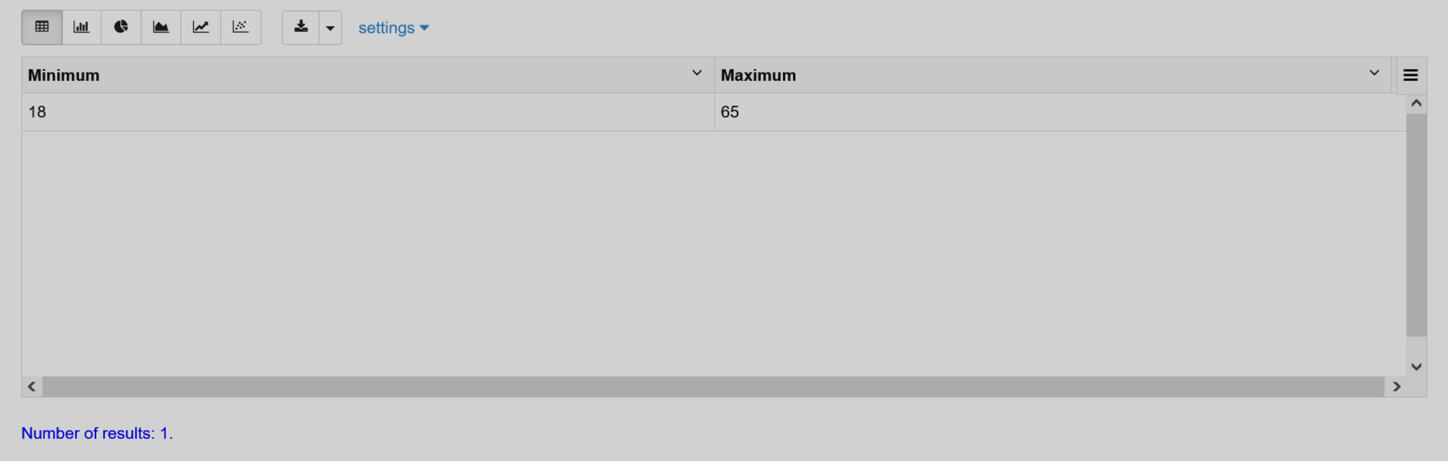
Took 0 sec. Last updated by anonymous at December 10 2021, 5:31:21 PM. (outdated)

Столбчатая диаграмма

6. Вы также можете использовать SQL-выражения. Введите следующий код в третьем разделе:

```
SELECT MIN(Age) AS Minimum, MAX(Age) AS Maximum FROM people_ages
```

Запрос возвращает минимальные и максимальные значения возрастов.


FINISHED
settings

```
SELECT MIN(Age) AS Minimum, MAX(Age) AS Maximum FROM people_ages
```

Minimum	Maximum
18	65

Number of results: 1.

Took 1 sec. Last updated by anonymous at December 10 2021, 6:14:28 PM. (outdated)

Выполнение SQL-запроса

Создание заметки Spark

Константин Алпашкин

Содержание

Назад к содержанию

Интерпретаторы Spark

Общие объекты контекста

Тестовые данные

Создание и запуск заметки Spark

- Scala
- PySpark
- SparkSQL

В данной статье описан процесс создания и запуска заметок Spark в Zeppelin. По умолчанию сервис Zeppelin использует ядро Spark3-сервиса в качестве бэкенда для выполнения заметок Spark. Если сервисы Spark3 и Zeppelin установлены в ADH, для выполнения кода заметок Zeppelin запускает задачи Spark в режиме YARN-кластера ADH.

Интерпретаторы Spark

Сервис Zeppelin поставляется с группой интерпретаторов `spark`, которая включает набор интерпретаторов для работы с различными языками, такими как Scala, Python и R. В следующей таблице перечислены доступные интерпретаторы.

Интерпретатор	Класс	Описание
%spark	SparkInterpreter	Предоставляет среду для выполнения кода Spark на Scala
%spark.sql	SparkSQLInterpreter	Позволяет выполнять ANSI SQL-запросы к объектам Spark DataFrame/DataSet
%spark.pyspark	PySparkInterpreter	Предоставляет среду для выполнения кода Spark на Python
%spark.ipyspark	IPySparkInterpreter	Предоставляет среду для выполнения кода Spark с использованием IPython
%spark.r	SparkRInterpreter	Предоставляет среду для выполнения кода Spark на языке R с поддержкой SparkR
%spark.ir	SparkIRInterpreter	Предоставляет среду на основе Jupyter IRKernel для выполнения кода Spark на языке R с поддержкой SparkR
%spark.shiny	SparkShinyInterpreter	Используется для запуска приложений на языке R с поддержкой фреймворка Shiny и SparkR

Общие объекты контекста

В данной статье предполагается, что интерпретатор `spark` используется со стандартными настройками биндинга (shared binding mode), то есть интерпретатор инстанциируется глобально в shared-режиме. Это означает, что Zeppelin создает один JVM-процесс с одной группой интерпретаторов (interpreter group) для обслуживания всех заметок Spark. В результате переменные, объявленные в одной заметке, доступны в других заметках при условии, что заметки используют один и тот же тип интерпретатора (например, обе заметки используют `%spark.pyspark`).

Для среды Scala, Python и R Zeppelin автоматически создает объекты `SparkContext`, `SQLContext`, `SparkSession` и `ZeppelinContext` и помещает их в переменные `sc`, `sqlContext`, `spark` и `z` соответственно. Являясь shared-переменными, одни и те же экземпляры этих объектов одновременно доступны в разных заметках Spark.

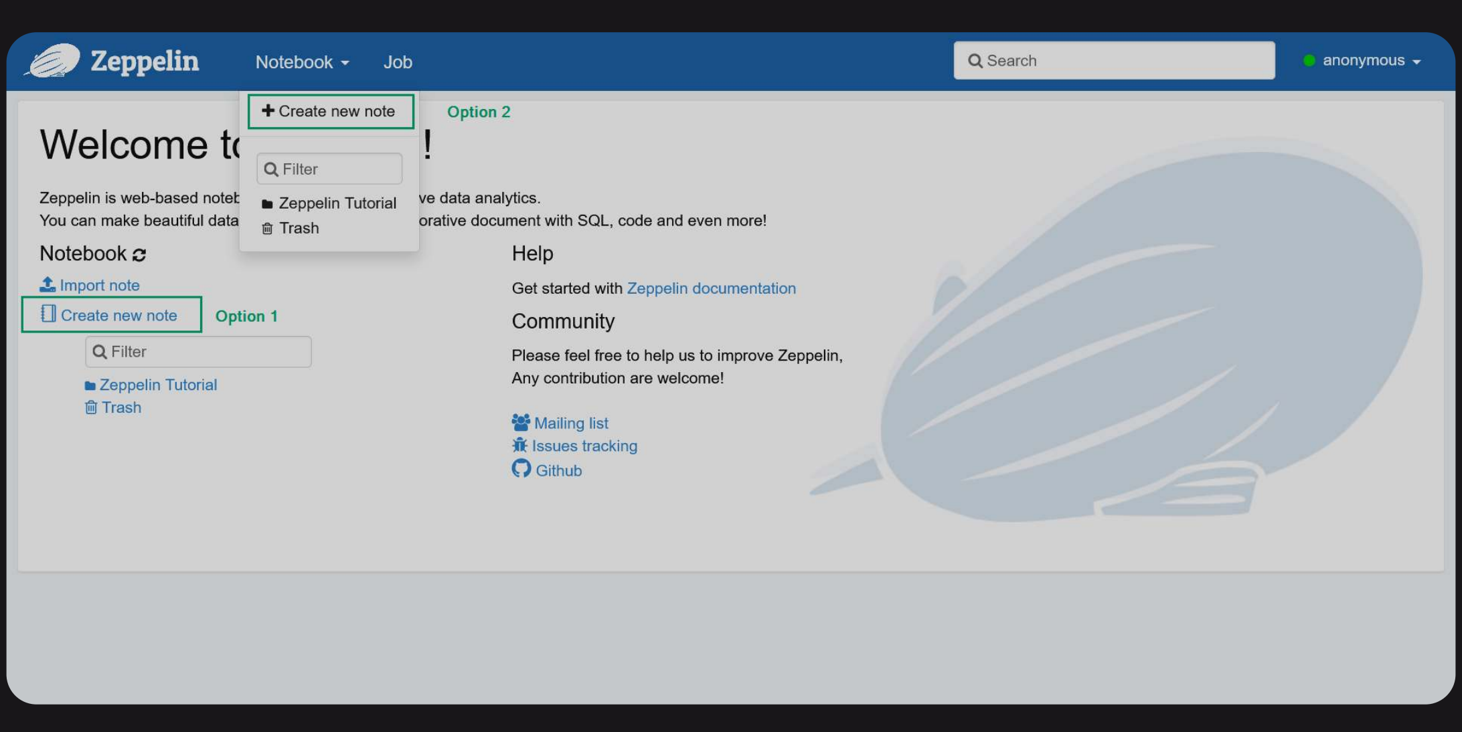
Тестовые данные

В статье в качестве примера используется тестовый набор данных people.csv. Для работы некоторых примеров датасет необходимо загрузить в HDFS. Для этого воспользуйтесь следующими командами:

```
$ hdfs dfs -mkdir /user/zeppelin/test_data
$ hdfs dfs -chown zeppelin:hadoop /user/zeppelin/test_data
$ hdfs dfs -put people.csv /user/zeppelin/test_data/people.csv
```

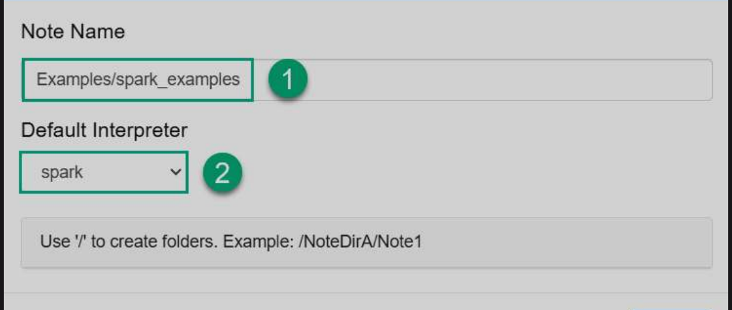
Создание и запуск заметки Spark

1. В веб-интерфейсе Zeppelin кликните ***Create new note*** или выберите пункт основного меню ***Notebook*** → ***Create new note***.



Добавление новой заметки

2. В открывшемся диалоге введите имя и укажите интерпретатор по умолчанию, который будет использован в случаях, когда явно не указан тип интерпретатора. Обратите внимание, что вы можете создавать папки для ваших заметок. Затем кликните ***Create***.



Create New Note

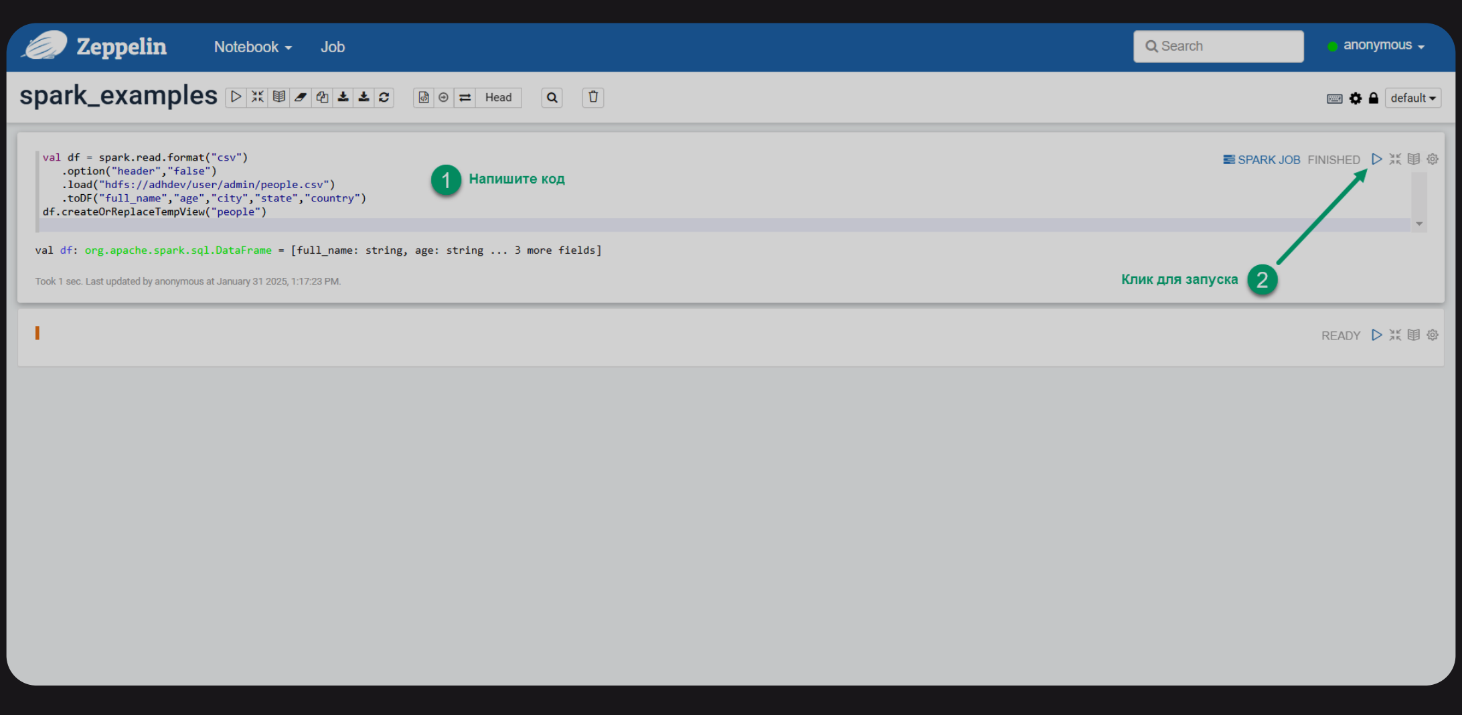
Note Name: Examples/spark_examples ①

Default Interpreter: spark ②

Use '/' to create folders. Example: /NoteDirA/Note1

Создание новой заметки

3. Введите код Spark в параграфе (paragraph) заметки и запустите параграф.



```
val df = spark.read.format("csv")
    .option("header","false")
    .load("hdfs://adhdev/user/admin/people.csv")
    .toDF("full_name","age","city","state","country")
df.createOrReplaceTempView("people")
```

```
val df: org.apache.spark.sql.DataFrame = [full_name: string, age: string ... 3 more fields]
```

Took 1 sec. Last updated by anonymous at January 31 2025, 1:17:23 PM.

Запуск кода Spark в Zeppelin

Далее приведены примеры кода Spark на Scala, PySpark и SparkSQL. Больше примеров доступно в заметках Spark-туториала, доступном в Zeppelin (***Notebook*** → ***Spark Tutorial***).

Scala

Для запуска Spark-заметки на Scala укажите интерпретатор `%spark` в первой строке параграфа. Технически интерпретатор `%spark` можно не указывать, так как он является стандартным в группе, однако рекомендуется объявлять интерпретаторы для каждого параграфа заметки.

```
%spark ①
val df_people = spark.read.format("csv") ②
   .option("header","false")
   .load("hdfs://adhdev/user/zeppelin/test_data/people.csv")
   .toDF("full_name","age","city","state","country")
df_people.printSchema()

case class SW_movie(id:Int, name:String, episode:Int, year:Int)

val df_sw_movies = spark.createDataFrame(Seq ③
    (
        SW_movie(1, "Revenge of the Sith", 3, 2005),
        SW_movie(2, "Attack of the Clones", 2, 2002),
        SW_movie(3, "The Phantom Menace", 1, 1999),
        SW_movie(4, "A New Hope", 4, 1977),
        SW_movie(5, "Return of the Jedi", 6, 1983),
        SW_movie(6, "The Empire Strikes Back", 5, 1980)
    )
)
df_sw_movies.printSchema()
val df_sw_movies_comments = df_sw_movies.withColumn("rank", lit(0.0)) ④
df_sw_movies_comments.printSchema()
df_sw_movies_comments.drop("rank") ⑤

val sorted_by_year = rdd.map(mv => ⑥
    (mv._1,mv._2,mv._3)
  ).sortBy(_._3).toDF("name", "episode", "year")
sorted_by_year.show()

sorted_by_year.write ⑦
  .mode("overwrite")
  .json("/user/zeppelin/test_data/star_wars_movies")
```

① Явное указание интерпретатора для выполнения кода Spark на языке Scala.

② Создание DataFrame из тестовых данных в HDFS.

③ Создание DataFrame из кортежа Case-class объектов.

④ Добавление в DataFrame столбца со статическими данными.

⑤ Создание DataFrame с применением сортировки по произвольному столбцу.

⑥ Удаление столбца из DataFrame.

⑦ Запись содержимого DataFrame в HDFS в виде JSON.

PySpark

Чтобы выполнить заметку PySpark, объявите интерпретатор `%spark.pyspark` в первой строке параграфа.

```
%spark.pyspark ①
df = spark.createDataFrame([ ②
    (1, "Ivan", 20, "Russia"),
    (2, "Yao", 23, "China"),
    (3, "Ann", 28, "Russia"),
    (4, "John", 20, "USA")
]).toDF("id", "name", "age", "country")
df.printSchema()
df.show()

from pyspark.sql.types import StructType
peopleSchema = StructType() \
    .add("full_name", "string") \
    .add("age", "integer") \
    .add("city", "string") \
    .add("state", "string") \
    .add("country", "string")
df_people = spark.read.csv("hdfs://adhdev/user/zeppelin/test_data/people.csv", schema=peopleSchema) ③
df_people.printSchema();

df_name_city = df_people.select("full_name", "city") ④
df_name_city.show(10)

import pyspark.sql.functions as f
df_name_uppered = df_people.withColumn("full_name_upper", f.upper(df_people["full_name"])) ⑤
df_name_uppered.printSchema()

df_people_stateless = df_people.drop("state"); ⑥
df_people_stateless.printSchema();

df = df_people.filter( ⑦
    (df_people["age"] >= 20) & (df_people["city"] == "Houston")
)
df.show()

from pyspark.sql.functions import udf
from pyspark.sql.types import BooleanType
@udf(returnType=BooleanType()) ⑧
def udf_older_than(e):
    if e > 60:
        return True;
    else:
        return False
df_people_older_than60 = df_people.filter(udf_older_than(df_people["age"]))
df_people_older_than60.show()
```

① Явное указание интерпретатора для выполнения кода PySpark.

② Создание DataFrame из списка кортежей Python.

③ Создание DataFrame из тестовых данных в HDFS.

④ Создание нового DataFrame путем выборки определенных столбцов из другого DataFrame.

⑤ Добавление столбца в DataFrame; строки в столбце приведены к верхнему регистру с помощью `pyspark.sql.functions`.

⑥ Удаление столбца из DataFrame.

⑦ Применение нескольких фильтров выборки.

⑧ Объявление UDF, которая далее используется при фильтрации.

SparkSQL

В заметках Scala/PySpark можно выполнять SQL-запросы к DataFrame с помощью конструкции `spark.sql()`. Например:

```
%spark.pyspark ①

from pyspark.sql.types import StructType

peopleSchema = StructType() \
    .add("full_name", "string") \
    .add("age", "integer") \
    .add("city", "string") \
    .add("state", "string") \
    .add("country", "string")
df_people = spark.read.csv("hdfs://adhdev/user/zeppelin/test_data/people.csv", schema=peopleSchema)
df_people.createOrReplaceTempView("people") ②

df_people30 = spark.sql("select full_name, age from people where age=30") ③
```

① Явное указание интерпретатора для выполнения кода PySpark.

② Создание временного представления (temporary view) на основе содержимого DataFrame. Это действие необходимо для выполнения запросов SparkSQL.

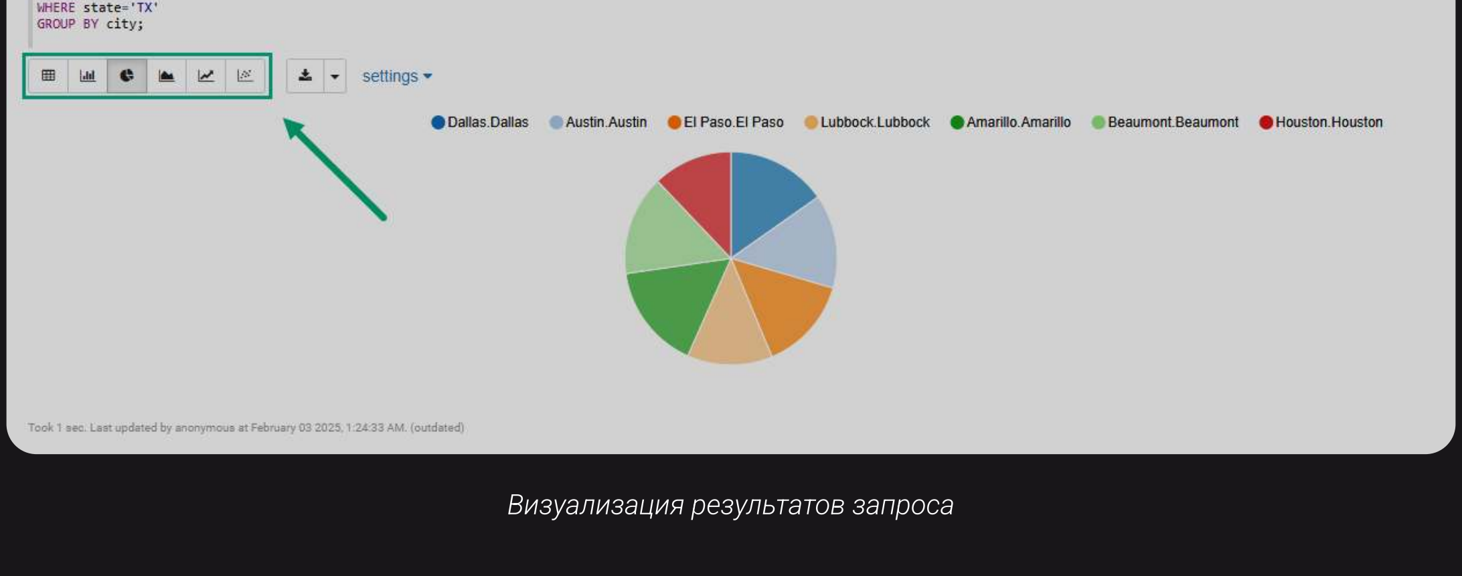
③ SQL-запрос к временному представлению. Результаты запроса сохраняются в отдельный DataFrame.

Указав интерпретатор `%spark.sql` в первой строке параграфа, можно визуализировать результаты запросов к объектам Spark DataFrame с помощью встроенных виджетов (диаграммы, графики, таблицы и так далее). Например:

```
%spark.sql

SELECT AVG(age), city
FROM people ①
WHERE state='TX'
GROUP BY city;
```

① Перед выполнением запроса объект `people` должен быть зарегистрирован в качестве временного представления (temporary view).

При выполнении заметки с интерпретатором `%spark.sql` результаты по умолчанию отображаются в виде таблицы. Используя встроенные виджеты, можно визуализировать данные в виде круговой диаграммы, линейного графика и прочих, как показано на рисунке ниже.



```
%spark.sql
SELECT AVG(age), city
FROM people
WHERE state='TX'
GROUP BY city;
```

Took 1 sec. Last updated by anonymous at February 03 2025, 1:24:33 AM. (outdated)

Визуализация результатов запроса

Создание заметки Python

Константин Алпашкин

Содержание

Назад к содержанию

- Выбор Python-интерпретатора
- Интерпретаторы Zeppelin для Python
- Общие переменные
- Создание и запуск заметки Python
  - Python (%python)
  - IPython (%python.ipython)
  - SQL для Pandas DataFrames (%python.sql)

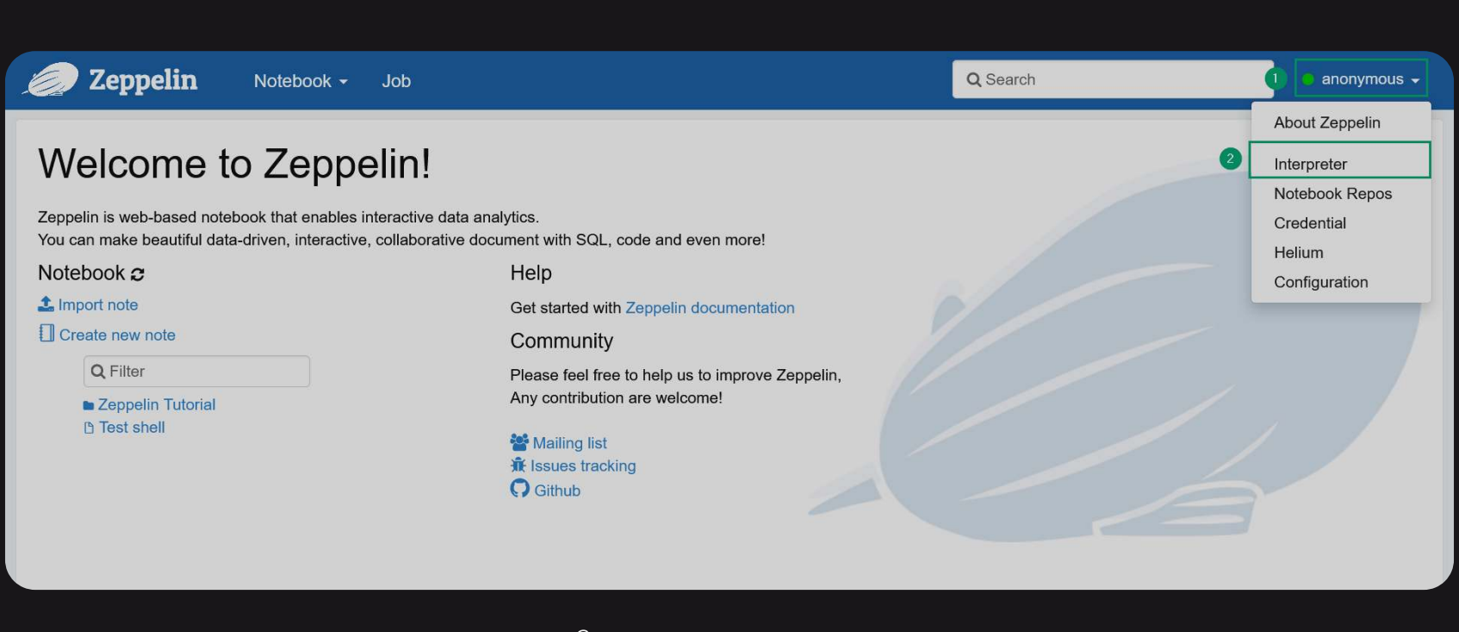
В данной статье описан процесс создания и запуска заметок Python в Zeppelin. По умолчанию для выполнения кода заметок на Python сервис Zeppelin использует интерпретатор Python, доступный в переменной окружения `$PATH`. Интерпретатор Python, используемый сервисом Zeppelin, можно изменить с помощью свойства `zeppelin.interpreter`. Больше информации доступно в разделе Выбор Python-интерпретатора.

Выбор Python-интерпретатора

Различные дистрибутивы операционных систем поставляются с разными версиями Python, добавленными в `$PATH`. ОС может поставляться с Python 2.x, Python 3 или вообще не содержать Python. В Zeppelin можно указать Python-интерпретатор определенной версии для выполнения кода заметок.

Для этого выполните следующие действия:

1. Выберите ***Interpreter*** в меню аккаунта, расположенном в правом верхнем углу домашней страницы Zeppelin.



Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Please feel free to help us to improve Zeppelin,
Any contribution are welcome!

Список интерпретаторов

2. Найдите интерпретатор `python` и кликните ***Edit***.

3. В параметре `zeppelin.python` укажите путь к нужному исполняемому файлу Python.

> **РЕКОМЕНДАЦИЯ**
>
> Интерпретатор Python, поставляемый с ADH, находится в директории */opt/<python_env>*.

4. Сохраните конфигурацию и перезапустите интерпретатор.

5. Выполните тестовую заметку, чтобы убедиться, что Zeppelin использует новый интерпретатор Python:

```
%python

import sys
print(f"""Current note has been executed using:
    Python location: {sys.executable}
    Python version: {sys.version_info[0]}
    """)
```

Пример вывода:

```
Current note has been executed using:
    Python location: /opt/python3.10/bin/python3
    Python version: 3
```

Интерпретаторы Zeppelin для Python

Сервис Zeppelin поставляется с группой интерпретаторов `python`, которая включает набор интерпретаторов, перечисленных в таблице. Все перечисленные интерпретаторы поддерживают Python 2 и Python 3.

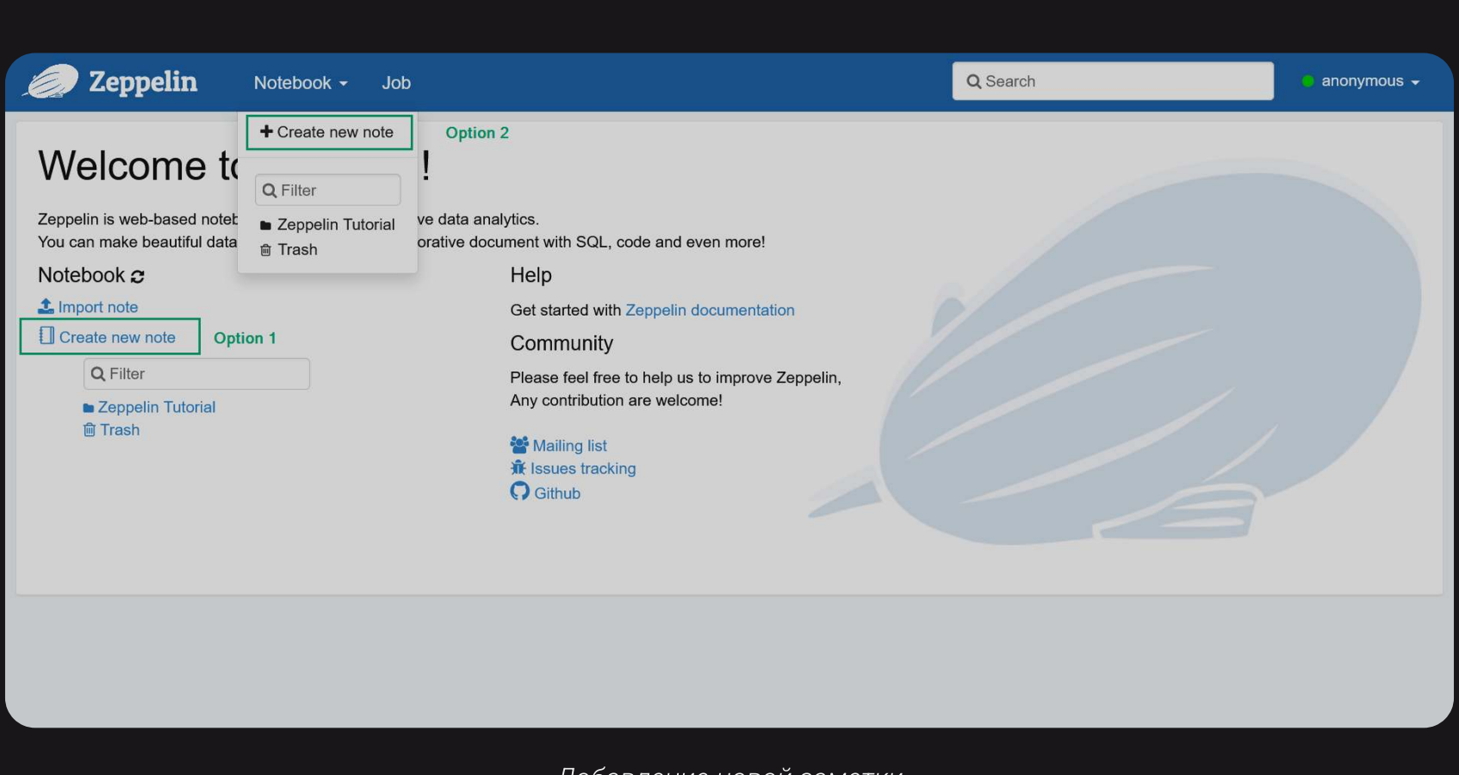
Интерпретатор	Класс	Описание
%python	PythonInterpreter	Ванильный интерпретатор Python с базовыми зависимостями. Требует установки только среды Python. Использует возможности IPython, если выполняются предварительные условия
%python.ipython	IPythonInterpreter	Предоставляет более функциональное окружение Python, построенное на IPython. Данный интерпретатор считается предпочтительным вариантом для работы с заметками Python
%python.sql	PythonInterpreterPandasSql	Предоставляет SQL-возможности для работы с Pandas DataFrames через `pandasql`

Общие переменные

В данной статье предполагается, что интерпретатор `python` используется со стандартными настройками биндинга (shared binding mode), то есть интерпретатор инстанциируется глобально в shared-режиме. Это означает, что Zeppelin создает один JVM-процесс с одной группой интерпретаторов (interpreter group) для обслуживания всех заметок Python. В результате переменные, объявленные в одной заметке, доступны и в других заметках. Совместное использование переменных также работает между заметками типа `%python` и `%python.ipython`.

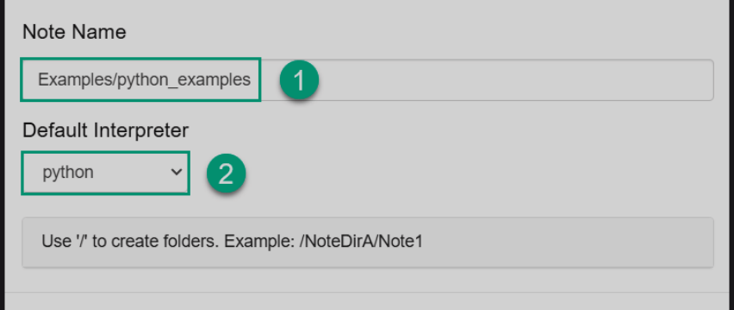
Создание и запуск заметки Python

1. В веб-интерфейсе Zeppelin кликните ***Create new note*** или выберите пункт основного меню ***Notebook*** → ***Create new note***.



Добавление новой заметки

2. В открывшемся диалоге введите имя и укажите интерпретатор по умолчанию, который будет использован в случаях, когда явно не указан тип интерпретатора. Обратите внимание, что вы можете создавать папки для ваших заметок. Затем кликните ***Create***.



Create New Note

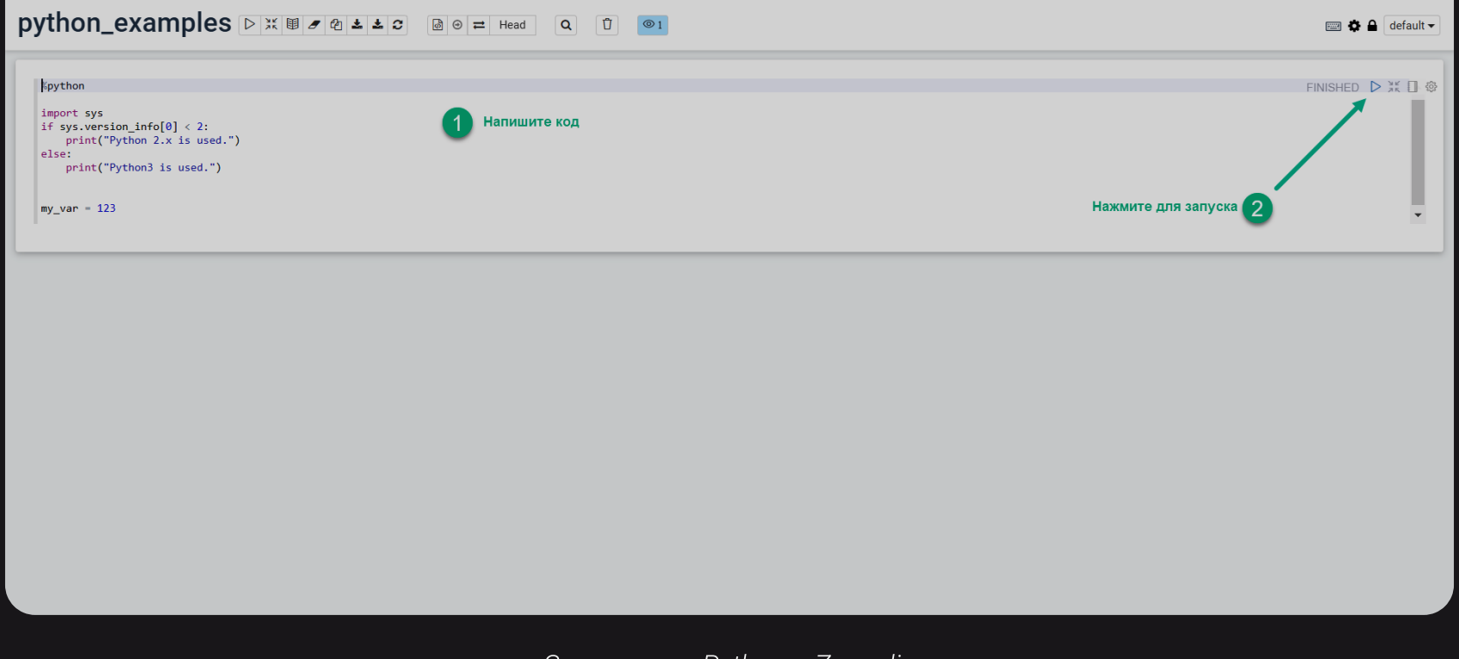
Note Name: Examples/python_examples

Default Interpreter: python

Use '/' to create folders. Example: /NoteDirA/Note1

Создание новой заметки

3. Введите код Python в параграфе (paragraph) заметки и выполните параграф.



```
%python

import sys
if sys.version_info[0] < 2:
    print("Python 2.x is used.")
else:
    print("Python3 is used.")

my_var = 123
```

Запуск кода Python в Zeppelin

Ниже приведены примеры заметок на Python. Код заметок доступен в виде архива .zpln, который можно импортировать в веб-интерфейс Zeppelin (***Zeppelin*** → ***Import Note***). Больше примеров доступно в Zeppelin UI (***Notebook*** → ***Python Tutorial***).

Python (%python)

Чтобы использовать ванильный Zeppelin-интерпретатор для Python, укажите `%python` в первой строке параграфа. Используйте `%python` когда нужна только базовая среда Python без дополнительных зависимостей.

В следующем примере показано базовое использование Pandas DataFrame и встроенной визуализации Zeppelin.

```
%python ①
import pandas as pd ②

txn_data = {'acc_id': [1001,1003,1002,1001,1001,1002,1003,1001,1003],
        'txn_value': [25.00, 30.00, 100.00, 250.00, 98.50, 23.50, 45.00, 60.00, 75.15],
        'txn_date': ['2025-01-07', '2025-01-03', '2025-01-09', '2025-01-02', '2025-01-01', '2025-01-04', '2025-01-08', '2025-01-11', '2025-01-03']}
df = pd.DataFrame(txn_data) ③
z.show(df) ④
```

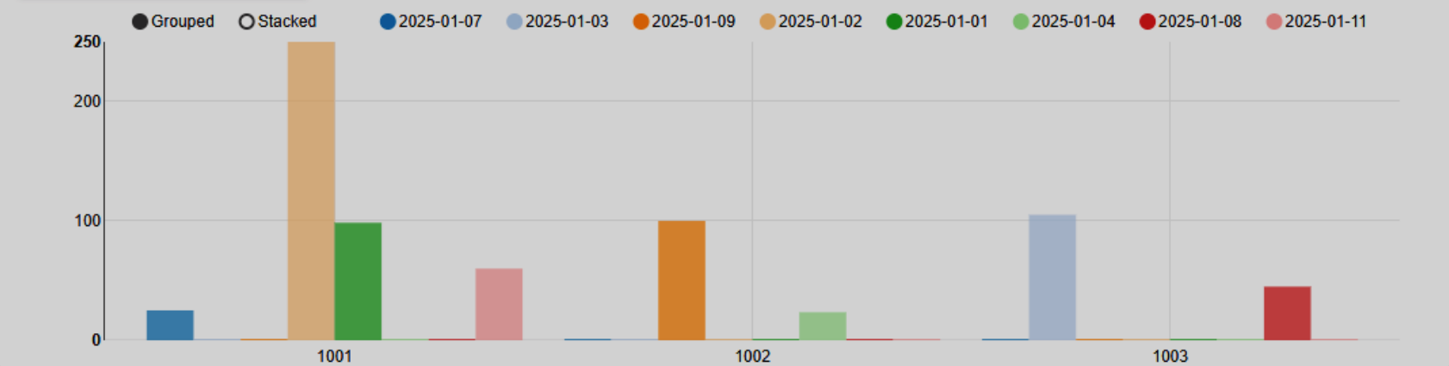
① Явное указание Zeppelin-интерпретатора для Python.

② Перед запуском заметки импортируемые модули должны быть установлены с помощью `pip install <module_name>`.

③ Создание Pandas DataFrame.

④ Визуализация DataFrame с помощью встроенных виджетов Zeppelin. Больше примеров визуализации доступно в разделе IPython.

Результат выполнения заметки представлен ниже.



Встроенная визуализация Zeppelin

IPython (%python.ipython)

IPython (`%python.ipython`) — это Jupyter-подобный интерпретатор, построенный на IPython и предоставляющий богатую среду для интерактивных вычислений, например:

- Улучшенное автозаполнение, подсветка синтаксиса и вывода.
- Специальные функции, такие как `%timeit`, `%matplotlib inline` и так далее.
- Расширенные возможности визуализации с использованием сторонних библиотек, таких как Seaborn, Plotly, Bokeh и так далее.

Чтобы использовать Zeppelin-интерпретатор `%python.ipython`, установите следующие pip-модули в Python-бэкенд, используемый Zeppelin:

- `jupyter-client`
- `ipython`
- `ipykernel`
- `grpcio`
- `protobuf`

> **ПРИМЕЧАНИЕ**
>
> В зависимости от версии Python, который используется сервисом Zeppelin, возможны конфликты при установке последней версии `protobuf`. Одним из вариантов обхода проблемы является понижение версии или установка более старой версии `protobuf`. Например, по состоянию на Python 3.10, который использовался для примеров в этой статье, версия `protobuf` 3.19.6 работала корректно.

Чтобы использовать возможности IPython, укажите `%python.ipython` в первой строке параграфа. Пример:

```
%python.ipython ①

import pandas as pd

data = {
    'full_name': ['Sarah Connor',
        'Kirk Hammett',
        'Ivan Ivanov'],
    'age': [25, 30, 35],
    'dep': ['it', 'sales', 'hr']}
df = pd.DataFrame(data) ②
print(" Describe DF:".center(50, "*"))
print(df.describe())
print(" DF types: ".center(50, "*"))
print(df.dtypes)
print(" Filter DF: ".center(50, "*"))
df1 = df[df['age'] > 30]
print(df1)
print(" Add column to DF: ".center(50, "*"))
df['salary'] = [100000, 200000, 300000]
print(df)
print(" Magic functions: ".center(50, "*"))
range? ③
%timeit range(100) ④
```

① Явное указание интерпретатора IPython.

② Создание Pandas DataFrame с последующими базовыми операциями.

③ Специальная функция IPython, которая выводит справочную информацию о методе.

④ Специальная функция IPython для замера длительности работы метода.

Пример вывода заметки

```
****************** Describe DF:*******************
        age
count   3.0
mean   30.0
std     5.0
min    25.0
25%    27.5
50%    30.0
75%    32.5
max    35.0
******************* DF types: ********************
full_name    object
age           int64
dep          object
dtype: object
******************* Filter DF: *******************
     full_name  age dep
2  Ivan Ivanov   35  hr
*************** Add column to DF: ****************
      full_name  age    dep  salary
0  Sarah Connor   25     it  100000
1  Kirk Hammett   30  sales  200000
2   Ivan Ivanov   35     hr  300000
**************** Magic functions: ****************
108 ns ± 0.816 ns per loop (mean ± std. dev. of 7 runs, 10,000,000 loops each)
Init signature: range(self, /, *args, **kwargs)
Docstring:
range(stop) -> range object
range(start, stop[, step]) -> range object

Return an object that produces a sequence of integers from start (inclusive)
to stop (exclusive) by step.  range(i, j) produces i, i+1, i+2, ..., j-1.
start defaults to 0, and stop is omitted!  range(4) produces 0, 1, 2, 3.
These are exactly the valid indices for a list of 4 elements.
When step is given, it specifies the increment (or decrement).
Type:           type
Subclasses:
```

Ниже приведены примеры визуализации данных в Zeppelin с помощью различных библиотек Python.

Matplotlib

Чтобы выполнить следующую заметку, установите Matplotlib в интерпретатор Python, который используется сервисом Zeppelin. Например:

```
$ /opt/python3.10/bin/python3 -m pip install matplotlib
```

Код заметки:

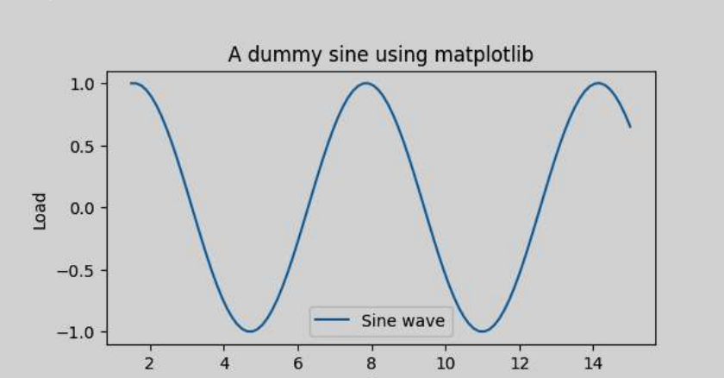
```
%python.ipython ①

import sys
import numpy as np ②
import matplotlib.pyplot as plt

print("Dummy text")

plt.figure(figsize=(6, 3))
x = np.linspace(1.5, 15, 100)
y = np.sin(x)
plt.plot(x, y, label="Sine wave")
plt.xlabel("Time")
plt.ylabel("Load")
plt.title("A dummy sine using matplotlib")
plt.legend()
plt.show() ③
```

① Явное указание интерпретатора IPython.

② Перед запуском заметки импортируемые модули должны быть установлены с помощью `pip install <module_name>`.

③ Рендеринг графика с помощью библиотеки `matplotlib`.

Пример графика приведен ниже.



Визуализация с помощью Matplotlib

Seaborn

Чтобы выполнить следующую заметку, установите Seaborn в интерпретатор Python, который используется сервисом Zeppelin. Например:

```
$ /opt/python3.10/bin/python3 -m pip install seaborn
```

Код заметки:

```
%python.ipython ①
%matplotlib inline ②

import seaborn as sns ③
import matplotlib.pyplot as plt
import pandas as pd

txn_data = {'acc_id': [1001,1003,1002,1001,1001,1002,1003,1001,1003],
        'txn_value': [25.00, 30.00, 100.00, 250.00, 98.50, 23.50, 45.00, 60.00, 150.15],
        'txn_date': ['2025-01-07', '2025-01-03', '2025-01-09', '2025-01-02', '2025-01-01', '2025-01-04', '2025-01-08', '2025-01-11', '2025-01-03']}
df = pd.DataFrame(txn_data)
plt.figure(figsize=(6, 4))
sns.boxplot(y='txn_value', x='acc_id', data=df) ④
plt.title("Transactions by account")
plt.xlabel("Account")
plt.ylabel("Transaction value")
plt.show()
```

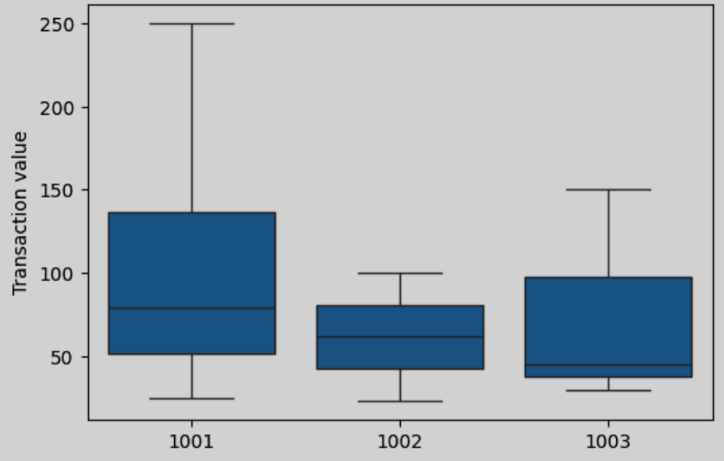
① Явное указание интерпретатора IPython.

② Специальная функция IPython для отображения графика непосредственно в блокноте.

③ Перед запуском заметки импортируемые модули должны быть установлены с помощью `pip install <module_name>`.

④ Генерация графика с помощью библиотеки `seaborn`.

Пример графика приведен ниже.



Визуализация с помощью Seaborn

Plotly

Чтобы использовать Plotly для визуализации данных, установите следующие pip-модули в интерпретатор Python, который используется сервисом Zeppelin.

```
$ /opt/python3.10/bin/python3 -m pip install plotly
$ /opt/python3.10/bin/python3 -m pip install nbformat
```

Код заметки:

```
%python.ipython ①
%matplotlib inline ②

import plotly.express as px ③
import matplotlib.pyplot as plt ③
import pandas as pd ③

txn_data = {'acc_id': [1001,1003,1002,1001,1001,1002,1003,1001,1003],
        'txn_value': [25.00, 30.00, 100.00, 250.00, 98.50, 23.50, 45.00, 60.00, 150.15],
        'txn_date': ['2025-01-02', '2025-01-03', '2025-01-04', '2025-01-01', '2025-01-01', '2025-01-05', '2025-01-03', '2025-01-06', '2025-01-05']}
df = pd.DataFrame(txn_data)
fig = px.scatter(df, ④
                x='txn_date',
                y='txn_value',
                color='acc_id',
                size="txn_value",
                title="Dummy Plotly scatter chart")
fig.update_layout(width=800, height=600)
fig.show()
```

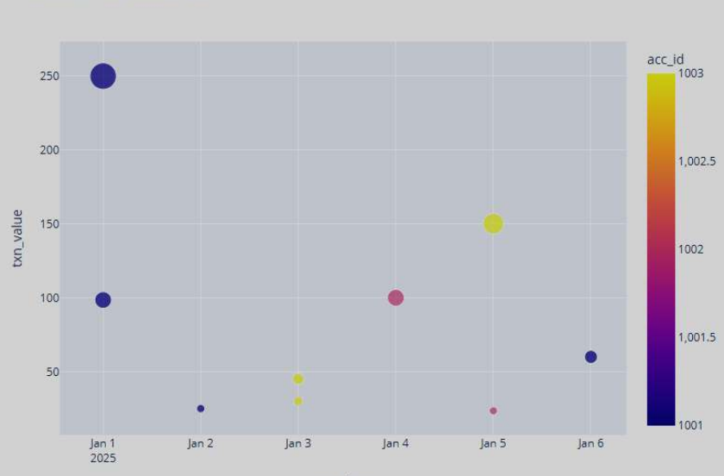
① Явное указание интерпретатора IPython.

② Специальная функция IPython для отображения графика непосредственно в блокноте.

③ Перед запуском заметки импортируемые модули должны быть установлены с помощью `pip install <module_name>`.

④ Генерация графика с помощью библиотеки `Plotly`.

Пример графика приведен ниже.



Визуализация с помощью Plotly

SQL для Pandas DataFrames (%python.sql)

С помощью Zeppelin-интерпретатора `%python.sql` можно использовать SQL для запросов к Pandas DataFrames и визуализации их содержимого. Данный интерпретатор требует установки следующих pip-модулей:

- `pandas`
- `pandasql`

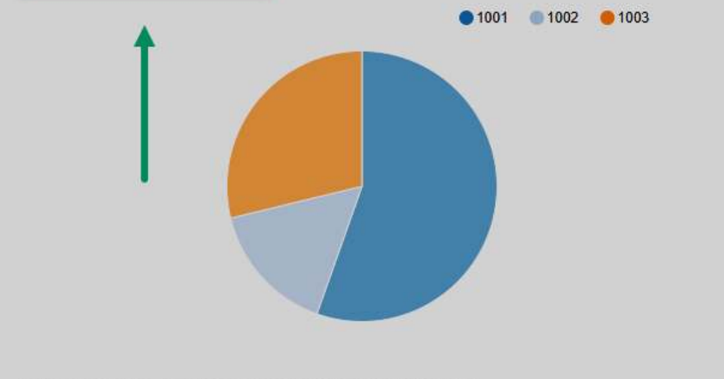
Пример заметки:

```
%python.sql ①
SELECT * FROM df WHERE txn_value > 100; ②
```

① Указание интерпретатора для использования SQL поверх Pandas DataFrames.

② DataFrame `df` должен быть создан до выполнения запроса (в другом параграфе).

Использование интерпретатора `%spark.sql` активирует встроенную визуализацию Zeppelin. Можно переключаться между виджетами для отображения данных на круговой диаграмме, линейном графике и так далее.



```
%python.sql
SELECT * FROM df WHERE txn_value > 10;
```

Took 0 sec. Last updated by anonymous at February 14 2025, 5:56:27 PM. (outdated)

Визуализация результатов запроса

Логирование в Zeppelin

Константин Алпашкин

Содержание

Назад к содержанию

- Расположение и формат лог-файлов
- Типы логов Zeppelin
- Настройка логирования в ADCM
  - Стандартная конфигурация log4j.properties
- Настройка логирования через REST API
- Типовые задачи

В процессе работы Zeppelin Server генерирует текстовые лог-файлы, которые могут быть полезными для анализа причин различных ошибок, возникающих при работе с интерпретаторами, заметками и так далее. Под капотом Zeppelin использует фреймворк Log4j для ведения логов. Для конфигурации процессов логирования в Zeppelin используйте настройки ***Custom log4j.properties*** в ADCM. Более подробная информация об этих настройках доступна в разделе Настройка логирования с помощью ADCM.

Расположение и формат лог-файлов

По умолчанию лог-файлы расположены в директории */var/log/zeppelin* на хосте, где установлен компонент Zeppelin Server. Чтобы увидеть логи, выполните следующее:

1. Присоединитесь к хосту с Zeppelin Server (по SSH либо другим способом) и выполните команду:

```
$ ls -la /var/log/zeppelin
```

Результат выглядит следующим образом:

```
total 1768
drwxr-xr-x.  2 zeppelin zeppelin   4096 Dec  8 10:22 .
drwxr-xr-x. 19 root     root       4096 Dec  8 06:10 ..
-rw-r--r--.  1 root     root        555 Dec  7 06:54 install-interpreter.log
-rw-r--r--.  1 root     root       1665 Dec  6 16:13 install-interpreter.log.2021-12-06
-rw-r--r--.  1 zeppelin zeppelin   2656 Dec  8 10:30 zeppelin-interpreter-jdbc-zeppelin-bds-adh-2.ru-central1.internal.log
-rw-r--r--.  1 zeppelin zeppelin   5918 Dec  6 12:24 zeppelin-interpreter-jdbc-zeppelin-bds-adh-2.ru-central1.internal.log.2021-
12-06
-rw-r--r--.  1 zeppelin zeppelin  82200 Dec  7 18:15 zeppelin-interpreter-jdbc-zeppelin-bds-adh-2.ru-central1.internal.log.2021-
12-07
-rw-r--r--.  1 zeppelin zeppelin   2376 Dec  7 17:27 zeppelin-interpreter-phoenix2-zeppelin-bds-adh-2.ru-central1.internal.log
-rw-r--r--.  1 zeppelin zeppelin   4638 Dec  8 10:31 zeppelin-interpreter-phoenix-zeppelin-bds-adh-2.ru-central1.internal.log
-rw-r--r--.  1 zeppelin zeppelin 112258 Dec  6 16:11 zeppelin-interpreter-phoenix-zeppelin-bds-adh-2.ru-central1.internal.log.
2021-12-06
-rw-r--r--.  1 zeppelin zeppelin  17404 Dec  7 17:27 zeppelin-interpreter-phoenix-zeppelin-bds-adh-2.ru-central1.internal.log.
2021-12-07
-rw-r--r--.  1 zeppelin zeppelin   3951 Dec  7 18:15 zeppelin-interpreter-python-zeppelin-bds-adh-2.ru-central1.internal.log
-rw-r--r--.  1 zeppelin zeppelin   4134 Dec  6 16:01 zeppelin-interpreter-sh-zeppelin-bds-adh-2.ru-central1.internal.log
-rw-r--r--.  1 zeppelin zeppelin  61835 Dec  8 12:09 zeppelin-zeppelin-bds-adh-2.ru-central1.internal.log
-rw-r--r--.  1 zeppelin zeppelin 722037 Dec  6 16:30 zeppelin-zeppelin-bds-adh-2.ru-central1.internal.log.2021-12-06
-rw-r--r--.  1 zeppelin zeppelin 662981 Dec  7 18:15 zeppelin-zeppelin-bds-adh-2.ru-central1.internal.log.2021-12-07
-rw-r--r--.  1 zeppelin zeppelin  78400 Dec  8 06:11 zeppelin-zeppelin-bds-adh-2.ru-central1.internal.out
```

ПРИМЕЧАНИЕ

Логи хранятся в локальной файловой системе, а не в HDFS.

2. Чтобы увидеть содержимое конкретного лог-файла, выполните следующее:

```
$ tail /var/log/zeppelin/<log_file_name>.log
```

Вывод имеет следующий вид:

```
 INFO [2021-12-08 10:31:02,382] ({pool-2-thread-4} RemoteInterpreterManagedProcess.java[start]:115) - Thrift server for callback
will start. Port: 38183
 INFO [2021-12-08 10:31:02,882] ({pool-2-thread-4} RemoteInterpreterManagedProcess.java[start]:190) - Run interpreter process
[/usr/lib/zeppelin/bin/interpreter.sh, -d, /usr/lib/zeppelin/interpreter/jdbc, -c, 10.92.6.9, -p, 38183, -r, :, -l,
/usr/lib/zeppelin/local-repo/phoenix, -g, phoenix]
 INFO [2021-12-08 10:31:04,138] ({pool-11-thread-1} RemoteInterpreterManagedProcess.java[callback]:123) - RemoteInterpreterServer
Registered: CallbackInfo(host:10.92.6.9, port:34481)
 INFO [2021-12-08 10:31:04,140] ({pool-2-thread-4} RemoteInterpreter.java[call]:168) - Create RemoteInterpreter org.apache.
zeppelin.jdbc.JDBCInterpreter
 INFO [2021-12-08 10:31:04,239] ({pool-2-thread-4} RemoteInterpreter.java[call]:142) - Open RemoteInterpreter org.apache.zeppelin.
jdbc.JDBCInterpreter
 INFO [2021-12-08 10:31:04,240] ({pool-2-thread-4} RemoteInterpreter.java[pushAngularObjectRegistryToRemote]:436) - Push local
angular object registry from ZeppelinServer to remote interpreter group phoenix:shared_process
 INFO [2021-12-08 10:31:04,830] ({pool-2-thread-4} NotebookServer.java[afterStatusChange]:2314) - Job 20211207-082338_940138511
is finished successfully, status: FINISHED
 INFO [2021-12-08 10:31:04,836] ({pool-2-thread-4} VFSNotebookRepo.java[save]:196) - Saving note:2GS5T8MS7
 INFO [2021-12-08 10:31:04,838] ({pool-2-thread-4} SchedulerFactory.java[jobFinished]:120) - Job 20211207-082338_940138511
finished by scheduler org.apache.zeppelin.interpreter.remote.RemoteInterpreter-phoenix:shared_process-shared_session
 INFO [2021-12-08 12:09:00,355] ({qtp837764579-66} NotebookServer.java[onClose]:372) - Closed connection to 10.92.2.6 : 54643.
(1001) null
```

Типы логов Zeppelin

Существует несколько типов лог-файлов, генерируемых Zeppelin. Их описания приведены ниже.

Имя лог-файла	Атрибуты	Описание
zeppelin-zeppelin-<host_name>.log	`<host_name>` — имя хоста Zeppelin	Содержит информацию о базовых операциях с соединением, задачами, заметками, такими как открытие, закрытие, сохранение и другие
zeppelin-zeppelin-<host_name>.out	`<host_name>` — имя хоста Zeppelin	Содержит вывод выполненных операций, например ошибки, возвращаемые в веб-интерфейсе
zeppelin-interpreter-<interpreter_name>-zeppelin-<host_name>.log	▪ `<host_name>` — имя хоста Zeppelin. ▪ `<interpreter_name>` — имя интерпретатора.	Содержит информацию об указанном интерпретаторе
install-interpreter.log	—	Содержит данные об установке новых интерпретаторов

Настройка логирования в ADCM

С использованием Log4j настройка процессов логирования в Zeppelin сводится к модификации конфигурационных параметров в файле *log4j.properties*. Этот файл находится в директории */etc/zeppelin/conf* на хосте, где установлен компонент Zeppelin Server, и Zeppelin считывает его при старте. Однако, чтобы не редактировать данный файл вручную на каждом хосте, ADCM предоставляет удобные настройки для изменения конфигураций Log4j на всех хостах кластера ADH. Для этого:

1. В ADCM перейдите на страницу ***Clusters*** → ***<your_cluster_name>*** → ***Services*** → ***Zeppelin*** → ***Primary Configuration*** и включите опцию ***Show advanced***.
2. Выберите ***Custom log4j.properties*** для редактирования конфигурации Log4j. Конфигурация Log4j, указанная через ADCM UI, перезапишет содержимое файла */etc/zeppelin/conf/log4j.properties* при рестарте сервиса Zeppelin. Подробная информация о конфигурационных свойствах Log4j доступна в документации Log4j.
3. Сохраните конфигурацию сервиса Zeppelin.
4. Перезапустите сервис Zeppelin.

Стандартная конфигурация log4j.properties

Ниже приведен файл *log4j.properties*, используемый Zeppelin по умолчанию. Основные параметры подсвечены комментариями, чтобы помочь вам начать работу с конфигурацией Log4j.

```
#
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements.  See the NOTICE file
# distributed with this work for additional information
# regarding copyright ownership.  The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#
log4j.rootLogger = INFO, dailyfile ①

log4j.appender.stdout = org.apache.log4j.ConsoleAppender ②
log4j.appender.stdout.layout = org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%5p [%d] ({%t} %F[%M]:%L) - %m%n ⑤

log4j.appender.dailyfile.DatePattern=.yyyy-MM-dd
log4j.appender.dailyfile.Threshold = INFO
log4j.appender.dailyfile = org.apache.log4j.DailyRollingFileAppender ③
log4j.appender.dailyfile.File = ${zeppelin.log.file} ④
log4j.appender.dailyfile.layout = org.apache.log4j.PatternLayout
log4j.appender.dailyfile.layout.ConversionPattern=%5p [%d] ({%t} %F[%M]:%L) - %m%n ⑤
```

① Установка базового уровня логирования. События с меньшим уровнем не попадают в лог. Например, при использовании `DEBUG` генерируются более детальные и объемные логи, а при уровне `ERROR` Zeppelin будет записывать в лог только ошибки и более критичные события.

② Использование ConsoleAppender, который выводит события логов в STDOUT.

③ Использование DailyRollingFileAppender, который записывает события в файл, ротируемый ежедневно.

④ Путь к лог-файлу. По умолчанию переменная `${zeppelin.log.file}` указывает на директорию */var/log/zeppelin*.

⑤ Шаблон для преобразования объекта события в строку. Шаблон по умолчанию имеет следующую структуру:

```
[Уровень] [Дата] ([Поток] [Имя файла][Метод][Строка]) [Сообщение]
```

Настройка логирования через REST API

Сервис Zeppelin предоставляет REST API, с помощью которого вы можете взаимодействовать с Zeppelin, включая настройку логирования. Запрос к REST-эндпойнту для изменения уровня логирования выглядит следующим образом:

```
PUT http://<zeppelin-host>:<zeppelin-port>/api/log/level/<LOG_LEVEL>
```

Где:

- `<zeppelin-host>` — адрес хоста, на котором установлен компонент Zeppelin Server.
- `<zeppelin-port>` — номер порта, который используется для Zeppelin REST API. Номер порта можно изменить с помощью свойства `zeppelin.server.port` в ADCM. Порт по умолчанию: `8180`.
- `<LOG_LEVEL>` — желаемый уровень логирования. Допустимые уровни: `ALL`, `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, `FATAL` и `OFF`.

Пример:

```
$ curl -X PUT http://localhost:8180/api/log/level/WARN
```

Ответ от сервера:

```
{"status":"OK"}
```

Типовые задачи

Ниже описаны типовые кейсы, которые могут возникнуть при настройке логирования Zeppelin с помощью Log4j.

Изменение стандартного уровня логирования

Чтобы изменить корневой уровень логирования (по умолчанию `INFO`), используйте свойство `log4j.rootLogger=<LEVEL>,<APPENDER>`. Вы можете указать корневой уровень, который будет наследоваться другими логерами Log4j, либо установить необходимый уровень для конкретного логера, например `log4j.logger.org.apache.zeppelin.interpreter=<OTHER_LEVEL>`.

Изменение местоположения логов по умолчанию

По умолчанию логи Zeppelin хранятся в директории */var/log/zeppelin*. Чтобы изменить стандартную директорию, используйте свойство `log4j.appender.<appender-name>.File`, где `<appender-name>` — это имя аппендера Log4j, привязанного к корневому логеру.

Добавление кастомного логера

В следующем примере используется кастомный логер (Logger), который записывает логи в отдельную директорию. Также пример показывает генерацию лог-событий в коде интерпретатора.

1. Добавьте следующие конфигурационные свойства в раздел ***Custom log4j.properties*** в ADCM. Основные свойства выделены комментариями.

```
# ...
# New logger definition:
log4j.logger.MyCustomLogger=DEBUG, mycustomappender ①
log4j.appender.mycustomappender=org.apache.log4j.RollingFileAppender ②
log4j.appender.mycustomappender.File = /var/log/zeppelin/customlogger/test.log ③
log4j.appender.mycustomappender.rollingPolicy.FileNamePattern=/var/log/zeppelin/customlogger/test.%i.log ④
log4j.appender.mycustomappender.layout=org.apache.log4j.PatternLayout
log4j.appender.mycustomappender.layout.conversionPattern=[%p] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{10} - %m %n
log4j.appender.mycustomappender.MaxFileSize=10MB ⑤
log4j.appender.mycustomappender.MaxBackupIndex=10 ⑥
{% endraw %}
```

① Объявление нового логера и аппендера.

② Использование RollingFileAppender, который ротирует файлы логов при достижении определенного размера.

③ Путь к лог-файлу, генерируемому пользовательским логером.

④ Шаблон, используемый для именования лог-файлов при выполнении переноса (rollover).

⑤ Максимальный размер лог-файла для выполнения переноса.

⑥ Максимальное количество лог-файлов, хранимых в директории.

2. Сохраните конфигурацию и перезапустите сервис Zeppelin.
3. Запустите тестовую заметку Spark, чтобы сгенерировать записи логов. Для этого создайте новую заметку Spark с использованием Zeppelin UI и выполните в ней следующий код:

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

val logger: Logger = LoggerFactory.getLogger("MyCustomLogger");
logger.info("Test info message");
```

4. Проверьте логи, сгенерированные кастомным логером. Для этого подключитесь к хосту с компонентом Zeppelin Server и выполните команду:

```
$ ls -l /var/log/zeppelin/customlogger
```

Вывод:

```
-rw-r--r-- 1 zeppelin zeppelin 114 Nov 25 13:14 test.log
```

Файл *test.log* содержит единственную запись, сгенерированную заметкой Spark:

```
[INFO] 2024-11-22 13:14:28.569 [FIFOScheduler-interpreter_453440904-Worker-1] MyCustomLogger - Test info message
```

Оптимизация производительности Zeppelin

Константин Алпашкин

Содержание

Назад к содержанию

- Память
- Режимы биндинга
- Оптимизации Spark
- Логирование
- Браузер
- Прочее

В этой статье описаны методы оптимизации работы сервиса Zeppelin, а также выделены основные факторы, влияющие на производительность Zeppelin.

Память

Основным фактором, влияющим на производительность сервиса Zeppelin, является объем оперативной памяти, доступной компоненту Zeppelin Server. Работу сервиса можно ускорить, выделив больше оперативной памяти JVM-процессу Zeppelin. Чтобы увеличить параметры `-Xms`/`-Xmx`, используйте настройки ***Zeppelin Server Heap Memory*** в ADCM (***Clusters → <clusterName> → Services → Zeppelin → Primary configuration***) и перезапустите сервис.

При расчете оптимального объема памяти, необходимой для конкретного кластера, стоит учитывать следующие показатели:

- Количество пользователей, которые будут одновременно подключаться к серверу Zeppelin.
- Режимы биндинга, в которых работают отдельные интерпретаторы.

Режимы биндинга

Интерпретаторы Zeppelin запускаются в виде отдельных JVM-процессов, которые выполняют код заметок и возвращают результаты, общаясь с демоном Zeppelin. Zeppelin поддерживает несколько режимов биндинга интерпретаторов (interpreter binding modes), которые определяют способы запуска процессов для каждого интерпретатора. Используя различные комбинации режимов биндинга, можно оптимизировать потребляемую память и выбрать подход, который лучше всего подходит для конкретного кластера. Краткое описание этих режимов представлено ниже.

- **Shared**.

  В этом режиме все заметки интерпретатора выполняются в одном JVM-процессе с общей сессией. Такой подход обеспечивает меньшее потребление памяти в многопользовательской среде, а также предоставляет возможности взаимодействия между заметками, например, доступ к переменным, созданным в других заметках. Последнее может быть особенно полезно при совместном использовании "тяжелых" объектов, таких как `SparkContext`, `SparkSession` и так далее.

  Чтобы увеличить объем оперативной памяти, выделяемой сервером Zeppelin каждому процессу интерпретатора, используйте параметр `ZEPPELIN_INTP_MEM` в настройках ***Custom zeppelin-env.sh*** в ADCM (***Clusters → <clusterName> → Services → Zeppelin → Primary configuration*** и активируйте ***Show advanced***). Например:

  ```
  ZEPPELIN_INTP_MEM="-Xms2048m -Xmx4096m -XX:MaxMetaspaceSize=1024m"
  ```

- **Scoped**.

  В этом режиме для каждого типа интерпретатора Zeppelin создает одну JVM, а каждая заметка выполняется в своей выделенной сессии в зависимости от того, выбрана опция ***Per Note*** или ***Per User***.

- **Isolated**.

  В этом режиме для каждой заметки создается отдельный JVM-процесс интерпретатора (в конфигурации ***Per note***). Общее потребление памяти увеличивается, однако в таком режиме пользователи Zeppelin могут выполнять заметки в полностью изолированной среде, исключая возможность конфликтов, в том числе имен переменных. Кроме того, перезапуск интерпретатора одним пользователем не влияет на работу других.

  Для многопользовательской среды этот режим является наиболее затратным с точки зрения оперативной памяти, поэтому необходимо убедиться, что хосты ADH с компонентом Zeppelin Server могут поддерживать работу множества JVM для каждой заметки.

Оптимизации Spark

При работе с интерпретаторами Spark для ускорения Spark-задач следует использовать методы оптимизации, описанные в статье Оптимизация производительности в Spark. Для установки конфигурационных свойств Spark используйте поле ***Custom zeppelin-env.sh*** в ADCM (***Clusters → <clusterName> → Services → Zeppelin → Primary configuration*** и активируйте ***Show advanced***). Например:

```
export SPARK_SUBMIT_OPTIONS="--executor-memory 4G"
```

> **ПРИМЕЧАНИЕ**
>
> Первый запуск заметки Spark всегда медленнее, поскольку интерпретатору Spark необходимо создать объекты контекста и Application Master.

Логирование

Использование уровня логирования `DEBUG` в Zeppelin может замедлить отклик в UI, особенно в момент сохранения конфигурации интерпретатора при создании новой заметки. Уровень `DEBUG` следует использовать только для устранения неполадок.

Для изменения уровня логирования в Zeppelin используйте настройки ***Custom log4j.properties*** в ADCM (***Clusters → <clusterName> → Services → Zeppelin → Primary configuration*** и активируйте ***Show advanced***).

Браузер

Используйте современный веб-браузер, способный загружать большие заметки (10 МБ и более), визуализировать десятки и сотни мегабайт выходных данных, поддерживать современные библиотеки визуализации и так далее. Например, в старых версиях IE сообщалось о проблемах с аутентификацией.

Удаление объемного вывода параграфа снижает нагрузку на браузер/JVM, так как результаты последнего выполнения заметки сохраняются во внутренних файлах Zeppelin.

Прочее

- Периодически перезапускайте интерпретаторы. Длительное функционирование интерпретатора без перезапуска может приводить к замедлению работы.
- Ограничьте количество строк, возвращаемых SQL-запросами. Например, для интерпретаторов Spark количество строк в результатах можно регулировать с помощью свойства конфигурации интерпретатора `zeppelin.spark.maxResult`.
- Используйте префикс `%table` для более эффективной визуализации больших наборов результатов.

Управление сервисом Zeppelin через ADCM

Константин Алпашкин

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом Zeppelin. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса Zeppelin, перечислены в таблице ниже.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты Zeppelin
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются

Подключение к ZooKeeper через CLI

Альберт Багдасарян

Назад к содержанию

Простейшим способом начать работать с ZooKeeper является использование командной строки (Command Line Interface, CLI). Консоль ZooKeeper CLI доступна на каждом сервере кластера ZooKeeper сразу после его установки. Откройте консоль с помощью скрипта *zkCli.sh*, расположенного в каталоге */bin*:

```
$ sudo /usr/lib/zookeeper/bin/zkCli.sh
```

Вывод на экран

```
2021-12-15 11:04:55,901 [myid:] - INFO  [main:Environment@100] - Client environment:zookeeper.version=3.4.14-
f42ef4b166f1e5c63117c75aed651589a0cce1b2, built on 08/06/2019 14:49 GMT
2021-12-15 11:04:55,903 [myid:] - INFO  [main:Environment@100] - Client environment:host.name=bds-adh-1.ru-central1.internal
2021-12-15 11:04:55,904 [myid:] - INFO  [main:Environment@100] - Client environment:java.version=1.8.0_312
2021-12-15 11:04:55,905 [myid:] - INFO  [main:Environment@100] - Client environment:java.vendor=Red Hat, Inc.
2021-12-15 11:04:55,906 [myid:] - INFO  [main:Environment@100] - Client environment:java.home=/usr/lib/jvm/java-1.8.0-openjdk-
1.8.0.312.b07-1.el7_9.x86_64/jre
2021-12-15 11:04:55,906 [myid:] - INFO  [main:Environment@100] - Client environment:java.class.
path=/usr/lib/zookeeper/bin/../zookeeper-
server/target/classes:/usr/lib/zookeeper/bin/../build/classes:/usr/lib/zookeeper/bin/../zookeeper-server/target/lib/*.
jar:/usr/lib/zookeeper/bin/../build/lib/*.jar:/usr/lib/zookeeper/bin/../lib/slf4j-log4j12-1.7.25.
jar:/usr/lib/zookeeper/bin/../lib/slf4j-api-1.7.25.jar:/usr/lib/zookeeper/bin/../lib/netty-3.10.6.Final.
jar:/usr/lib/zookeeper/bin/../lib/log4j-1.2.17.jar:/usr/lib/zookeeper/bin/../lib/jline-0.9.94.
jar:/usr/lib/zookeeper/bin/../lib/audience-annotations-0.5.0.jar:/usr/lib/zookeeper/bin/../zookeeper-3.4.14.
jar:/usr/lib/zookeeper/bin/../zookeeper-server/src/main/resources/lib/*.jar:/usr/lib/zookeeper/bin/../conf:
2021-12-15 11:04:55,906 [myid:] - INFO  [main:Environment@100] - Client environment:java.library.
path=/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib
2021-12-15 11:04:55,906 [myid:] - INFO  [main:Environment@100] - Client environment:java.io.tmpdir=/tmp
2021-12-15 11:04:55,906 [myid:] - INFO  [main:Environment@100] - Client environment:java.compiler=<NA>
2021-12-15 11:04:55,906 [myid:] - INFO  [main:Environment@100] - Client environment:os.name=Linux
2021-12-15 11:04:55,906 [myid:] - INFO  [main:Environment@100] - Client environment:os.arch=amd64
2021-12-15 11:04:55,906 [myid:] - INFO  [main:Environment@100] - Client environment:os.version=3.10.0-1160.11.1.el7.x86_64
2021-12-15 11:04:55,907 [myid:] - INFO  [main:Environment@100] - Client environment:user.name=root
2021-12-15 11:04:55,907 [myid:] - INFO  [main:Environment@100] - Client environment:user.home=/root
2021-12-15 11:04:55,907 [myid:] - INFO  [main:Environment@100] - Client environment:user.dir=/home/dasha
2021-12-15 11:04:55,908 [myid:] - INFO  [main:ZooKeeper@442] - Initiating client connection, connectString=localhost:2181
sessionTimeout=30000 watcher=org.apache.zookeeper.ZooKeeperMain$MyWatcher@4b85612c
Welcome to ZooKeeper!
2021-12-15 11:04:55,928 [myid:] - INFO  [main-SendThread(localhost:2181):ClientCnxn$SendThread@1025] - Opening socket
connection to server localhost/127.0.0.1:2181. Will not attempt to authenticate using SASL (unknown error)
JLine support is enabled
2021-12-15 11:04:55,973 [myid:] - INFO  [main-SendThread(localhost:2181):ClientCnxn$SendThread@879] - Socket connection
established to localhost/127.0.0.1:2181, initiating session
2021-12-15 11:04:55,981 [myid:] - INFO  [main-SendThread(localhost:2181):ClientCnxn$SendThread@1299] - Session establishment
complete on server localhost/127.0.0.1:2181, sessionid = 0x100005560090006, negotiated timeout = 30000

WATCHER::

WatchedEvent state:SyncConnected type:None path:null
[zk: localhost:2181(CONNECTED) 0]
```

Приглашение ZooKeeper CLI имеет вид `[zk: localhost:2181(CONNECTED) 0]`. После него можно вводить команды.

Для выхода из ZooKeeper CLI используйте следующую команду:

```
quit
```

Подключение к ZooKeeper через API

Елена Костюченко

Содержание

Назад к содержанию

Клиент ZooKeeper
Java
C

ZooKeeper предоставляет API с байндингами для Java и C. API для других языков создаются и поддерживаются сообществом ZooKeeper.

API ZooKeeper позволяет приложениям подключаться к серверу, выполнять команды, манипулировать данными и координировать задачи.

Клиент ZooKeeper

Приложение, которое взаимодействует с сервером ZooKeeper, называется клиентом ZooKeeper.

Предпочтительный алгоритм взаимодействия клиента с сервером ZooKeeper выглядит следующим образом:

1. Создание соединения с сервером. При подключении сервер назначит ID сессии для клиента.
2. Отправка heartbeat на сервер. Если клиент не будет активен какое-то время, то срок действия ID сессии закончится и клиенту придется подключаться повторно.
3. Выполнение необходимых операций на сервере, например, получение или установка znodes.
4. Отключение от сервера. Если клиент неактивен в течение длительного времени, сервер автоматически отключит его.

Узнать больше о разработке для Zookeeper можно в статье ZooKeeper Programmer's Guide.

Java

Чтобы подключиться к серверу ZooKeeper с помощью Java API, необходимо настроить среду разработки и установить зависимости. Приведенный ниже пример кода создает соединение с сервером ZooKeeper, выводит подтверждение при успешном подключении и возвращает сообщение об ошибке, если подключиться не удалось.

Для выполнения примера используются следующие инструменты разработки: текстовый редактор Vi, виртуальная машина с CentOS 7 и компилятор Maven. При использовании других инструментов будет необходимо изменить некоторые шаги.

1. Установите Maven.
2. Напишите класс для создания соединения:

```
package my.example.zookeeper;

import java.io.IOException;
import java.util.concurrent.CountDownLatch;

import org.apache.zookeeper.KeeperException;
import org.apache.zookeeper.WatchedEvent;
import org.apache.zookeeper.Watcher;
import org.apache.zookeeper.Watcher.Event.KeeperState;
import org.apache.zookeeper.ZooKeeper;
import org.apache.zookeeper.AsyncCallback.StatCallback;
import org.apache.zookeeper.KeeperException.Code;
import org.apache.zookeeper.data.Stat;

public class Main {
   private class MyWatcher implements Watcher {
      public void process(WatchedEvent we) {
         if (we.getState() == KeeperState.SyncConnected) {
            connectedSignal.countDown();
         }
      }
   }

   private ZooKeeper zoo;
   final CountDownLatch connectedSignal = new CountDownLatch(1);

   public ZooKeeper connect(String host) throws IOException,InterruptedException {

      zoo = new ZooKeeper(host,2181,new MyWatcher());

      connectedSignal.await();
      return zoo;
   }

   public void close() throws InterruptedException {
      zoo.close();
   }

   public static void main(String[] args){
      try {
         Main app = new Main();
         app.connect("127.0.0.1");
         app.close();
      } catch (Exception e) {

      }
   }
}
```

3. Создайте файл *pom.xml* для сборки проекта с помощью Maven.

Пример файла pom.xml

```
<project>
        <modelVersion>4.0.0</modelVersion>
        <groupId>my.example.zookeeper</groupId>
        <artifactId>zoo-connect</artifactId>
        <version>0.1.0-SNAPSHOT</version>
        <packaging>jar</packaging>
  <properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
  <dependencies>
        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-hdfs</artifactId>
            <version>2.7.1</version>
        </dependency>

        <dependency>
            <groupId>org.apache.zookeeper</groupId>
            <artifactId>zookeeper</artifactId>
            <version>3.4.11</version>
        </dependency>

        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-common</artifactId>
            <version>2.7.1</version>
        </dependency>

        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-core</artifactId>
            <version>1.2.1</version>
        </dependency>

        <dependency>
            <groupId>org.apache.hadoop</groupId>
            <artifactId>hadoop-client</artifactId>
            <version>2.7.1</version>
        </dependency>
  </dependencies>
<build>
    <plugins>
      <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-assembly-plugin</artifactId>
            <version>3.1.0</version>
            <configuration>
              <archive>
                <manifest>
                  <addClasspath>true</addClasspath>
                  <mainClass>my.example.zookeeper.Main</mainClass>
                </manifest>
              </archive>
                <descriptorRefs>
                    <descriptorRef>jar-with-dependencies</descriptorRef>
                </descriptorRefs>
            </configuration>
            <executions>
                <execution>
                    <id>assemble-all</id>
                    <phase>package</phase>
                    <goals>
                        <goal>single</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
      <plugin>
        <!-- Build an executable JAR -->
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-jar-plugin</artifactId>
        <version>3.1.0</version>
        <configuration>
          <archive>
            <manifest>
              <addClasspath>true</addClasspath>
              <mainClass>my.example.zookeeper.Main</mainClass>
            </manifest>
             <addClasspath>true</addClasspath>
              <mainClass>my.example.zookeeper.Main</mainClass>
            </manifest>
          </archive>
        </configuration>
      </plugin>
    </plugins>
  </build>

</project>
```

4. Соберите проект:

```
$ mvn package
```

5. Запустите Java-класс:

```
$ java -jar <java-class>.jar
```

Здесь `<java-class>` — имя Java-программы для подключения к HDFS.

В консоль будет выведена информация об успешном подключении:

```
23/12/14 19:43:34 INFO zookeeper.ZooKeeper: Initiating client connection, connectString=127.0.0.1 sessionTimeout=2181 watcher=my.example.zookeeper.Main$MyWatcher@433c675d
23/12/14 19:43:34 INFO zookeeper.ClientCnxn: Opening socket connection to server 127.0.0.1/127.0.0.1:2181. Will not attempt to authenticate using SASL (unknown error)
23/12/14 19:43:34 INFO zookeeper.ClientCnxn: Socket connection established to 127.0.0.1/127.0.0.1:2181, initiating session
23/12/14 19:43:34 INFO zookeeper.ClientCnxn: Session establishment complete on server 127.0.0.1/127.0.0.1:2181, sessionid = 0x1000000cebe0004, negotiated timeout = 4000
23/12/14 19:43:34 INFO zookeeper.ZooKeeper: Session: 0x1000000cebe0004 closed
```

C

Чтобы подключиться к серверу ZooKeeper с помощью C API, необходимо настроить среду разработки и установить зависимости. Приведенный ниже пример кода создает соединение с сервером ZooKeeper, выводит подтверждение при успешном подключении и возвращает сообщение об ошибке, если подключиться не удалось.

В примере используются следующие инструменты разработки: текстовый редактор Vi, виртуальная машина с Ubuntu 23.10 и компилятор GCC. При использовании других инструментов будет необходимо изменить некоторые шаги.

Другой пример клиента на C можно найти в репозитории Apache ZooKeeper.

Прежде чем создавать клиент ZooKeeper на C, необходимо загрузить библиотеку zookeeper.h или собрать ее самостоятельно используя инструмент для сборки `ant`. Если вы создаете многопоточный клиент, скомпилируйте его с флагом компилятора `-DTHREADED`, чтобы включить многопоточную версию библиотеки, а затем прилинкуйте ее к библиотеке zookeeper_mt. Если вы создаете однопоточный клиент, прилинкуйте библиотеку zookeeper_st и не используйте дополнительный флаг при компиляции.

Чтобы создать пример клиента ZooKeeper на языке C:

1. Напишите программу для подключения к серверу. Например:

```
#include <stdio.h>
#include <stdlib.h>
#include <zookeeper/zookeeper.h>
#include <errno.h>

static int completed = 0;

void watcher(zhandle_t *zkH, int type, int state, const char *path, void *watcherCtx) {
    if (type == ZOO_SESSION_EVENT) {
        if (state == ZOO_CONNECTED_STATE) {
        } else if (state == ZOO_NOTCONNECTED_STATE ) {
        } else if (state == ZOO_EXPIRED_SESSION_STATE) {
            completed = 1;
            zookeeper_close(zkH);
        }
    }
}

void aget_strings_completion(int rc, const struct String_vector *strings, const void *data) {
    int i;
    if (strings) {
        for (i=0; i < strings->count; i++) {
            printf("\t%s\n", strings->data[i]);
        }
    }
    completed = 1;
}

int main(int argc, char *argv[]) {
    zoo_set_debug_level(ZOO_LOG_LEVEL_DEBUG);
    zhandle_t *handler = zookeeper_init("127.0.0.1:2181", watcher, 30000, NULL, NULL, 0);
    if (!handler) {
        return EXIT_FAILURE;
    }
    int rc = zoo_aget_children(handler, "/", 1, aget_strings_completion, NULL);
    if (rc) {
        fprintf(stderr, "Error %d\n", rc);
        return EXIT_FAILURE;
    }
    while (!completed) {
    }
    zookeeper_close(handler);
    return EXIT_SUCCESS;
}
```

2. Скомпилируйте код. Пример команды для компиляции:

```
$ gcc main.c -o <program-name> -lzookeeper_mt
```

Здесь `<program-name>` — название программы.

При настроенном выводе о работе клиента в консоль можно увидеть следующее:

```
2023-12-18 12:23:11,953:6904(0x7fa0e9d146c0):ZOO_INFO@check_events@2992: initiated connection to server 127.0.0.1:2181
2023-12-18 12:23:11,965:6904(0x7fa0e9d146c0):ZOO_INFO@finalize_session_establishment@2866: session establishment complete on server 127.0.0.1:2181, sessionId=0x1000000cebe0005, negotiated timeout=30000
2023-12-18 12:23:11,965:6904(0x7fa0e9d146c0):ZOO_DEBUG@finalize_session_establishment@2876: Calling a watcher for a ZOO_SESSION_EVENT and the state=ZOO_CONNECTED_STATE
2023-12-18 12:23:11,965:6904(0x7fa0e95136c0):ZOO_DEBUG@process_completions@3341: Calling a watcher for node [], type = -1 event=ZOO_SESSION_EVENT
2023-12-18 12:23:11,966:6904(0x7fa0e9d146c0):ZOO_DEBUG@zookeeper_process@3503: Queueing asynchronous response
2023-12-18 12:23:11,966:6904(0x7fa0e95136c0):ZOO_DEBUG@deserialize_response@3233: Calling COMPLETION_STRINGLIST for xid=0x658039b0 failed=0 rc=0
        zookeeper
        arenadata
        hadoop-ha
2023-12-18 12:23:11,966:6904(0x7fa0e95136c0):ZOO_DEBUG@do_completion@483: completion thread terminated
2023-12-18 12:23:11,966:6904(0x7fa0e9d146c0):ZOO_DEBUG@do_io@460: IO thread terminated
2023-12-18 12:23:11,966:6904(0x1a373c0):ZOO_INFO@zookeeper_close@3854: Closing zookeeper sessionId=0x1000000cebe0005 to 127.0.0.1:2181
```

Работа с znodes

Альберт Багдасарян

Содержание

Назад к содержанию

Шаг 1. Создание znode
Шаг 2. Получение данных znode
Шаг 3. Запись данных в znode
Шаг 4. Создание дочерних узлов znode
Шаг 5. Получение списка дочерних узлов для определенного znode
Шаг 6. Получение метаданных znode
Шаг 7. Наблюдение за изменениями в znode
Шаг 8. Удаление znode

ПРИМЕЧАНИЕ

Перед выполнением последующих команд рекомендуем ознакомиться с информацией о znodes и watches.

Шаг 1. Создание znode

Для создания узла znode используйте команду `create`, после которой определите полный путь к новому znode, а затем введите данные для сохранения в этом znode. Существуют следующие типы znode (при их создании могут потребоваться дополнительные флаги в команде `create`):

- **Постоянный (persistent)**. Команда `create` не требует дополнительного флага (тип по умолчанию):

```
create /z_persistent "Test configuration"
```

Вывод на экран:

```
Created /z_persistent
```

- **Последовательный (sequential)**. Используйте флаг `-s` после ключевого слова `create`:

```
create -s /z_sequential "Test configuration 2"
```

ПРИМЕЧАНИЕ

ZooKeeper добавляет 10-значный постфикс к заданному названию znode. Это изменение гарантирует уникальность названия последовательного znode.

Вывод на экран:

```
Created /z_sequential0000000007
```

- **Эфемерный (ephemeral)**. Используйте флаг `-e` после ключевого слова `create`:

```
create -e /z_ephemeral "Test configuration 3"
```

Вывод на экран:

```
Created /z_ephemeral
```

Шаг 2. Получение данных znode

Для получения данных от заданного znode воспользуйтесь командой `get` и укажите полный путь к znode:

```
get /z_persistent
```

Вывод на экран содержит не только данные, хранимые в указанном znode, но также его метаданные из структуры stat:

```
Test configuration
cZxid = 0x2000000019
ctime = Wed Dec 15 11:15:27 UTC 2021
mZxid = 0x2000000019
mtime = Wed Dec 15 11:15:27 UTC 2021
pZxid = 0x2000000019
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 18
numChildren = 0
```

Ниже приведены примеры получения данных от двух других znodes.

ВАЖНО

Применяя команду `get` к последовательным znodes, вводите их полные названия, включая 10-значные названия, которые ZooKeeper назначил автоматически. Это правило также действует для команд set и rmr.

```
[zk: localhost:2181(CONNECTED) 7]get /z_sequential0000000007
Test configuration 2
cZxid = 0x200000001c
ctime = Wed Dec 15 11:17:47 UTC 2021
mZxid = 0x200000001c
mtime = Wed Dec 15 11:17:47 UTC 2021
pZxid = 0x200000001c
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 20
numChildren = 0
[zk: localhost:2181(CONNECTED) 8]get /z_ephemeral
Test configuration 3
cZxid = 0x200000001d
ctime = Wed Dec 15 11:18:35 UTC 2021
mZxid = 0x200000001d
mtime = Wed Dec 15 11:18:35 UTC 2021
pZxid = 0x200000001d
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x100005560090006
dataLength = 20
numChildren = 0
```

ПРИМЕЧАНИЕ

Начиная с версии ZooKeeper 3.5 команда `get /znode` не выводит данные о znode для нод-контейнеров. В этом случае используйте команду `get -s /znode`.

Шаг 3. Запись данных в znode

Для обновления данных в заданном znode введите команду `set`, полный путь к этому znode и затем данные для сохранения:

```
set /z_persistent "The test configuration has been updated with the command set"
```

Пример вывода:

```
cZxid = 0x2000000019
ctime = Wed Dec 15 11:15:27 UTC 2021
mZxid = 0x200000001e
mtime = Wed Dec 15 11:24:46 UTC 2021
pZxid = 0x2000000019
cversion = 0
dataVersion = 1
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 60
numChildren = 0
```

После этого команда `get` вернет обновленные данные из znode:

```
get /z_persistent
```

Вывод на экран:

```
The test configuration has been updated with the command set
cZxid = 0x2000000019
ctime = Wed Dec 15 11:15:27 UTC 2021
mZxid = 0x200000001e
mtime = Wed Dec 15 11:24:46 UTC 2021
pZxid = 0x2000000020
cversion = 2
dataVersion = 1
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 60
numChildren = 2
```

Шаг 4. Создание дочерних узлов znode

Для добавления дочерних узлов к определенному znode используйте стандартную команду `create` и задайте путь к родительскому узлу вместо корневого пути `/`:

```
create /z_persistent/child1 "Test child 1"
```

Приведенные ниже примеры команд создают дочерние узлы в родительском узле `z_persistent`:

- Создайте первый дочерний узел:

```
create /z_persistent/child1 "Test child 1"
```

Вывод на экран:

```
Created /z_persistent/child1
```

- Создайте второй дочерний узел:

```
create /z_persistent/child2 "Test child 2"
```

Вывод на экран:

```
Created /z_persistent/child2
```

Шаг 5. Получение списка дочерних узлов для определенного znode

Чтобы получить список дочерних узлов для определенного znode, введите команду `ls` и после нее укажите полный путь к родительскому znode:

```
ls /z_persistent
```

Вывод на экран:

```
[child2, child1]
```

Для получения списка дочерних узлов корневого узла используйте путь `/`:

```
ls /
```

Полученный список:

```
[zookeeper, Arenadata.Hadoop-1.solr.server, z_sequential0000000007, hadoop-ha, z_ephemeral, hbase, z_persistent]
```

Кроме того, можно использовать команду `ls2` для получения не только списка дочерних узлов заданного znode, но также и его метаданных.

```
ls2 /
```

Вывод на экран:

```
[zookeeper, Arenadata.Hadoop-1.solr.server, hadoop-ha, z_ephemeral, z_sequential0000000007, hbase, z_persistent]
cZxid = 0x0
ctime = Thu Jan 01 00:00:00 UTC 1970
mZxid = 0x0
mtime = Thu Jan 01 00:00:00 UTC 1970
pZxid = 0x2000000030
cversion = 17
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 0
numChildren = 7
```

Шаг 6. Получение метаданных znode

Для получения метаданных заданного znode, входящих в структуру stat, введите команду `stat` и полный путь к этому znode. В отличие от команды `get`, вывод содержит метаданные, но не сами данные:

```
stat /z_persistent
```

Вывод на экран:

```
cZxid = 0x2000000019
ctime = Wed Dec 15 11:15:27 UTC 2021
mZxid = 0x200000001e
mtime = Wed Dec 15 11:24:46 UTC 2021
pZxid = 0x2000000020
cversion = 2
dataVersion = 1
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 60
numChildren = 2
[zk: localhost:2181(CONNECTED) 15] stat /z_sequential0000000007
cZxid = 0x200000001c
ctime = Wed Dec 15 11:17:47 UTC 2021
mZxid = 0x200000001c
mtime = Wed Dec 15 11:17:47 UTC 2021
pZxid = 0x200000001c
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 20
numChildren = 0
[zk: localhost:2181(CONNECTED) 16] stat /z_ephemeral
cZxid = 0x200000001d
ctime = Wed Dec 15 11:18:35 UTC 2021
mZxid = 0x200000001d
mtime = Wed Dec 15 11:18:35 UTC 2021
pZxid = 0x200000001d
cversion = 0
dataVersion = 0
aclVersion = 0
ephemeralOwner = 0x100005560090006
dataLength = 20
numChildren = 0
```

Шаг 7. Наблюдение за изменениями в znode

Средства наблюдения в ZooKeeper позволяют клиенту получать уведомления об изменениях в наблюдаемых узлах. Добавьте watch с помощью команды `get` и дополнительного аргумента, задающего название watcher.

РЕКОМЕНДАЦИЯ

Можно добавлять watch с помощью команд: `get`, `stat`, `ls` и `ls2`.

```
get /z_persistent 1
```

Вывод на экран такой же, как и при обычном вызове команды `get`:

```
The test configuration has been updated with the command set
cZxid = 0x2000000019
ctime = Wed Dec 15 11:15:27 UTC 2021
mZxid = 0x200000001e
mtime = Wed Dec 15 11:24:46 UTC 2021
pZxid = 0x2000000020
cversion = 2
dataVersion = 1
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 60
numChildren = 2
```

Теперь проверьте настройку, изменив данные в узле с помощью команды `set`:

```
set /z_persistent "Updated"
```

Вывод на экран содержит не только обычный вывод этой команды, но также уведомление, сообщающее клиенту о произошедшем изменении в налюдаемом znode:

```
WATCHER::

WatchedEvent state:SyncConnected type:NodeDataChanged path:/z_persistent
cZxid = 0x2000000019
ctime = Wed Dec 15 11:15:27 UTC 2021
mZxid = 0x2000000021
mtime = Wed Dec 15 11:37:02 UTC 2021
pZxid = 0x2000000020
cversion = 2
dataVersion = 2
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 7
numChildren = 2
```

Учитывайте, что каждый watch срабатывает только один раз. При следующем выполнении команды `set` уведомление не будет послано, если клиент не настроит новый watch перед выполнением этой команды.

```
[zk: localhost:2181(CONNECTED) 24] set /z_persistent "Updated 2"
cZxid = 0x2000000019
ctime = Wed Dec 15 11:15:27 UTC 2021
mZxid = 0x2000000022
mtime = Wed Dec 15 11:37:51 UTC 2021saying
pZxid = 0x2000000020
cversion = 2
dataVersion = 3
aclVersion = 0
ephemeralOwner = 0x0
dataLength = 9
numChildren = 2
```

Шаг 8. Удаление znode

Для окончательного удаления znode введите команду `rmr` и после нее путь к этому узлу.

```
rmr /z_persistent
```

Если теперь попробовать получить данные от znode, которого уже нет, или его дочерних узлов, вывод на экран будет свидетельствовать о его удалении:

- Проверка родительского узла:

```
get /z_persistent
```

Вывод на экран:

```
Node does not exist: /z_persistent
```

- Проверка дочернего узла:

```
get /z_persistent/child1
```

Вывод на экран:

```
Node does not exist: /z_persistent/child1
```

В завершение проверьте автоматическое удаление эфемерных узлов при закрытии соединения с клиентом. Для этого закройте текущий сеанс с помощью команды `quit` и начните новый сеанс с помощью скрипта *zkCli.sh*. После этого командой `ls /` проверьте список дочерних узлов корневого znode. Вы получите следующий результат:

- Постоянный znode `/z_persistent` удален, так как мы удалили его ранее.
- Последовательный znode `/z_sequential0000000007` доступен.
- Эфемерный znode `/z_ephemeral` удален автоматически.

```
ls /
```

Вывод на экран:

```
[zookeeper, Arenadata.Hadoop-1.solr.server, z_sequential0000000007, hadoop-ha, hbase]
```

Логирование в ZooKeeper

Елена Костюченко

Содержание

Назад к содержанию

Обзор
Поиск по логам
Уровни логирования
Настройки логирования

Обзор

ZooKeeper записывает выходные данные и события процессов файловой системы в текстовые логи. Эта информация может быть полезна при диагностике технических проблем в системе.

Каждый компонент ZooKeeper хранит логи локально, на хосте, на котором он установлен. Логи имеют расширение *.log* и расположены в директории */var/log/zookeeper/*. Здесь же находятся файлы *.out*, содержащие информацию о запуске компонентов.

Имена файлов с логами имеют следующий формат: `zookeeper-<пользователь>-<компонент>-<хост>.log`.

Здесь:

- `<пользователь>` — пользователь, запускающий ZooKeeper;
- `<компонент>` — имя компонента, например, сервер ZooKeeper;
- `<хост>` — FQDN хоста компонента.

ZooKeeper использует для логирования библиотеки SLF4J и Log4J.

Поиск по логам

Чтобы найти в логах интересующую вас информацию, например, сообщения об ошибках, подключитесь к хосту с компонентом, логи которого вы хотите просмотреть, и воспользуйтесь командой `grep`.

Например:

```
$ cat /var/log/zookeeper/zookeeper-zookeeper-cli-elenas-adh3.ru-central1.internal.log | grep -i -A3 -B1 error
```

Эта команда просматривает лог ZooKeeper CLI на хосте `elenas-adh3.ru-central1.internal` и ищет сообщения, содержащие слово `error`. Опция `i` позволяет игнорировать различия в регистре, а опции `-A3 -B1` добавляют к найденной строке с ошибкой одну строчку до нее и три строчки после.

Пример вывода команды:

```
2023-11-28 12:39:02,986 [myid:] - INFO  [main:ClientCnxn@1681] - zookeeper.request.timeout value is 0. feature enabled=
2023-11-28 12:39:02,991 [myid:elenas-adh3.ru-central1.internal:2181] - INFO  [main-SendThread(elenas-adh3.ru-central1.
internal:2181):ClientCnxn$SendThread@1125] - Opening socket connection to server elenas-adh3.ru-central1.internal/127.0.0.1:2181.
Will not attempt to authenticate using SASL (unknown error)
2023-11-28 12:39:03,041 [myid:elenas-adh3.ru-central1.internal:2181] - INFO  [main-SendThread(elenas-adh3.ru-central1.
internal:2181):ClientCnxn$SendThread@972] - Socket connection established, initiating session, client: /127.0.0.1:53166, server:
elenas-adh3.ru-central1.internal/127.0.0.1:2181
2023-11-28 12:39:03,065 [myid:elenas-adh3.ru-central1.internal:2181] - INFO  [main-SendThread(elenas-adh3.ru-central1.
internal:2181):ClientCnxn$SendThread@1415] - Session establishment complete on server elenas-adh3.ru-central1.internal/127.0.0.
1:2181, sessionid = 0x1000010f7e50000, negotiated timeout = 30000
2023-11-28 12:39:03,855 [myid:] - INFO  [main:Environment@109] - Client environment:zookeeper.version=3.5.10-
c4d50586fc8f6fdbdc95b05e9b21b49ec0bd8566, built on 11/03/2022 08:23 GM
```

Уровни логирования

ZooKeeper поддерживает следующие уровни логирования (от наименее информативного к наиболее информативному):

1. `FATAL` — сообщает, что операция не может быть выполнена и будет завершена.
2. `ERROR` — уведомляет, что программа работает неправильно или прекратила работу.
3. `WARN` — предупреждает о потенциальных проблемах. Это означает, что программа работает не по стандартному сценарию и в будущем могут возникнуть проблемы.
4. `INFO` — передает информацию о жизненном цикле или состоянии программы.
5. `DEBUG` — выводит отладочную информацию о внутренних состояниях программы.
6. `TRACE` — выводит сообщения, отслеживающие ход выполнения программы.

Logger Log4j также принимает значения уровня логирования: `OFF` — для отключения логирования, и `ALL` — для разрешения всех типов сообщений.

Включение одного уровня логирования включит этот уровень и все уровни до него. Например, если установлен уровень логирования `WARN`, то в логи будут попадать только `WARN`, `ERROR` и `FATAL`, но не `INFO`, `DEBUG` и `TRACE`.

Настройки логирования

Настроить параметры логирования в ZooKeeper можно в файле конфигурации *log4j.properties*, который расположен на каждом хосте ZooKeeper в директории *etc/zookeeper/conf/*.

Файл *log4j.properties* содержит общие настройки логирования, а также настройки компонентов логирования: logger и appender. Параметры, которые вы устанавливаете для logger-компонентов, влияют на то, что программы сообщают системе. Параметры, которые вы устанавливаете для appender-компонентов, влияют на то, какая информация от logger-компонентов попадает в логи.

Конфигурация *log4j.properties* начинается с параметров по умолчанию для root logger. Он обрабатывает все сообщения в ZooKeeper, поскольку это единственный обработчик сообщений. Настройки root logger по умолчанию: `INFO, CONSOLE, ROLLINGFILE`. Это означает, что сообщения ниже уровня `INFO` отбрасываются, а остальные сообщения попадают как в консоль (CONSOLE appender), так и в логи (ROLLINGFILE appender).

Appender-компоненты по умолчанию:

- `CONSOLE` — выводит сообщения в консоль (например, информацию о работе команды CLI);
- `ROLLINGFILE` — сохраняет сообщения в логи и периодически обновляет их.

Вы можете использовать класс `appenders layout`, чтобы изменить формат сообщений в логах. Кроме самого сообщения шаблон сообщений по умолчанию включает: уровень логирования, дату, информацию о выполнении команды и ее вызове.

Более подробно об архитектуре Log4j читайте в документации Apache Logging Services.

Файл ZooKeeper log4j.properties по умолчанию

```
# Define some default values that can be overridden by system properties
zookeeper.root.logger="{{ services.zookeeper.config['zookeeper_env_content']['ZOO_LOG4J_PROP'] }}"
zookeeper.console.threshold=INFO
zookeeper.log.dir="{{ services.zookeeper.config['zookeeper_env_content']['ZOO_LOG_DIR'] }}"
zookeeper.log.file=zookeeper.log
zookeeper.log.threshold=INFO
zookeeper.tracelog.dir="{{ services.zookeeper.config['zookeeper_env_content']['ZOO_LOG_DIR'] }}"
zookeeper.tracelog.file=zookeeper.log

#
# ZooKeeper Logging Configuration
#

# Format is "<default threshold> (, <appender>)+

# DEFAULT: console appender only
log4j.rootLogger=${zookeeper.root.logger}

# Example with rolling log file
#log4j.rootLogger=DEBUG, CONSOLE, ROLLINGFILE

# Example with rolling log file and tracing
#log4j.rootLogger=TRACE, CONSOLE, ROLLINGFILE, TRACEFILE

#
# Log INFO level and above messages to the console
#
log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender
log4j.appender.CONSOLE.Threshold=${zookeeper.console.threshold}
log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
log4j.appender.CONSOLE.layout.ConversionPattern=%d{ISO8601} [myid:%X{myid}] - %-5p [%t:%C{1}@%L] - %m%n

#
# Add ROLLINGFILE to rootLogger to get log file output
#    Log DEBUG level and above messages to a log file
log4j.appender.ROLLINGFILE=org.apache.log4j.RollingFileAppender
log4j.appender.ROLLINGFILE.Threshold=${zookeeper.log.threshold}
log4j.appender.ROLLINGFILE.File=${zookeeper.log.dir}/${zookeeper.log.file}

# Max log file size of 10MB
log4j.appender.ROLLINGFILE.MaxFileSize=10MB
# uncomment the next line to limit number of backup files
#log4j.appender.ROLLINGFILE.MaxBackupIndex=10

log4j.appender.ROLLINGFILE.layout=org.apache.log4j.PatternLayout
log4j.appender.ROLLINGFILE.layout.ConversionPattern=%d{ISO8601} [myid:%X{myid}] - %-5p [%t:%C{1}@%L] - %m%n


#
# Add TRACEFILE to rootLogger to get log file output
#    Log DEBUG level and above messages to a log file
log4j.appender.TRACEFILE=org.apache.log4j.FileAppender
log4j.appender.TRACEFILE.Threshold=TRACE
log4j.appender.TRACEFILE.File=${zookeeper.tracelog.dir}/${zookeeper.tracelog.file}

log4j.appender.TRACEFILE.layout=org.apache.log4j.PatternLayout
### Notice we are including log4j's NDC here (%x)
log4j.appender.TRACEFILE.layout.ConversionPattern=%d{ISO8601} [myid:%X{myid}] - %-5p [%t:%C{1}@%L][%x] - %m%n
```

Чтобы изменить *log4j.properties* на всех хостах одновременно, используйте ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на ZooKeeper.
3. Выберите ***Components*** и затем ***Zookeeper Server***.
4. Включите опцию ***Show Advanced*** и найдите поле ***log4j.properties template***.
5. Внесите необходимые изменения в файл по умолчанию.
6. Подтвердите изменения в конфигурации ZooKeeper, нажав ***Save***.
7. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

ПРИМЕЧАНИЕ

Файл *log4j.properties* содержит значения по умолчанию для параметров логирования. Они могут быть перезаписаны компонентами ZooKeeper при перезапуске. Чтобы этого избежать, можно установить значения параметров через переменные среды в файле *zookeeper-env.sh*.

Чтобы изменить настройки логирования в файле *zookeeper-env.sh* через ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на ZooKeeper.
3. Включите опцию ***Show Advanced*** и найдите поле ***zookeeper-env.sh***.
4. Выберите нужный параметр, например, ***ZOO_LOG4J_PROP***, и внесите изменения.
5. Подтвердите изменения в конфигурации ZooKeeper, нажав ***Save***.
6. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Настройка ZooKeeper

Елена Костюченко

Содержание

Назад к содержанию

Конфигурационные файлы
  - zoo.cfg
  - zookeeper-env.sh
  - log4j.properties

Изменение конфигурации через ADCM

Кворум другого кластера

Рекомендации

Конфигурационные файлы

У ZooKeeper есть три основных файла конфигурации, расположенные в директории *etc/zookeeper/conf/*. Их можно редактировать вручную в текстовом редакторе, подключившись к хосту с ZooKeeper, или через Arenadata Cluster Manager (ADCM).

При редактировании вручную убедитесь, что вы обновили файлы конфигурации на всех хостах с компонентами ZooKeeper.

> **ПРИМЕЧАНИЕ**
>
> ZooKeeper поддерживает динамическую реконфигурацию. Эта функция позволяет изменять конфигурацию без перезапуска сервера, но это может помешать работе ADCM.

В этой статье не рассматриваются дополнительные параметры конфигурации. Более подробно о конфигурации ZooKeeper читайте в документации ZooKeeper Administrator's Guide.

zoo.cfg

Файл конфигурации *zoo.cfg* содержит основные настройки ZooKeeper. Если ZooKeeper был установлен через ADCM, он уже содержит минимальные параметры конфигурации. Если ZooKeeper был установлен вручную, необходимо настроить следующие параметры:

- `clientPort` — порт для клиентских подключений к ZooKeeper. Для SSL-соединений настройте параметр `SecureClientPort`. `clientPort` содержит порт для обычных соединений, а `secureClientPort` содержит порт для SSL-соединений. Если указаны оба параметра, включится смешанный режим.
- `dataDir` — директория, где ZooKeeper хранит снимки базы данных и журнал транзакций.
- `tickTime` — длина одного тика в миллисекундах. Это основная единица времени, используемая в ZooKeeper для регулирования heartbeat и timeout.

Информация об основных свойствах файла *zoo.cfg* доступна в разделе ***Main*** настроек ZooKeeper. Информацию о других свойствах *zoo.cfg* смотрите в разделе ***zoo.cfg***.

zookeeper-env.sh

Файл *zookeeper-env.sh* используется для определения параметров конфигурации, связанных с операционной средой ZooKeeper, таких как `Java_HOME`.

Вы также можете настраивать параметры в других файлах конфигурации, установив переменные среды в *zookeeper-env.sh*.

Дополнительную информацию о содержимом *zookeeper-env.sh* можно получить в разделе ***zookeeper-env.sh*** настроек ZooKeeper.

log4j.properties

Файл конфигурации *log4j.properties* содержит настройки логирования. Подробнее о настройке логирования читайте в статье Логирование в ZooKeeper.

Изменение конфигурации через ADCM

Чтобы настроить ZooKeeper с помощью ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на ZooKeeper.
3. Найдите нужное свойство на странице конфигурации сервиса или странице ***Components***. Дополнительные свойства можно отобразить, выключив опцию ***Show advanced***.
4. Внесите необходимые изменения и нажмите ***Save***.
5. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Кворум другого кластера

Вы можете использовать кворум ZooKeeper из другого кластера для обслуживания сервисов в режиме высокой доступности.

Для этого укажите полные доменные имена (FQDN) внешних хостов ZooKeeper в конфигурации сервиса с помощью кастомных параметров в ADCM.

Например, чтобы интегрировать высокодоступный HBase с внешним кворумом ZooKeeper в ADCM:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и выберите ***HBase***.
3. Используя параметр ***Custom hbase-site.xml***, укажите `hbase.zookeeper.quorum` в качестве имени поля, а FQDN хостов кворума ZooKeeper — в качестве значения.
4. Подтвердите изменения в конфигурации, нажав ***Save***.
5. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.

Значение по умолчанию для параметра `hbase.zookeeper.quorum` будет заменено на новое на всех хостах.

Рекомендации

Распространенные ошибки в конфигурации ZooKeeper, а также некоторые дополнительные настройки, которые могут улучшить производительность кластера:

- Если с одним сервером ZooKeeper взаимодействует множество клиентов, использование опции `chroot` при подключении может упростить их работу. Эта опция позволяет выполнять команды, интерпретируя все пути относительно корня (аналогично команде unix `chroot`). Пример строки подключения: `127.0.0.1:4545/app/a`.
- Чтобы избежать предупреждений fsync в логах ZooKeeper, настройте выделенный диск для директории `dataLogDir`. Для этого добавьте параметр `dataLogDir` в файл *zoo.cfg*.
- Для повышения производительности поместите журнал транзакций (`dataDir`) на выделенное устройство. Если журнал транзакций находится на сервере с высокой нагрузкой, это отрицательно скажется на производительности сервера ZooKeeper.
- Поддерживайте согласованность списка серверов для клиентов и серверов ZooKeeper.

Управление сервисом ZooKeeper через ADCM

Елена Костюченко

Назад к содержанию

Пользовательский интерфейс ADCM позволяет выполнять действия, управляющие сервисом ZooKeeper. За информацией о том, как их запускать, обратитесь к статье Действия с сервисами ADH.

Действия, доступные для сервиса ZooKeeper, перечислены в таблице ниже.

Подробное описание сервиса и его компонентов приведено в статье ZooKeeper.

Действие	Описание
Add/Remove components	При выполнении этого действия открывается интерфейс распределения компонентов, где вы можете добавить, удалить или перераспределить компоненты ZooKeeper
Check	Выполняет проверку работоспособности сервиса и его отдельных компонентов
Start	Запускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то запуск сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются
Stop	Останавливает сервис
Remove	Удаляет сервис из кластера. Данное действие предназначено для удаления уже установленных сервисов. В то время как кнопка 🗑 используется для удаления сервисов, чьи компоненты еще не были распределены по узлам кластера
Restart	Перезапускает сервис. Для этого действия доступна опция ***Apply configs from ADCM***. Если выбрано значение `true`, то рестарт сервиса выполняется с учетом всех параметров, указанных в настройках этого сервиса в ADCM. В противном случае настройки сервиса в ADCM игнорируются. Сервис поддерживает функцию ***Rolling restart*** (включена по умолчанию), которая позволяет перезапускать компоненты ZooKeeper Server по одному, а не все сразу. Данная опция помогает избежать простоев сервиса и обеспечивает более гибкую конфигурацию перезапуска. Опция ***Rolling restart*** имеет следующие параметры: ▪ ***delay*** — тайм-аут в секундах между перезапусками отдельных компонентов ZooKeeper Server. ▪ ***max_failed_instances_number*** — максимально допустимое количество неудачных рестартов ZooKeeper Server. При достижении этого значения ADCM-действие ***Restart*** завершается с ошибкой. ▪ ***health_checks*** — указывает, нужно ли выполнять проверку компонентов ZooKeeper Server после рестарта.

Действия с кластером Arenadata Hyperwave

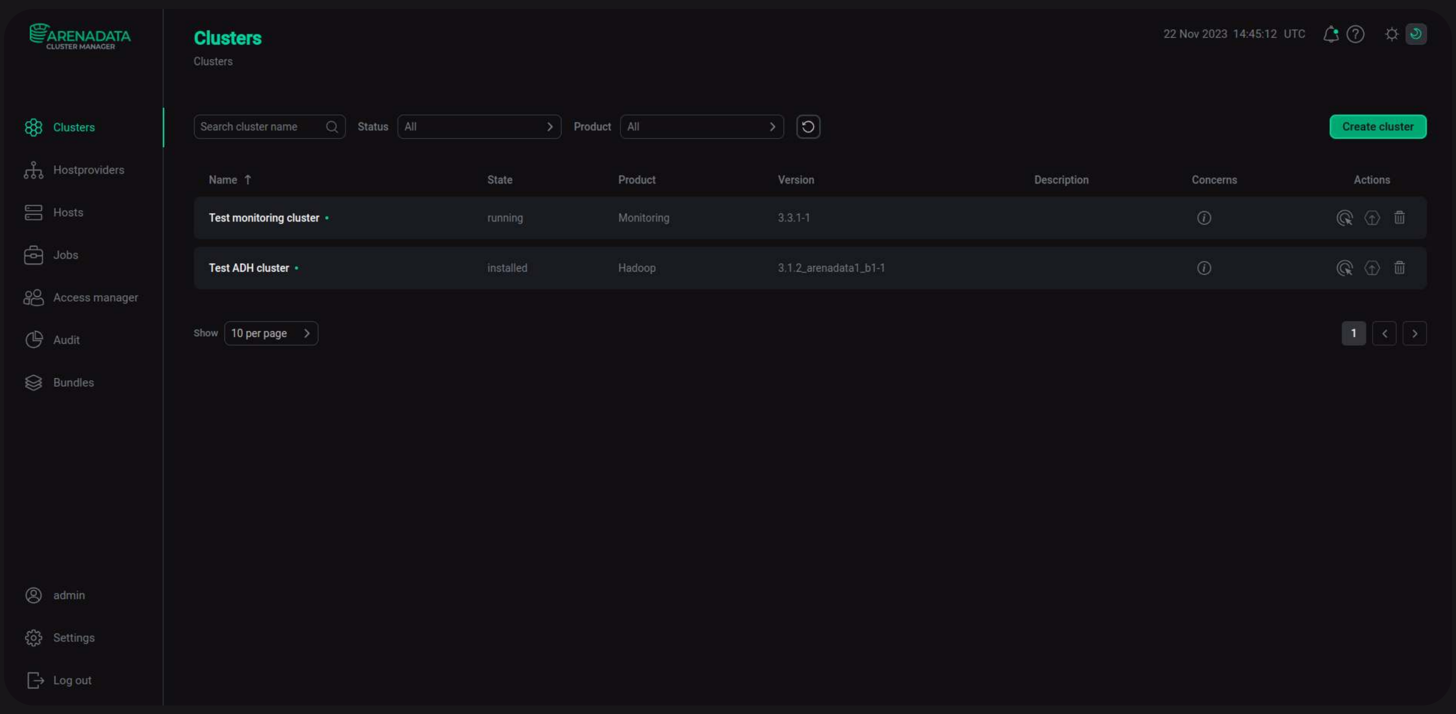
Евгения Кузина

Содержание

Назад к содержанию

Действия для работы с кластером

Статья описывает действия, которые можно выполнять с кластером ADH в пользовательском интерфейсе ADCM. Действия запускаются на странице ***Clusters***.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters
22 Nov 2023 14:45:12 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Show
10 per page
1

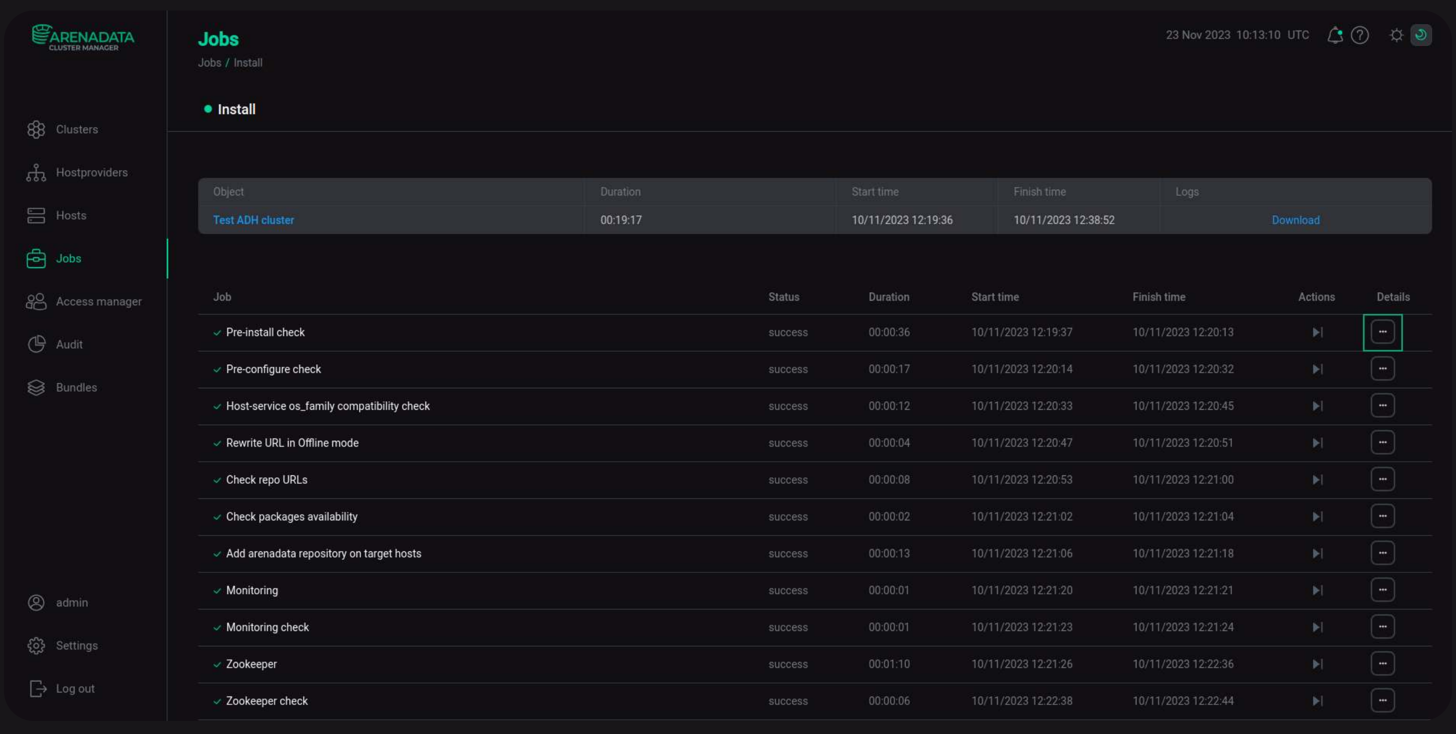
Name	State	Product	Version	Description	Concerns	Actions
Test monitoring cluster	running	Monitoring	3.3.1-1		ⓘ	
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1		ⓘ	

Страница Clusters

Сраница ***Clusters*** содержит таблицу со следующими столбцами:

- ***Name*** — имя, указанное при создании кластера.
- ***State*** — текущее состояние кластера (`created`, `installed` или `running`).
- ***Product*** — имя продукта.
- ***Version*** — версия бандла, который был использован для установки кластера.
- ***Description*** — описание, указанное при создании кластера.
- ***Concerns*** — отображает иконку, отражающую состояние запущенных процессов в кластере. Иконка ⓘ служит индикатором корректной настройки кластера. Если в конфигурации кластера обнаружены критические ошибки или запущено действие, в столбце отображается иконка ⓘ. При наведении на нее курсора появится всплывающее окно с описанием и ссылкой, по которой можно перейти, чтобы выполнить необходимую настройку или получить подробную информацию.
- ***Actions*** — в столбце показываются иконки для управления кластером:
  - ⊙ — открывает список действий для работы с кластером.
  - ⇧ — указывает, доступна ли новая версия бандла, и позволяет запустить обновление кластера. См. Обновление кластера.
  - 🗑 — удаляет информацию о кластере из ADCM. Это действие не удаляет сервисы ADH и не производит никаких изменений на хостах, относящихся к кластеру.

Вы можете использовать иконки, описанные выше, для выполнения действий. После того как действие запущено, ADCM отображает процесс его выполнения и результат на странице ***Jobs***. На этой странице можно кликнуть имя задачи, чтобы открыть страницу с подробной информацией о ходе её выполнения. Если задача состоит из нескольких этапов, нажмите многоточие в столбце ***Details***, чтобы получить дополнительную информацию о текущем этапе.


ARENADATA
CLUSTER MANAGER
Jobs
Jobs / Install
23 Nov 2023 10:13:10 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Install

Object	Duration	Start time	Finish time	Logs
Test ADH cluster	00:19:17	10/11/2023 12:19:36	10/11/2023 12:38:52	Download

Job	Status	Duration	Start time	Finish time	Actions	Details
✓ Pre-install check	success	00:00:36	10/11/2023 12:19:37	10/11/2023 12:20:13	▶|	⋯
✓ Pre-configure check	success	00:00:17	10/11/2023 12:20:14	10/11/2023 12:20:32	▶|	⋯
✓ Host-service os_family compatibility check	success	00:00:12	10/11/2023 12:20:33	10/11/2023 12:20:45	▶|	⋯
✓ Rewrite URL in Offline mode	success	00:00:04	10/11/2023 12:20:47	10/11/2023 12:20:51	▶|	⋯
✓ Check repo URLs	success	00:00:08	10/11/2023 12:20:53	10/11/2023 12:21:00	▶|	⋯
✓ Check packages availability	success	00:00:02	10/11/2023 12:21:02	10/11/2023 12:21:04	▶|	⋯
✓ Add arenadata repository on target hosts	success	00:00:13	10/11/2023 12:21:06	10/11/2023 12:21:18	▶|	⋯
✓ Monitoring	success	00:00:01	10/11/2023 12:21:20	10/11/2023 12:21:21	▶|	⋯
✓ Monitoring check	success	00:00:01	10/11/2023 12:21:23	10/11/2023 12:21:24	▶|	⋯
✓ Zookeeper	success	00:01:10	10/11/2023 12:21:26	10/11/2023 12:22:36	▶|	⋯
✓ Zookeeper check	success	00:00:06	10/11/2023 12:22:38	10/11/2023 12:22:44	▶|	⋯

Страница "Jobs"

Действия для работы с кластером

Для кластера ADH доступны следующие действия:

- Check
- Manage credential encryption
- Manage Kerberos
- Manage SSL
- Prepare hosts
- Reinstall status-checker
- Start
- Stop

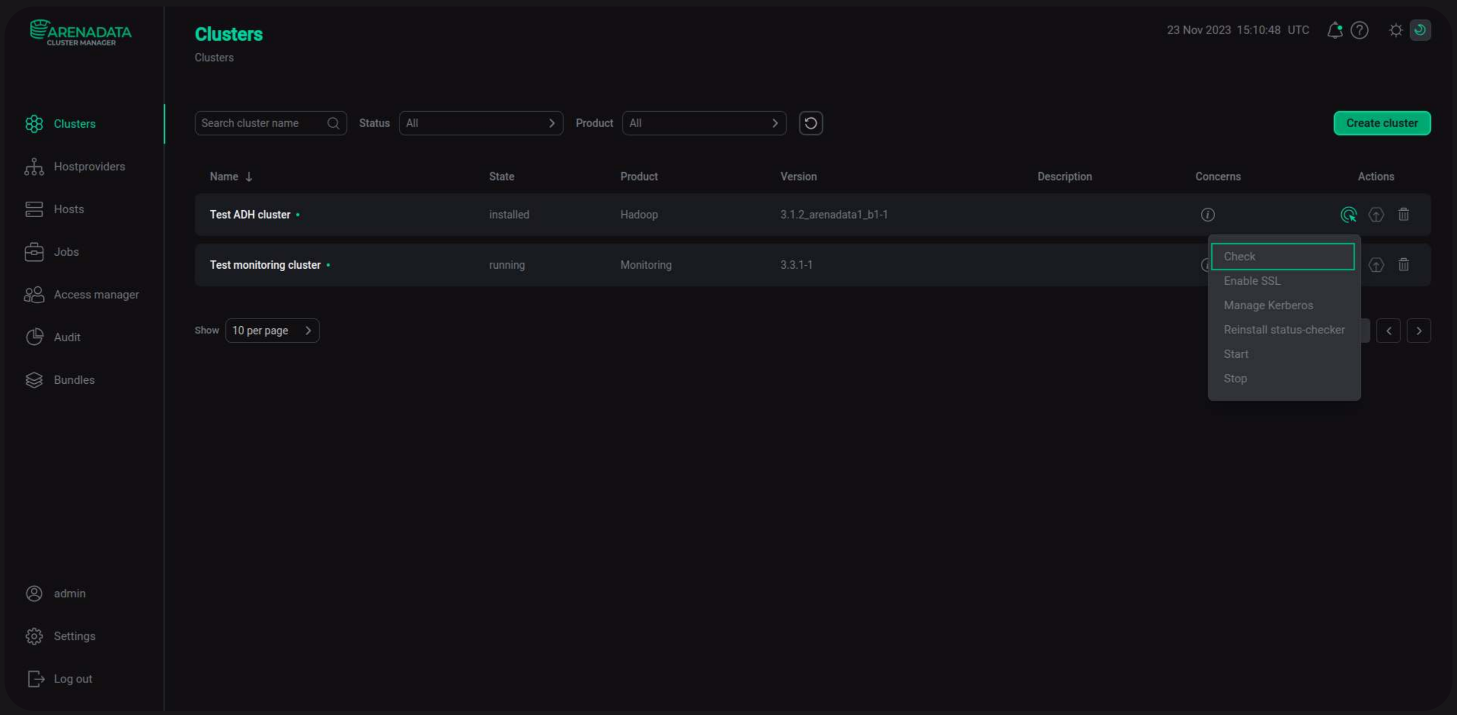
Check

Евгения Кузина

Назад к содержанию

Действие ***Check*** проверяет правильно ли работают сервисы ADH. ***Check*** последовательно запускает тесты для каждого сервиса, чтобы проверить работоспособность сервиса и его компонентов.

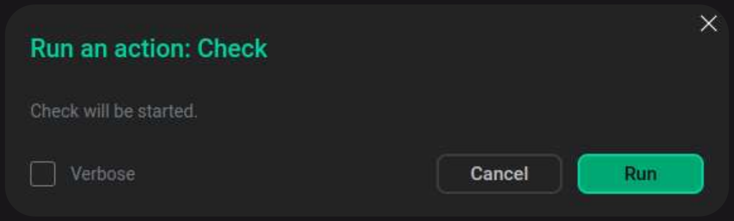
Чтобы выполнить действие, перейдите на страницу ***Clusters***, выберите кластер ADH, нажмите и выберите действие из появившегося списка.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters
23 Nov 2023 15:10:48 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
Show
10 per page

Name ↓	State	Product	Version	Description	Concerns	Actions
Test ADH cluster •	installed	Hadoop	3.1.2_arenadata1_b1-1			
Test monitoring cluster •	running	Monitoring	3.3.1-1			

Выполнение действия "Check"

Когда вы выбираете действие, ADCM отображает диалоговое окно для подтверждения выбора. В этом диалоговом окне можно установить флажок ***Verbose***, чтобы была выведена дополнительная информация о выполнении действия.


Run an action: Check
☐ Verbose
Cancel
Run

Check will be started.

Окно "Run an action"

Результат действия и процесс его выполнения отображаются на странице ***Jobs***. ***Jobs*** содержит детали выполнения всех действий, включая подробный STDOUT/STDERR вывод Ansible.

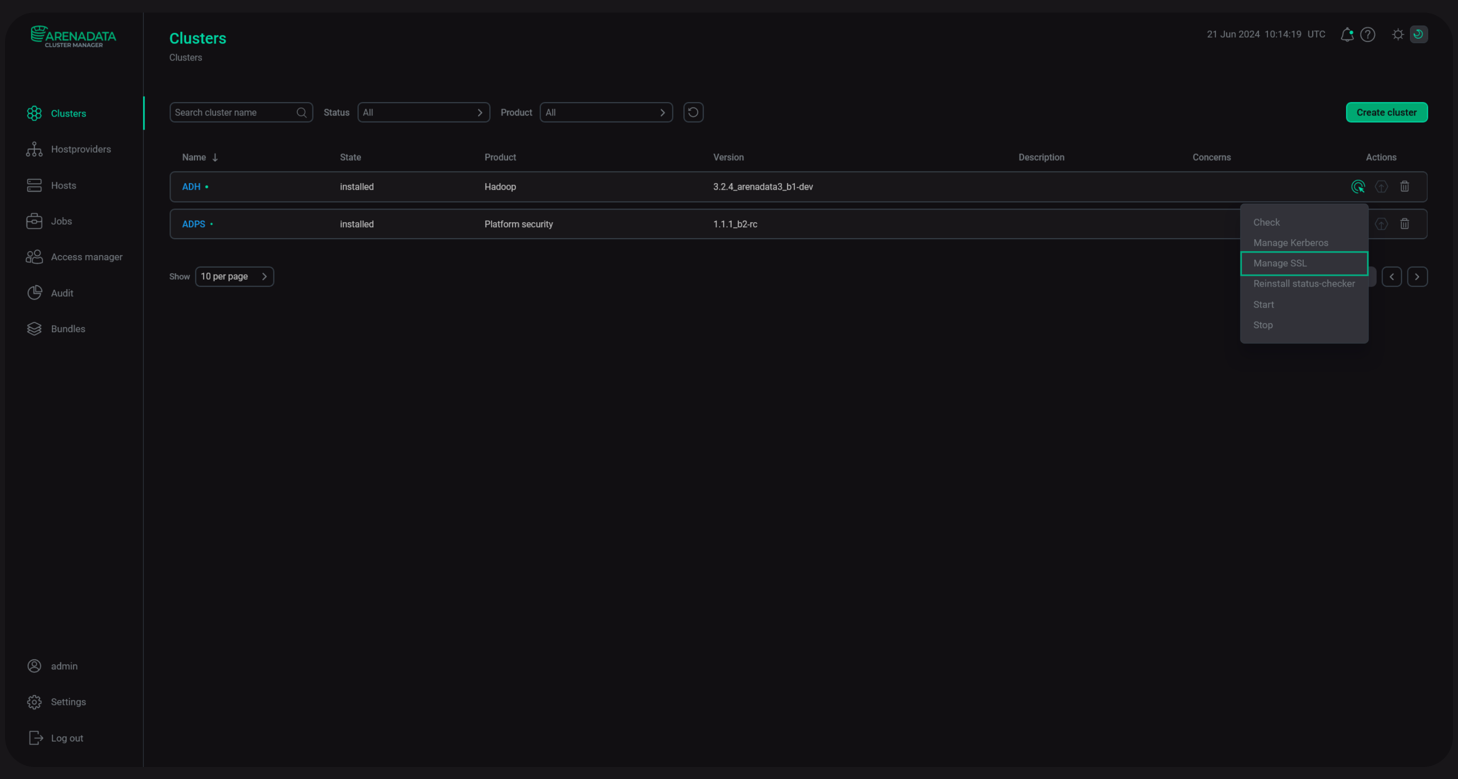
Manage SSL

Сергей Тихомиров, Евгения Кузина

Назад к содержанию

Действие ***Manage SSL*** позволяет управлять SSL-шифрованием всех сервисов в кластере ADH. За дополнительной информацией о настройках SSL можно обратиться к статье Требования к SSL-сертификатам.

Чтобы выполнить действие, перейдите на страницу ***Clusters***, выберите кластер ADH, нажмите и выберите действие из появившегося списка.


ARENADATA
CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
21 Jun 2024 10:14:19 UTC
Search cluster name
Status
All
Product
All
Create cluster
Check
Manage Kerberos
Manage SSL
Reinstall status-checker
Start
Stop
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
ADH	installed	Hadoop	3.2.4_arenadata3_b1-dev			
ADPS	installed	Platform security	1.1.1_b2-rc			

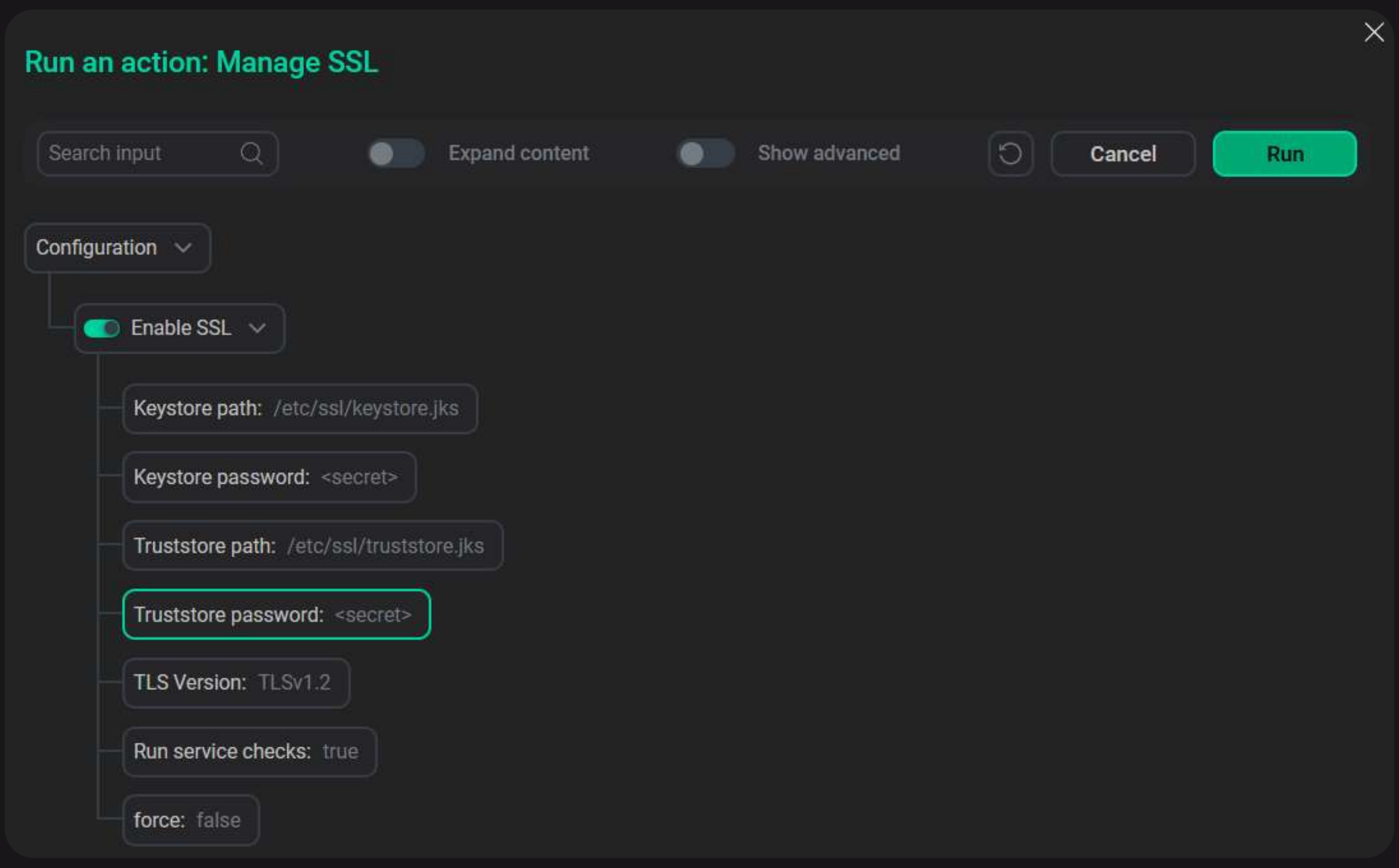
Выполнение действия "Manage SSL"

Когда вы выберете ***Manage SSL***, откроется диалоговое окно, в котором необходимо указать настройки SSL, перечисленные в таблице ниже.

Настройки SSL

Параметр	Описание	Значение по умолчанию
Keystore path	Путь к keystore-файлу	/etc/ssl/keystore.jks
Keystore password	Пароль для keystore-файла	—
Truststore path	Путь к truststore-файлу	/etc/ssl/truststore.jks
Truststore password	Пароль для truststore-файла	—
TLS Version	Версия протокола TLS. См. Установка версии TLS для кластера ADH	TLSv1.2
Run service checks	Запускает проверку работоспособности сервиса после применения настроек	True
force	Определяет, следует ли перезаписать текущие параметры SSL указанными значениями	False

Для активации SSL запустите действие с активированной группой ***Enable SSL*** и заполненными параметрами.

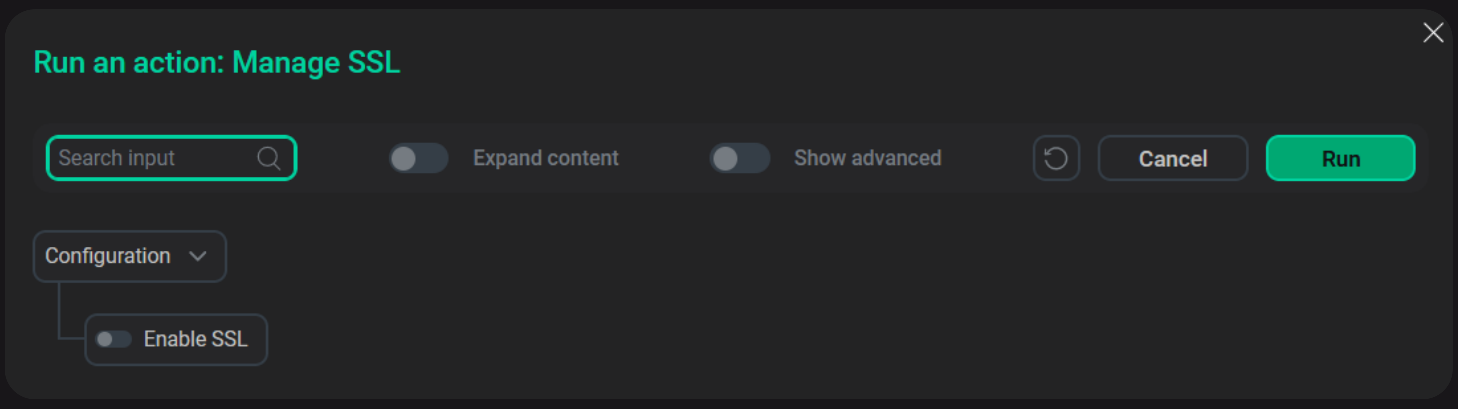

Run an action: Manage SSL
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Enable SSL

Keystore path: /etc/ssl/keystore.jks
Keystore password: <secret>
Truststore path: /etc/ssl/truststore.jks
Truststore password: <secret>
TLS Version: TLSv1.2
Run service checks: true
force: false

Включение SSL действием "Manage SSL"

После указания параметров нажмите ***Run***. В окне с подтверждением действия, которое появится, можно установить флажок ***Verbose***, чтобы была выведена дополнительная информация о выполнении действия.

Для отключения SSL запустите действие ***Manage SSL*** с деактивированной группой ***Enable SSL***.


Run an action: Manage SSL
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Enable SSL

Отключение SSL действием "Manage SSL"

Во время выполнения действия ***Manage SSL*** ADCM последовательно останавливает все сервисы, перенастраивает их для включения или выключения SSL-шифрования, затем запускает сервисы и производит их проверку.

Результат действия и процесс его выполнения отображаются на странице ***Jobs***. ***Jobs*** содержит детали выполнения всех действий, включая подробный STDOUT/STDERR вывод Ansible.

ВАЖНО

Если активирован плагин Ranger Solr, то при активации/деактивации SSL требуется вручную перезапустить Solr

Manage Credential Encryption

Сергей Тихомиров

Содержание

Назад к содержанию

Активация шифрования кластерным действием

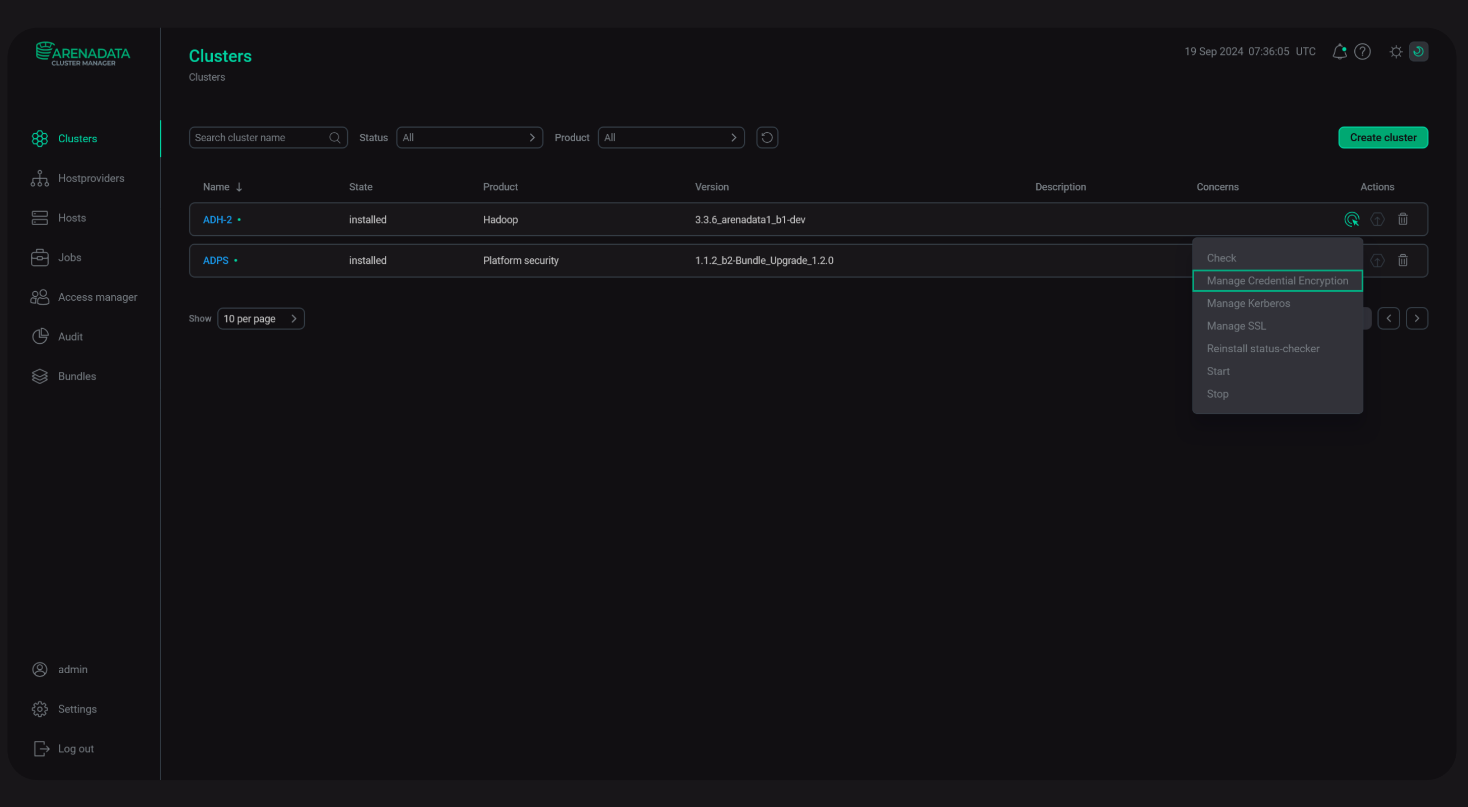
Активация шифрования для сервиса

Действие ***Manage Credential Encryption*** позволяет вам контролировать шифрование данных для сервисов кластера ADH. Шифрование реализовано с помощью провайдера Hadoop или KMS, каждый из которых поддерживает хранение паролей в JCEKS-файлах (защищенный контейнер с паролями, credential store) вместо конфигурационных файлов. По умолчанию credential store защищен паролем, который хранится в среде для каждого зашифрованного сервиса, но вы можете хранить пароль в файле, если сервис допускает это.

Активация шифрования кластерным действием

Для активации шифрования данных следуйте шагам ниже:

1. На странице ***Clusters*** запустите действие ***Manage Credential Encryption*** для вашего кластера ADH.

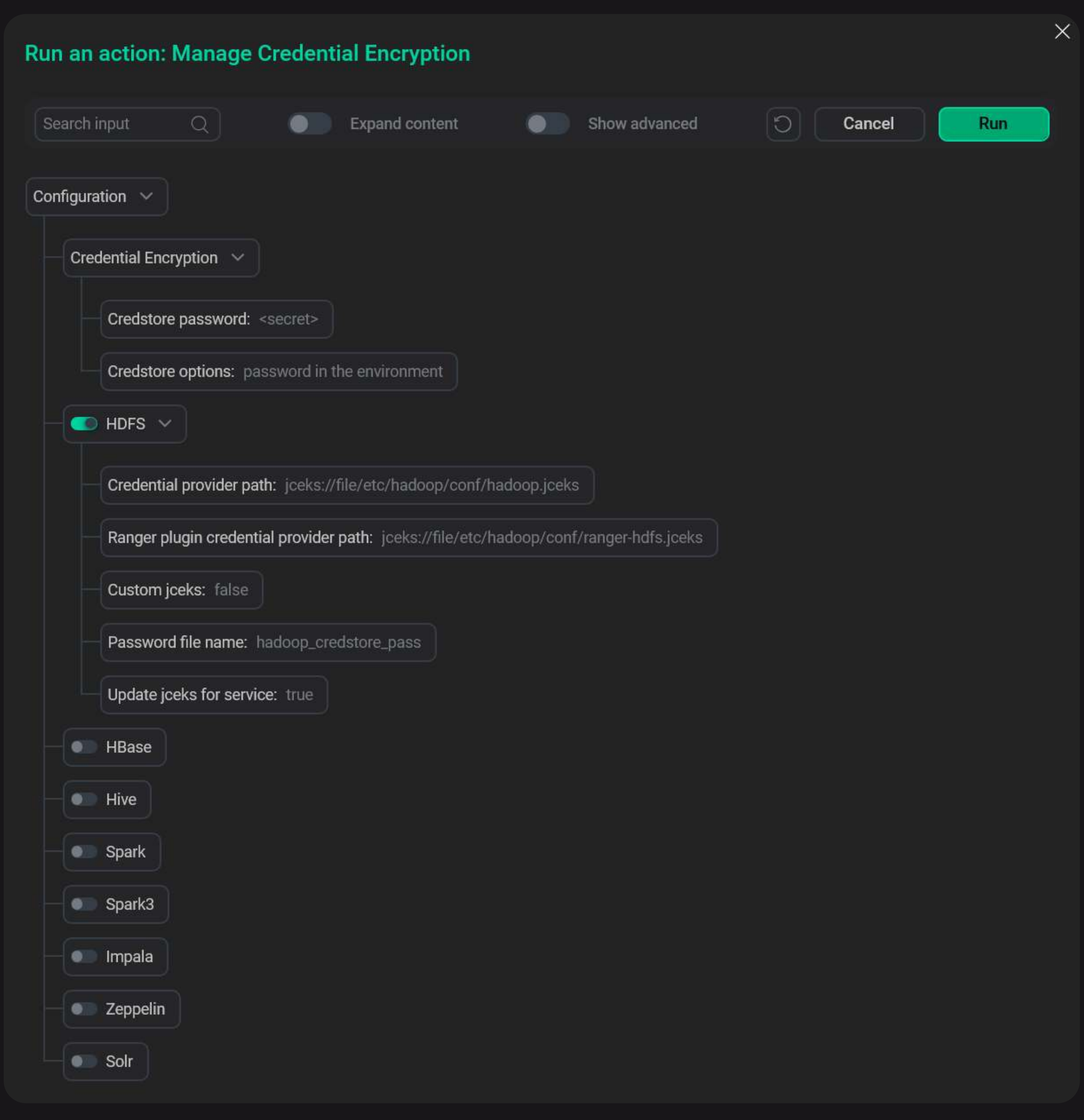


ARENADATA
CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
19 Sep 2024 07:36:05 UTC
Search cluster name
Status
Product
Create cluster
Check
Manage Credential Encryption
Manage Kerberos
Manage SSL
Reinstall status-checker
Start
Stop
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
ADH-2	installed	Hadoop	3.3.6_arenadata1_b1-dev			
ADPS	installed	Platform security	1.1.2_b2-Bundle_Upgrade_1.2.0			

Запуск действия Manage Credential Encryption

2. В открывшемся окне установите настройки шифрования:
   - ***Credstore password*** — пароль провайдера шифрования.
   - ***Credstore options*** — способ хранения пароля провайдера шифрования. Доступны три опции: `no password`, `password-file` и `password in the environment`.



Run an action: Manage Credential Encryption
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Credential Encryption
HDFS
HBase
Hive
Spark
Spark3
Impala
Zeppelin
Solr

Credstore password: <secret>

Credstore options: password in the environment

Credential provider path: jceks://file/etc/hadoop/conf/hadoop.jceks

Ranger plugin credential provider path: jceks://file/etc/hadoop/conf/ranger-hdfs.jceks

Custom jceks: false

Password file name: hadoop_credstore_pass

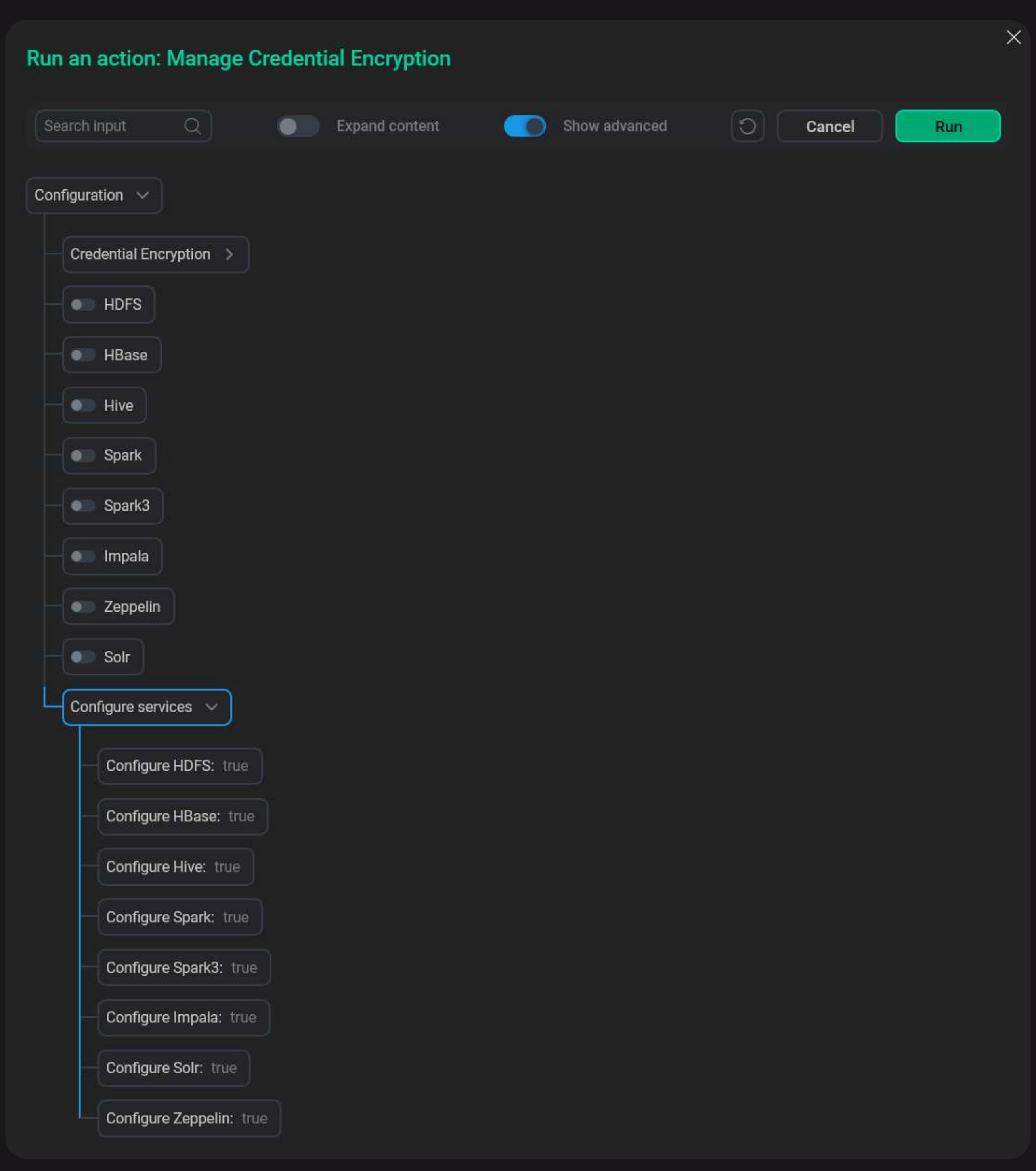
Update jceks for service: true

Параметры шифрования данных

3. В том же окне выберите сервисы, для которых хотите активировать шифрование, и укажите нужные значения параметров:
   - ***Credential provider path*** — путь к JCEKS-файлу сервиса. Может быть локальным или HDFS-путем.
   - ***Ranger plugin credential provider path*** — путь к JCEKS-файлу плагина Ranger для сервиса.
   - ***Custom jceks*** — при активном флаге используются пользовательские JCEKS-файлы, находящиеся по указанным выше путям. При неактивном состоянии флага JCEKS-файлы генерируются автоматически.
   - ***Password file name*** — имя файла, в котором хранится пароль. Используется только при выборе в параметре ***Credstore options*** способа хранения пароля `password-file`.
   - ***Update jceks for service*** — определяет, нужно ли перегенерировать JCEKS-файлы для этого сервиса. Имеет смысл только если JCEKS-файлы были созданы ранее. Следует использовать при смене паролей SSL (keystore, truststore), паролей LDAP, пароля для подключения к БД и прочих паролей из конфигурации сервисов.

   Активировать шифрование можно для сервисов HDFS (вместе с YARN), HBase, Hive, Impala, Kyuubi, Spark/Spark3, Zeppelin и Solr. Список параметров, приведенный выше, одинаков для каждого сервиса за исключением тех, которые не требуют полного списка параметров.

4. У данного действия также есть режим advanced, в котором можно выбирать сервисы, требующие реконфигурации. При активации флага ***Show advanced*** появляется группа параметров ***Configure services***, в которой можно отключить реконфигурацию для сервисов, не изменяемых действием ***Manage Credential Encryption***.



Run an action: Manage Credential Encryption
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Credential Encryption
HDFS
HBase
Hive
Spark
Spark3
Impala
Zeppelin
Solr
Configure services

Configure HDFS: true

Configure HBase: true

Configure Hive: true

Configure Spark: true

Configure Spark3: true

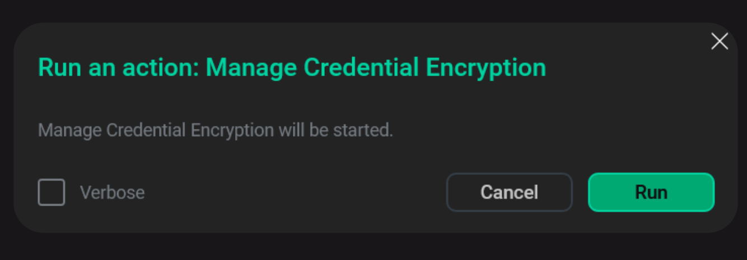
Configure Impala: true

Configure Solr: true

Configure Zeppelin: true

Режим advanced

5. Нажмите ***Run*** и подтвердите действие в появившемся окне.



Run an action: Manage Credential Encryption
Verbose
Cancel
Run

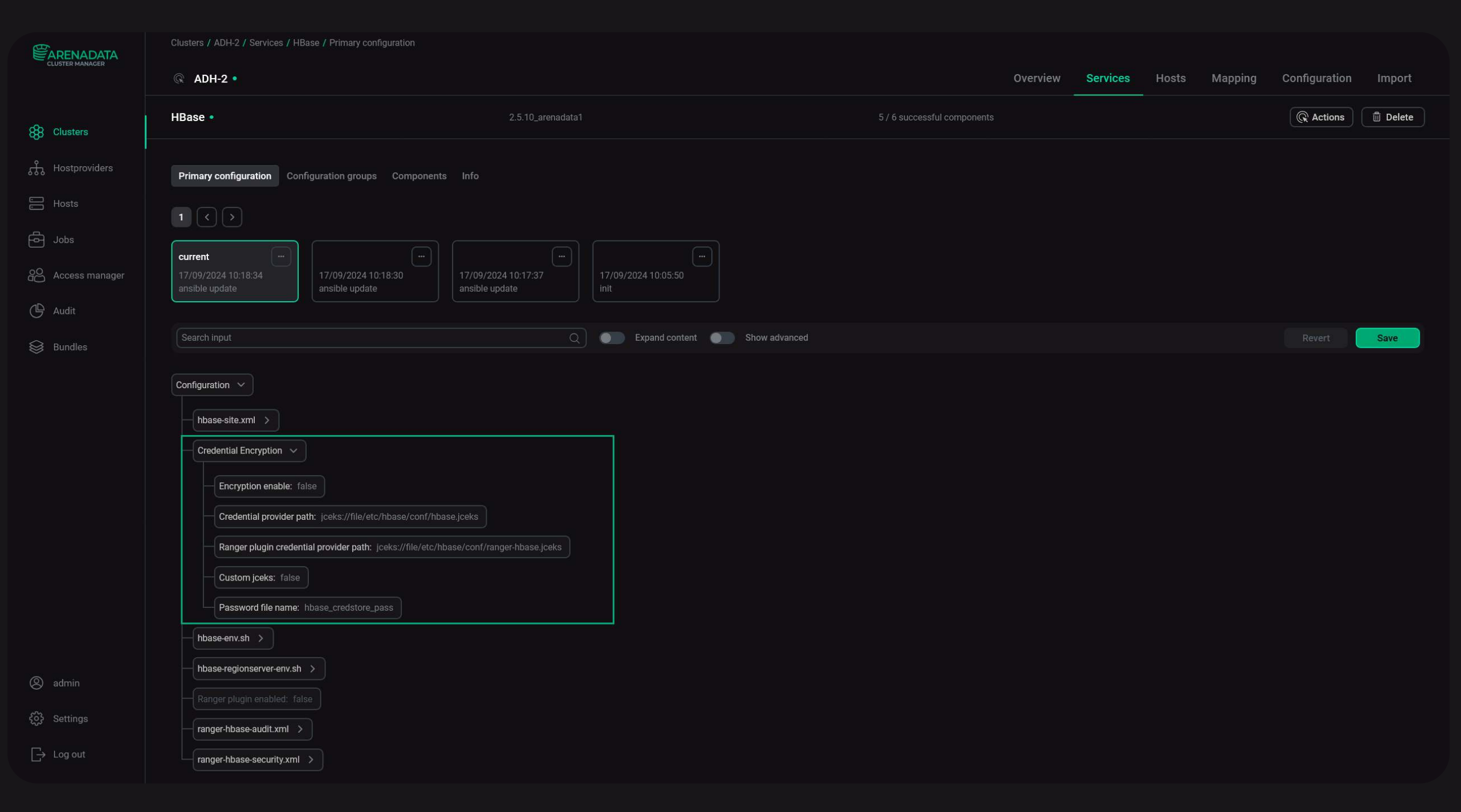
Manage Credential Encryption will be started.

Окно подтверждения

Активация шифрования для сервиса

Существует альтернативный способ активировать шифрование для сервиса, который поддерживает это. Вместо запуска действия ***Manage Credential Encryption*** вы можете выполнить следующее:

1. Перейдите на вкладку ***Services*** вашего кластера ADH и выберите сервис, для которого хотите включить шифрование.
2. Раскройте группу ***Credential Encryption*** и настройте параметры, указанные на рисунке ниже.



ARENADATA
CLUSTER MANAGER
Clusters / ADH-2 / Services / HBase / Primary configuration
ADH-2
Overview
Services
Hosts
Mapping
Configuration
Import
HBase
2.5.10_arenadata1
5 / 6 successful components
Actions
Delete
Primary configuration
Configuration groups
Components
Info
current
17/09/2024 10:18:34
ansible update
17/09/2024 10:18:30
ansible update
17/09/2024 10:17:37
ansible update
17/09/2024 10:05:50
init
Search input
Expand content
Show advanced
Revert
Save
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Configuration
hbase-site.xml
Credential Encryption
hbase-env.sh
hbase-regionserver-env.sh
ranger-hbase-audit.xml
ranger-hbase-security.xml

Encryption enable: false

Credential provider path: jceks://file/etc/hbase/conf/hbase.jceks

Ranger plugin credential provider path: jceks://file/etc/hbase/conf/ranger-hbase.jceks

Custom jceks: false

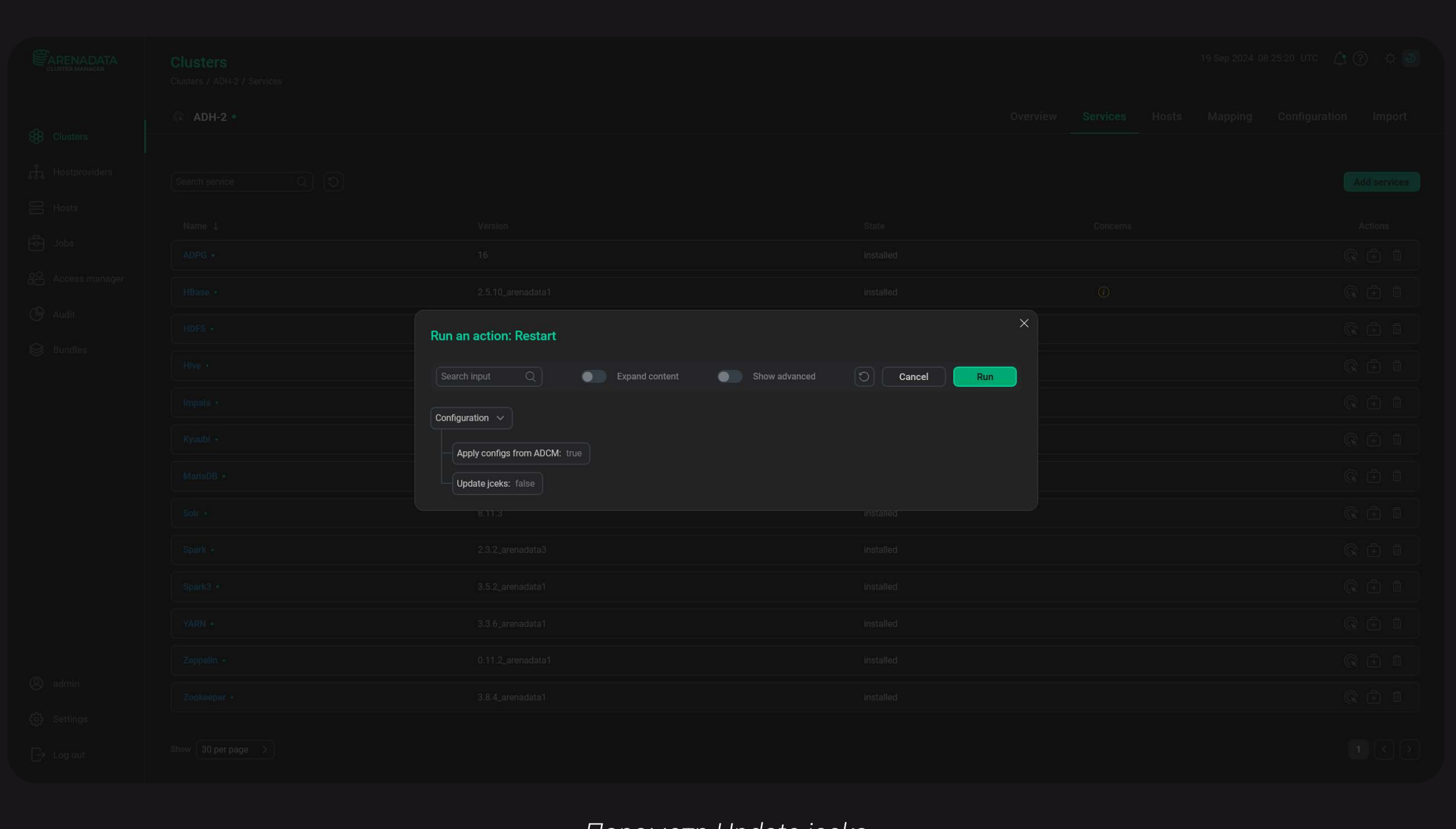
Password file name: hbase_credstore_pass

Ranger plugin enabled: false

Настройка сервиса

Список параметров похож на имеющийся в действии ***Manage Credential Encryption***, но здесь присутствует новый параметр — ***Encryption enable***. Для активации шифрования для сервиса требуется активировать его и сохранить конфигурацию.

3. Перезапустите сервис. Параметр ***Update jceks*** появляется в меню действия только после активации конфигурационного параметра ***Encryption enable***. Его необходимо активировать при первом включении шифрования через действие ***Restart*** (в отличие от действия ***Manage encryption***, которое создает архив с паролями автоматически без необходимости дополнительных действий).



Run an action: Restart
Search input
Expand content
Show advanced
Cancel
Run
Configuration

Apply configs from ADCM: true

Update jceks: false

Параметр Update jceks

Для обновления JCEKS-файлов (при изменении сервисного пароля или обновлении настроек JCEKS-провайдера) активируйте его и нажмите ***Run***.

Manage Kerberos

Сергей Тихомиров

Содержание

Назад к содержанию

Действие Manage Kerberos

Опция Custom kerberization settings

Рекомендации для пользовательских keytab-файлов

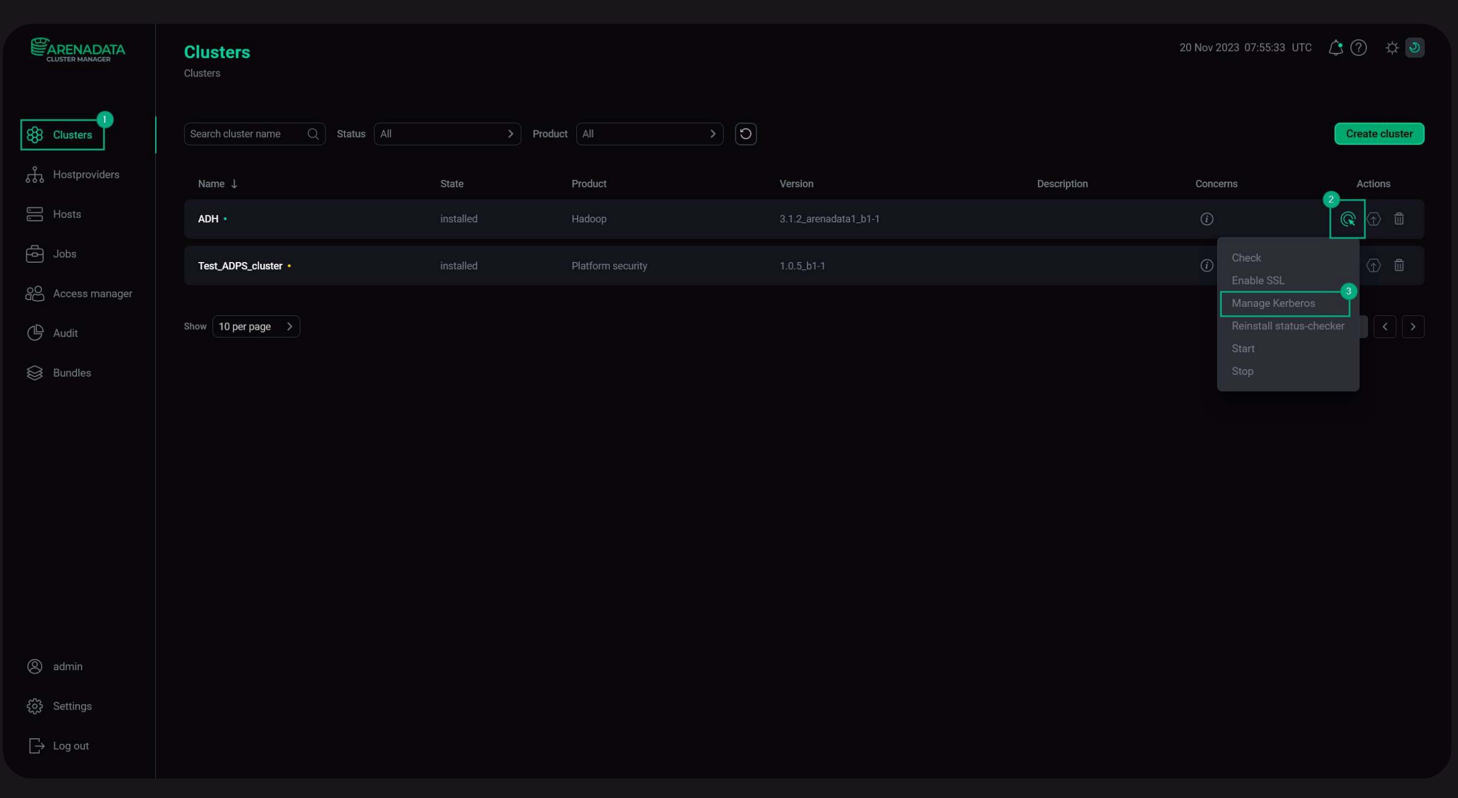
Действие Manage Kerberos

Действие ***Manage Kerberos*** позволяет включать, реконфигурировать и отключать Kerberos. Для его запуска потребуется установленный ADH-кластер (см. Установка Arenadata Hyperwave) и один или несколько Центров Распределения Ключей (Key Distribution Center, KDC). Прежде чем приступать к управлению Kerberos, рекомендуется прочитать статью Обзор Kerberos о требованиях для керберизации.

Среди прочего, ***Manage Kerberos*** создан для реализации следующих сценариев:

- Включение Kerberos без данных администратора с помощью существующего клиента, принципалов и/или keytab-файлов.
- Быстрый перевод кластера из состояния kerberized в состояние non-kerberized в случае возникновения ошибки в начале процесса керберизации (например, неверные данные kadmin) без необходимости реконфигурации сервисов, удаления принципалов и keytab-файлов.
- Отложенная реконфигурация сервисов — возможность сконфигурировать Kerberos в моменте, а реконфигурировать и перезапустить сервисы позже.

Для запуска действия в веб-интерфейсе ADCM перейдите на страницу ***Clusters***. Выберите установленный и подготовленный кластер ADH, и запустите действие ***Manage Kerberos***.



ARENADATA CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
20 Nov 2023 07:55:33 UTC
Search cluster name
Status
All
Product
All
Create cluster
Show
10 per page
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
1
2
3

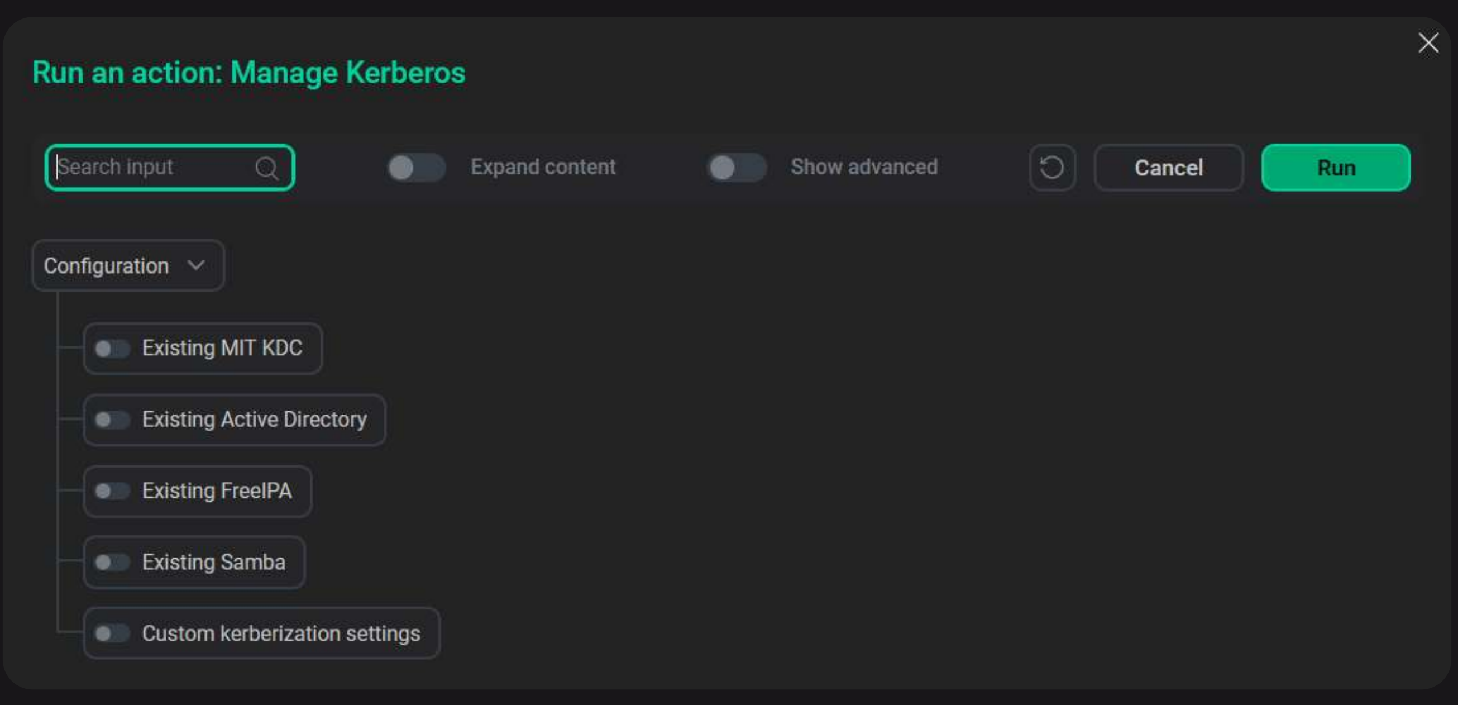
Name	State	Product	Version	Description	Concerns	Actions
ADH	installed	Hadoop	3.1.2_arenadata1_b1-1			
Test_ADPS_cluster	installed	Platform security	1.0.5_b1-1			

Manage Kerberos

В появившемся окне будет предложено несколько опций для запуска действия:

- С помощью существующего MIT KDC.
- С помощью существующей MS Active Directory.
- С помощью существующей FreeIPA.
- С помощью существующей Samba.
- С помощью опции Custom kerberization settings.

Для правильного запуска действия выберите одну из предложенных опций KDC: ***Existing MIT KDC***, ***Existing Active Directory***, ***Existing FreeIPA*** или ***Existing Samba***. Каждую из этих опций можно совместить с опцией ***Custom kerberization settings***.



Run an action: Manage Kerberos
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Existing Samba
Custom kerberization settings

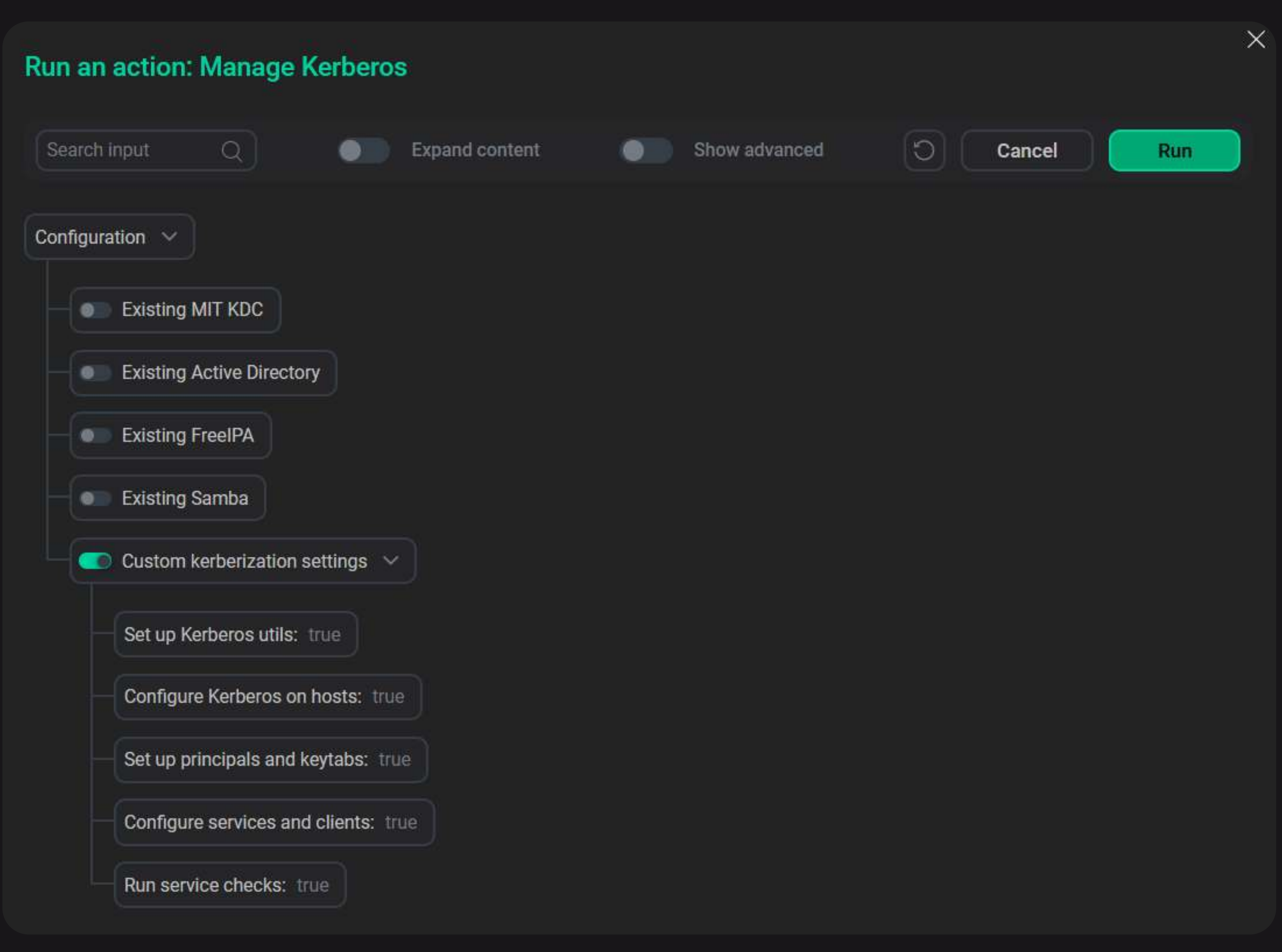
Способы управления Kerberos

ВАЖНО

Запуск действия ***Manage Kerberos*** с выбранным KDC автоматически активирует Kerberos.

Опция Custom kerberization settings

Опция ***Custom kerberization settings*** позволяет пользователю выбирать шаги керберизации, например, создание принципалов и keytab-файлов.



Run an action: Manage Kerberos
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Existing Samba
Custom kerberization settings

Set up Kerberos utils: true
Configure Kerberos on hosts: true
Set up principals and keytabs: true
Configure services and clients: true
Run service checks: true

Параметры Custom kerberization settings

Описание параметров Custom kerberization settings

Параметр	Описание	Значение по умолчанию
Set up Kerberos utils	Активирует возможность устанавливать и удалять клиенты и утилиты Kerberos. Данный параметр учитывается в действиях ***Expand*** и ***Install***	True
Configure Kerberos on hosts	Активирует возможность конфигурировать кластер, включая *krb5.conf*, *ldap.conf*	True
Set up principals and keytabs	Активирует возможность создавать, пересоздавать и удалять принципалов и keytab-файлы. Пароли для принципалов генерируются случайным образом перед созданием keytab-файлов. Данный параметр учитывается в действиях ***Expand*** и ***Install***. Бандл ADCM установит owner и разрешения для keytab-файлов только если этот флаг установлен. В случае отсутствия прав администратора пользователь должен предоставить подготовленный keytab-файл с правильно установленными owner и разрешениями (см. Рекомендации для пользовательских keytab-файлов)	True
Configure services and clients	Активирует возможность обновлять конфигурации сервисов и клиентов	True
Run service checks	Активирует возможность проводить проверку сервисов	True

Рекомендации для пользовательских keytab-файлов

Ниже представлены рекомендации для значений owner, group и прав для keytab-файлов.

Рекомендации для keytab-файлов

Short name компонента	Owner keytab-файла	Group keytab-файла	Права
zookeeper	zookeeper	zookeeper	600
hdfs-datanode	hdfs	hadoop	600
hdfs-namenode	hdfs	hadoop	600
hdfs-journalnode	hdfs	hadoop	600
hdfs-secondarynamenode	hdfs	hadoop	600
hdfs-zkfc	hdfs	hadoop	600
HTTP	hdfs	hadoop	640
httpfs	httpfs	httpfs	600
hdfs	hdfs	hadoop	640
yarn	yarn	hadoop	640
yarn-resourcemanager	yarn	hadoop	600
yarn-nodemanager	yarn	hadoop	600
yarn-timelineserver	yarn	hadoop	600
mapreduce-historyserver	mapred	hadoop	600
hbase	hbase	hadoop	640
hbase-phoenix_queryserver	phoenix	phoenix	600
hbase-thrift2	hbase	hbase	600
hive	hive	hive	640
solr	solr	solr	600
spark	spark	spark	640
livy	livy	hadoop	600
airflow	airflow	airflow	600
flink	flink	flink	600
zeppelin	zeppelin	zeppelin	600
impala	impala	impala	600

Prepare hosts

Елена Костюченко

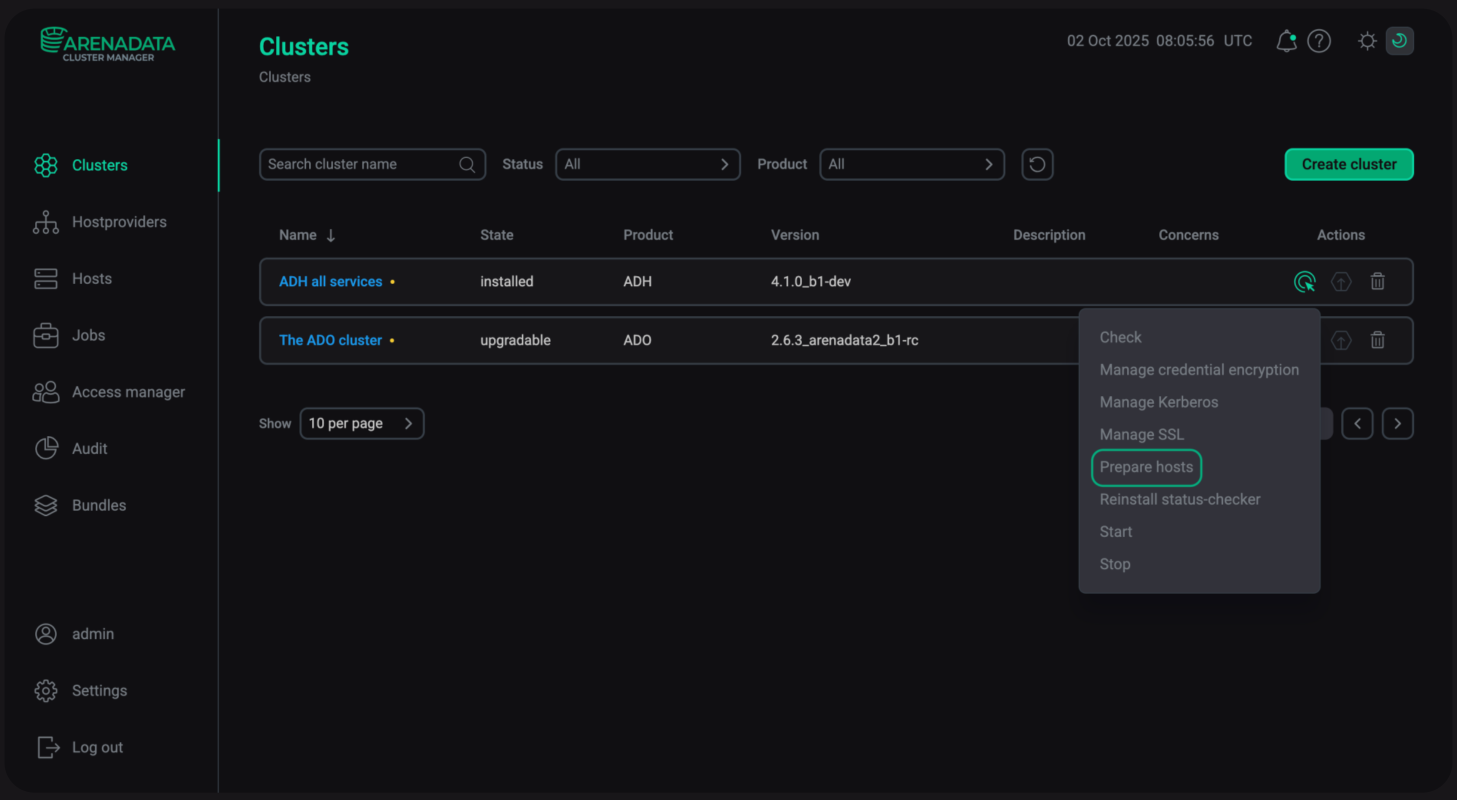
Назад к содержанию

Действие ***Prepare hosts*** устанавливает на хосты все пакеты, объявленные в бандле кластера, тем самым подготавливая новые хосты перед добавлением компонентов сервисов. Предварительная установка необходимых зависимостей позволяет избежать длительного периода блокировки кластера при добавлении новых сервисов. При выполнении этого действия на новых хостах не устанавливаются и не запускаются компоненты сервисов.

При добавлении новых хостов в кластер рекомендуется выполнить действие ***Prepare hosts*** перед установкой любых компонентов сервисов. Если компоненты установлены без подготовки хостов, те же операции будут выполнены во время установки компонента, но кластер будет заблокирован более продолжительное время.

Вы можете активировать действие ***Prepare hosts*** через действия кластера на странице ***Clusters*** или через действие для группы хостов. Чтобы выполнить действие, перейдите на страницу ***Clusters***, выберите кластер ADH, нажмите и выберите действие из появившегося списка.

Начиная с версии ADCM 2.8.0 подготовить хосты также можно через меню ***Actions*** на странице ***Hosts***.

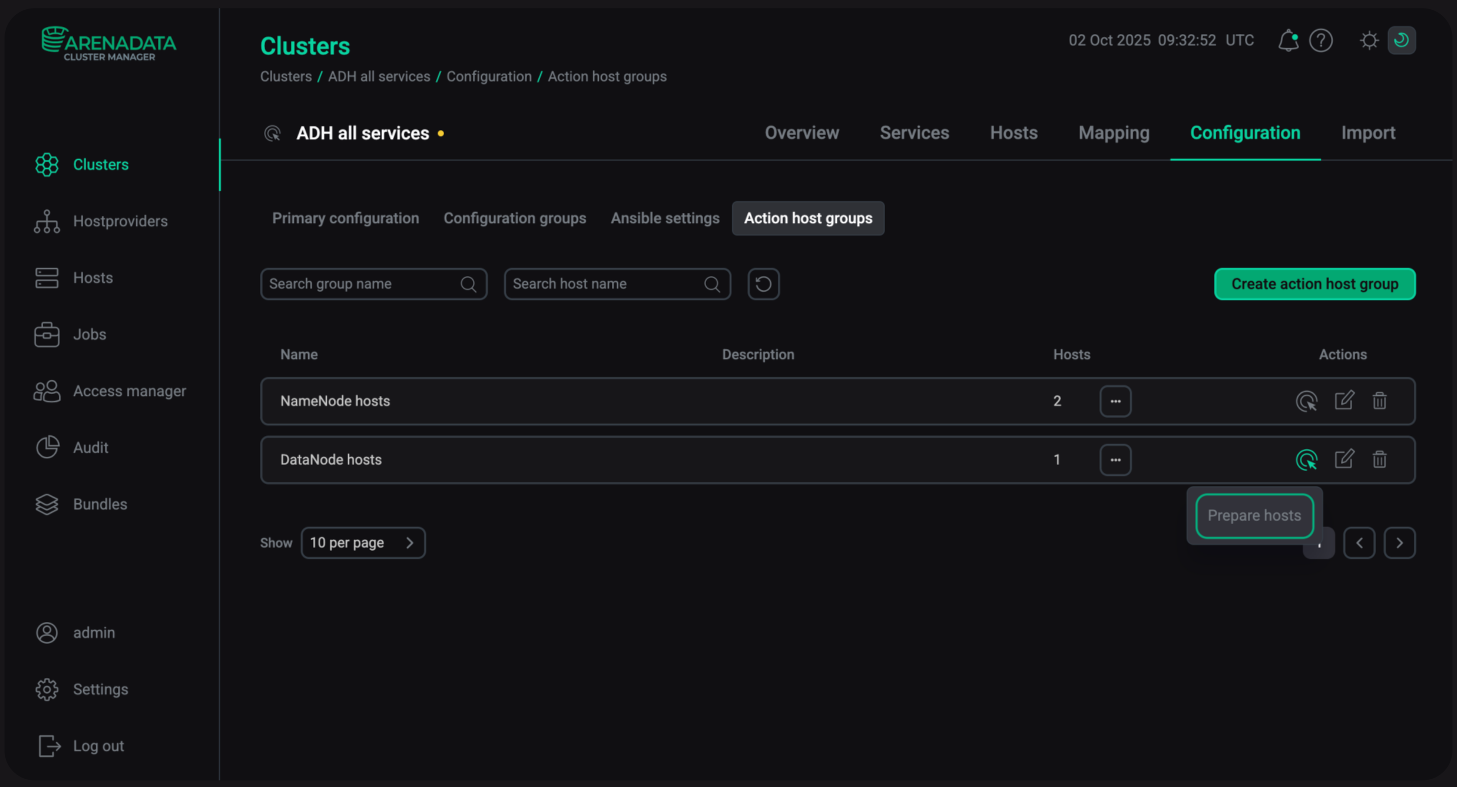

ARENADATA
CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters
02 Oct 2025 08:05:56 UTC
Search cluster name
Status
All
Product
All
Create cluster
Check
Manage credential encryption
Manage Kerberos
Manage SSL
Prepare hosts
Reinstall status-checker
Start
Stop
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
ADH all services	installed	ADH	4.1.0_b1-dev			
The ADO cluster	upgradable	ADO	2.6.3_arenadata2_b1-rc			

Выполнение действия "Prepare hosts" на странице Clusters

Чтобы подготовить группу хостов:

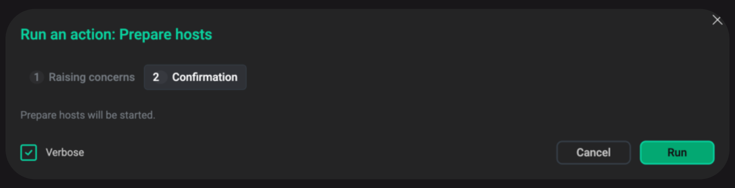
1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Configuration*** и выберите ***Action host groups***.
3. Для выбранной группы хостов откройте меню ***Actions*** и выберите ***Prepare hosts***.


ARENADATA
CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / ADH all services / Configuration / Action host groups
02 Oct 2025 09:32:52 UTC
ADH all services
Overview
Services
Hosts
Mapping
Configuration
Import
Primary configuration
Configuration groups
Ansible settings
Action host groups
Search group name
Search host name
Create action host group
Prepare hosts
Show
10 per page

Name	Description	Hosts	Actions
NameNode hosts		2	
DataNode hosts		1	

Выполнение действия "Prepare hosts" для группы хостов

Когда вы выбираете действие, ADCM отображает диалоговое окно для подтверждения выбора. В этом диалоговом окне можно установить флажок ***Verbose***, чтобы была выведена дополнительная информация о выполнении действия.


Run an action: Prepare hosts
1 Raising concerns
2 Confirmation
Cancel
Run

Prepare hosts will be started.

☑ Verbose

Окно "Run an action"

Результат действия и процесс его выполнения отображаются на странице ***Jobs***. ***Jobs*** содержит детали выполнения всех действий, включая подробный STDOUT/STDERR вывод Ansible.

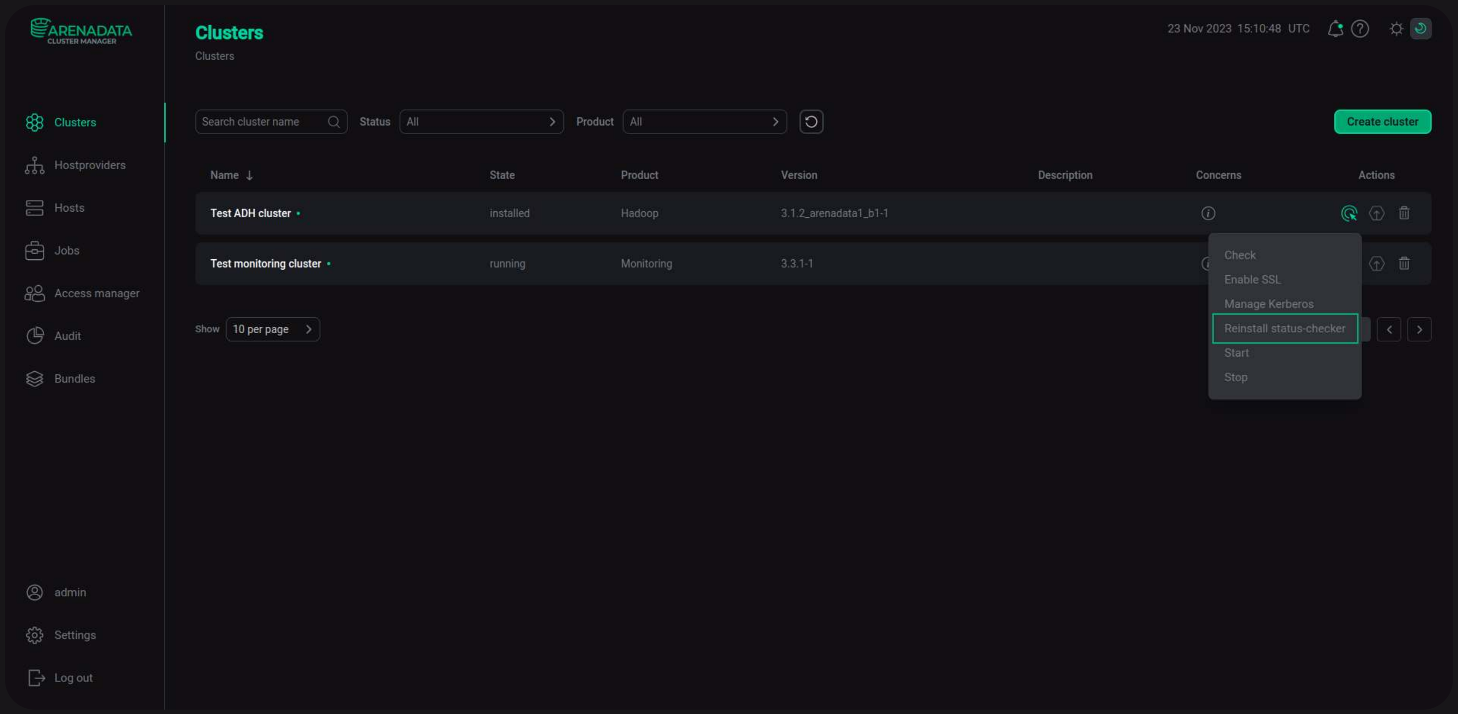
Reinstall status-checker

Евгения Кузина

Назад к содержанию

Действие ***Reinstall status-checker*** последовательно переустанавливает status-checker для всех сервисов.

Чтобы выполнить действие, перейдите на страницу ***Clusters***, выберите кластер ADH, нажмите ◎ и выберите действие из появившегося списка.



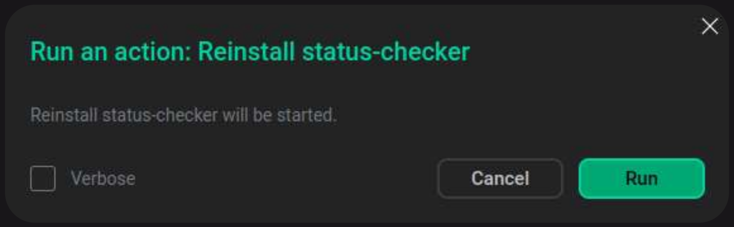
Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1			
Test monitoring cluster	running	Monitoring	3.3.1-1			

23 Nov 2023 15:10:48 UTC

ARENADATA
CLUSTER MANAGER
Clusters
Clusters
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
Show
10 per page

Выполнение действия "Reinstall status-checker"

Когда вы выбираете действие, ADCM отображает диалоговое окно для подтверждения выбора. В этом диалоговом окне можно установить флажок ***Verbose***, чтобы была выведена дополнительная информация о выполнении действия.



Run an action: Reinstall status-checker

Reinstall status-checker will be started.

☐ Verbose
Cancel
Run

Окно "Run an action"

Результат действия и процесс его выполнения отображаются на странице ***Jobs***. ***Jobs*** содержит детали выполнения всех действий, включая подробный STDOUT/STDERR вывод Ansible.

Start

Евгения Кузина

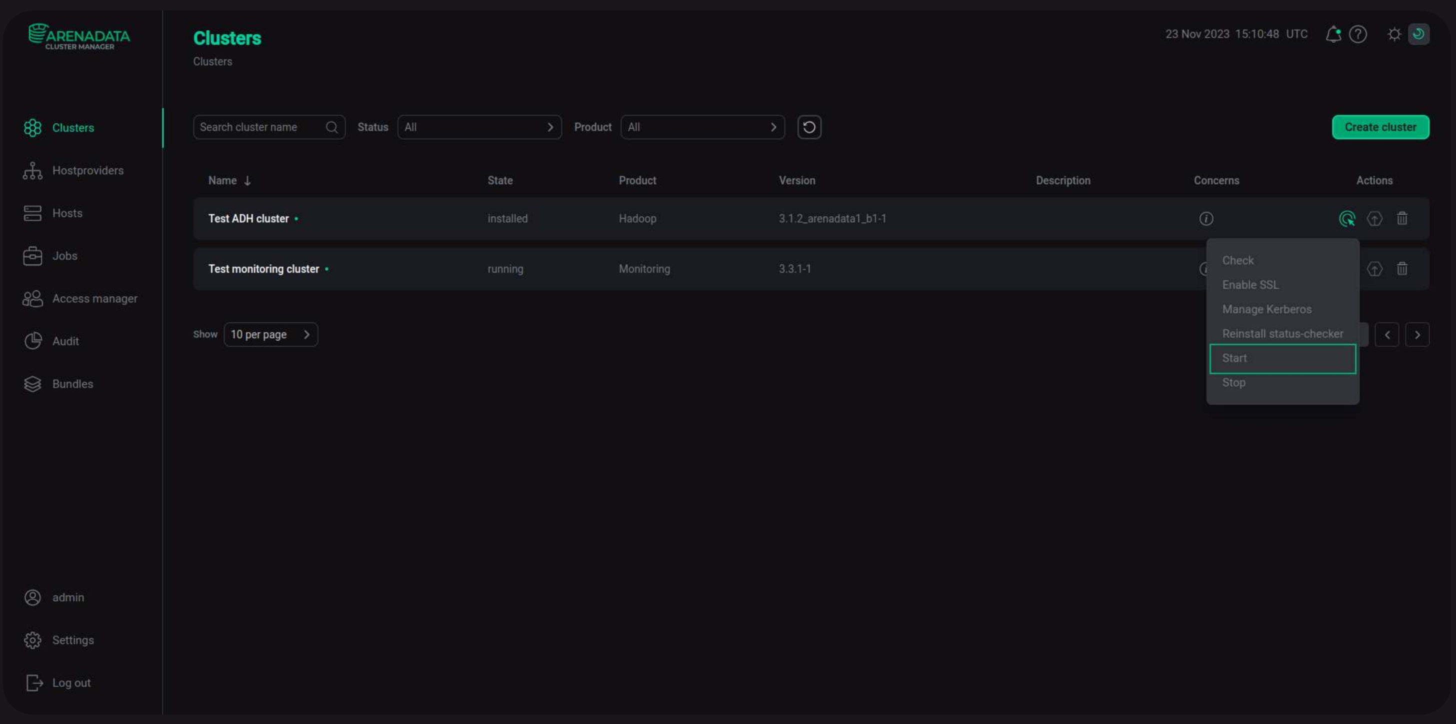
Назад к содержанию

ВАЖНО

Если вы установили кластер мониторинга и хотите использовать его с кластером ADH, сначала запустите кластер мониторинга. Для этого выполните действие ***Start*** кластера мониторинга так же, как описано для кластера ADH.

Действие ***Start*** запускает все сервисы, установленные в кластере ADH, один за другим в правильном порядке.

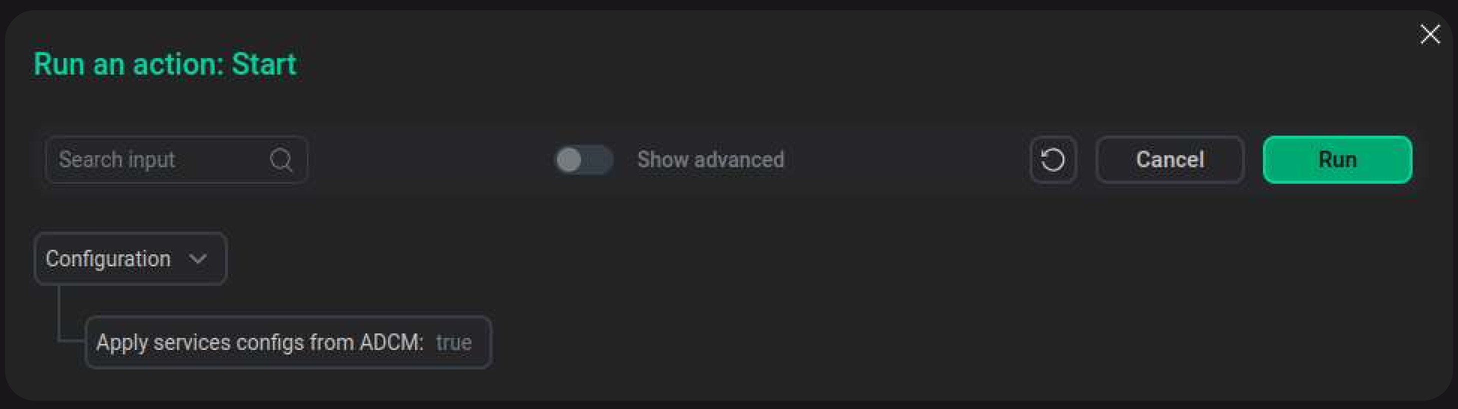
Чтобы выполнить действие, перейдите на страницу ***Clusters***, выберите кластер ADH, нажмите и выберите действие из появившегося списка.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters
23 Nov 2023 15:10:48 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1			
Test monitoring cluster	running	Monitoring	3.3.1-1			

Выполнение действия "Start"

Диалоговое окно действия ***Start*** содержит опцию ***Apply Services configs from ADCM***, которая позволяет определить, следует ли применять параметры конфигурации, указанные в пользовательском интерфейсе ADCM, к сервисам ADH.


Run an action: Start
Search input
Show advanced
Cancel
Run
Configuration

Apply services configs from ADCM: true

Окно "Run an action"

Нажмите ***Run***, чтобы запустить кластер. В появившемся окне подтверждения действия вы можете установить флажок ***Verbose***, чтобы была выведена дополнительная информация о выполнении действия.

Результат действия и процесс его выполнения отображаются на странице ***Jobs***. ***Jobs*** содержит детали выполнения всех действий, включая подробный STDOUT/STDERR вывод Ansible.

Также можно использовать действие ***Start*** определенного сервиса, чтобы вручную запустить только необходимые сервисы. Следует запускать сервисы в порядке, указанном в таблице ниже.

Номер по порядку	Название сервиса
1	Monitoring (если используется кластер мониторинга)
2	Zookeeper
3	MySQL
4	HDFS
5	YARN
6	HBase
7	Hive
8	Spark
9	Flink
10	Solr
11	Airflow

Большинству сервисов ADH требуются следующие работающие сервисы:

- Zookeeper
- HDFS
- YARN

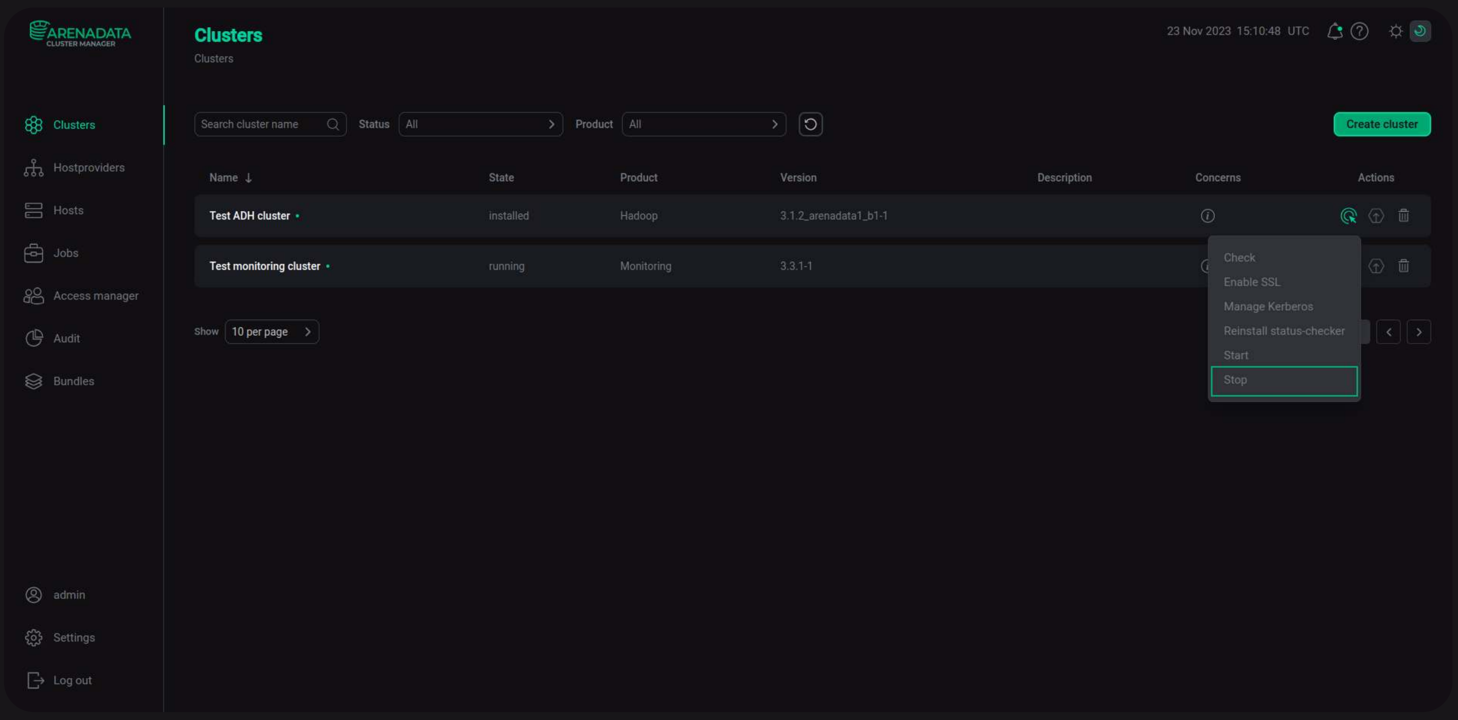
Stop

Евгения Кузина

Назад к содержанию

Действие ***Stop*** останавливает все сервисы, установленные в кластере ADH, в правильном порядке.

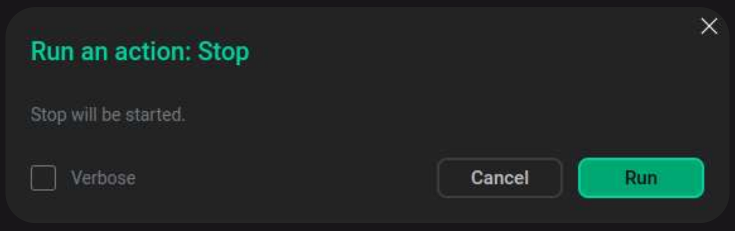
Чтобы выполнить действие, перейдите на страницу ***Clusters***, выберите кластер ADH, нажмите и выберите действие из появившегося списка.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters
23 Nov 2023 15:10:48 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
Show
10 per page

Name	State	Product	Version	Description	Concerns	Actions
Test ADH cluster	installed	Hadoop	3.1.2_arenadata1_b1-1			
Test monitoring cluster	running	Monitoring	3.3.1-1			

Выполнение действия "Stop"

Когда вы выбираете действие, ADCM отображает диалоговое окно для подтверждения выбора. В этом диалоговом окне можно установить флажок ***Verbose***, чтобы была выведена дополнительная информация о выполнении действия.


Run an action: Stop

Stop will be started.

☐ Verbose
Cancel
Run

Окно "Run an action"

Результат действия и процесс его выполнения отображаются на странице ***Jobs***. ***Jobs*** содержит детали выполнения всех действий, включая подробный STDOUT/STDERR вывод Ansible.

Также можно использовать действие ***Stop*** определенного сервиса, чтобы вручную остановить необходимые сервисы или весь кластер. Для остановки кластера следует останавливать сервисы в порядке, указанном в таблице ниже.

Номер по порядку	Название сервиса
1	Airflow
2	Solr
3	Flink
4	Spark
5	Hive
6	HBase
7	YARN
8	HDFS
9	MySQL
10	Zookeeper
11	Monitoring

Обновление кластера

Константин Алпашкин

Содержание

Назад к содержанию

Схема именования бандлов
Доступные версии бандлов
Процесс обновления
  - Обновление мажорной версии
  - Обновление минорной версии

Данный раздел содержит информацию о версиях ADH-бандлов, а также рекомендации по обновлению кластера на более новые версии.

Схема именования бандлов

Имена ADH-бандлов имеют следующий формат.

До версии 3.x

```
adcm_cluster_hadoop_v<X>.<Y>.<Z>_b<N>-<M>_<E>.tgz
```

Где:

- `<X>.<Y>.<Z>` — версия пакета.
- `b<N>-<M>` — версия ansible-скриптов, описывающих бандл.
- `<E>` — версия издания, Community или Enterprise.

3.x и более поздние

```
adcm_cluster_hadoop_v<X>_arenadata<A>_b<N>-<M><E>.tgz
```

Где:

- `<X>` — версия пакета, соответствующая версии ключевого сервиса (HDFS).
- `<A>` — последовательный номер релиза с версией `<X>`.
- `b<N>-<M>` — версия ansible-скриптов, описывающих бандл.
- `<E>` — версия издания, Community или Enterprise.

Например:

- `adcm_cluster_hadoop_v2.1.8_b1-1_enterprise.tgz` — версия пакета 2.1.8, версия кода бандла b1-1, а издание — Enterprise.
- `adcm_cluster_hadoop_v3.1.2_arenadata1_b1-enterprise.tgz` — версия пакета 3.1.2_arenadata1, первый релиз в рамках версии 3.x, версия кода бандла b1, а издание — Enterprise.

Доступные версии бандлов

Ниже приведены доступные версии бандлов ADH/ADPS.

Бандлы ADH

Версия бандла	Версия кода
2.1.4	b1 — b11
2.1.6	b1 — b4
2.1.7	b1
2.1.8	b3
2.1.10	b1
3.1.2.1	b1 — b2
3.2.4.1	b1 — b3
3.2.4.2	b1 — b2
3.2.4.3	b1

Бандлы ADPS

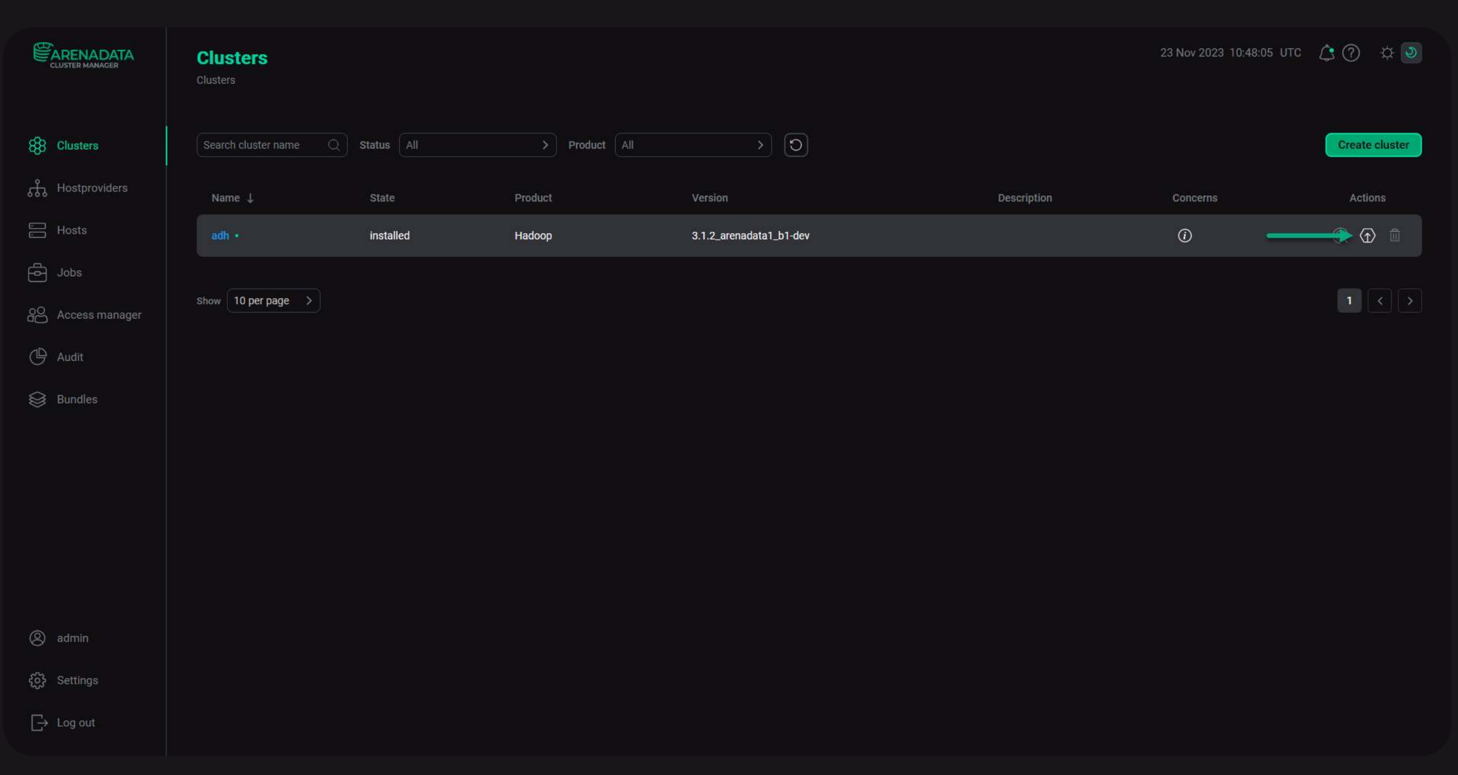
Версия бандла	Версия кода
1.0.2	b1 — b9
1.0.3	b1 — b2
1.0.4	b1 — b4
1.0.5	b1 — b2
1.1.0	b1 — b2
1.1.1	b1 — b2
1.1.2	b1
1.2.0	b1

Процесс обновления

ПРИМЕЧАНИЕ

Перед обновлением кластера убедитесь, что конфигурации на хостах соответствуют конфигурациям в ADCM. Если файлы конфигурации редактировались на хостах вручную, синхронизируйте настройки самостоятельно.

Для обновления кластера ADH необходимо загрузить новый бандл в ADCM. После загрузки ADCM распознает новый бандл и действие ***Upgrade*** станет доступным для кластера в ADCM UI, как показано ниже.



Name	State	Product	Version	Description	Concerns	Actions
adh	installed	Hadoop	3.1.2_arenadata1_b1-dev			

Действие Upgrade

ВНИМАНИЕ

При обновлении кластера ADH порядок обновления продуктов должен быть следующим:

1. ADCM
2. ADPS
3. ADH

Ниже приведены основные правила по обновлению бандлов на новую мажорную/минорную версию.

Обновление мажорной версии

Обновление мажорной версии подразумевает обновление содержимого пакетов и бандла. Обновление ADH-кластера до следующей мажорной версии возможно только с **максимальной** минорной версии предыдущего мажорного релиза. Несколько примеров обновлений приведены ниже:

- ✅ 2.1.6_b4 → 2.1.7_b1. Допустимое обновление.
- 2.1.8_b3 → 2.1.10_b1. Допустимое обновление.
- 3.1.2_arenadata1_b1 → 3.1.2_arenadata2_b1. Допустимое обновление.
- 3.1.2_arenadata2_b1 → 3.2.4_arenadata1_b1. Допустимо, если версия 3.1.2_arenadata2_b1 является последней в рамках 3.1.2.
- ❌ 2.1.6_b3 → 2.1.7_b1. Запрещено. Версия 2.1.6_b3 не является последней в релизе 2.1.6.
- 2.1.6_b4 → 2.1.8_b1. Запрещено. Нельзя "перепрыгнуть" через минорную версию 2.1.7.
- 3.1.2_arenadata1_b1 → 3.1.2_arenadata3_b1. Запрещено. Нельзя "перепрыгнуть" через версию 3.1.2_arenadata2_bX.

Обновление минорной версии

Обновление минорной версии подразумевает обновление кода бандла. Обновление минорной версии релиза можно выполнить с любой версии в рамках одного мажорного релиза, например:

- ✅ 2.1.4_b1 → 2.1.4_b2
- 2.1.4_b1 → 2.1.4_b11
- ✅ 3.1.2_arenadata1_b1 → 3.1.2_arenadata1_b5

ПРИМЕЧАНИЕ

Вышеуказанные правила обновления также применимы для ADPS-бандлов.

Добавление хоста в кластер

Альберт Багдасарян, Евгения Кузина

Назад к содержанию

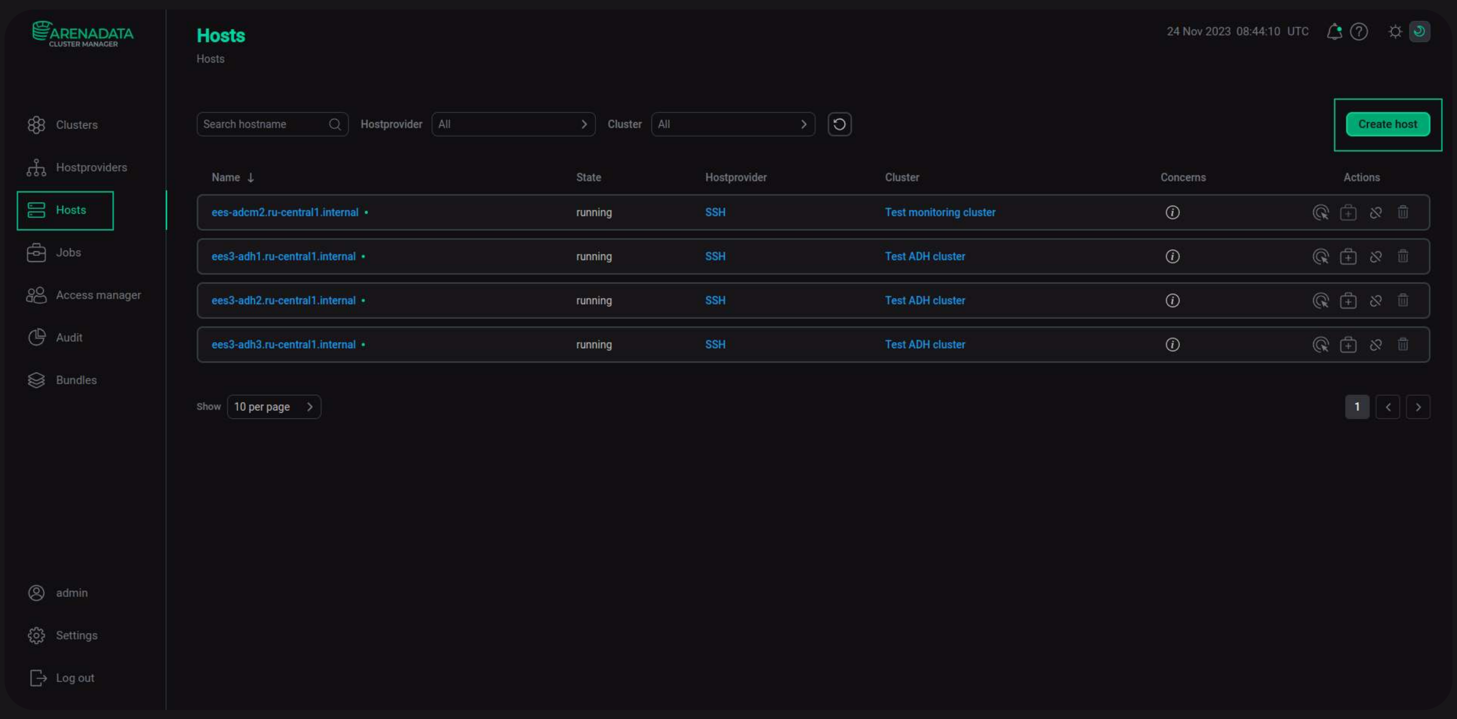
Используйте ADCM для добавления хостов в кластер ADH, как описано ниже.

ПРИМЕЧАНИЕ

Для управления кластером ADH необходимо последовательно установить Arenadata Cluster Manager (ADCM) и ADH, следуя руководству Установка Arenadata Hyperwave.

Чтобы добавить хост в кластер, выполните следующие шаги:

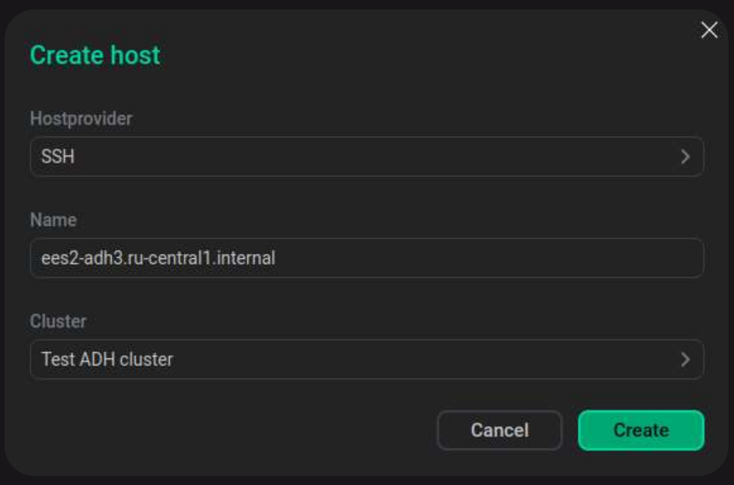
1. Выберите пункт левого навигационного меню ***Hosts*** и нажмите ***Create host***.


ARENADATA
CLUSTER MANAGER
Hosts
Hosts
24 Nov 2023 08:44:10 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search hostname
Hostprovider
All
Cluster
All
Create host
Show
10 per page
1

Name ↓	State	Hostprovider	Cluster	Concerns	Actions
ees-adcm2.ru-central1.internal •	running	SSH	Test monitoring cluster	ⓘ	
ees3-adh1.ru-central1.internal •	running	SSH	Test ADH cluster	ⓘ	
ees3-adh2.ru-central1.internal •	running	SSH	Test ADH cluster	ⓘ	
ees3-adh3.ru-central1.internal •	running	SSH	Test ADH cluster	ⓘ	

Добавление хоста

2. В открывшемся окне:
   - Выберите хостпровайдера из раскрывающегося списка в поле ***Hostprovider***. Хостпровайдеры отображаются в списке с именами, присвоенными им на этапе добавления в ADCM.
   - Введите FQDN хоста в поле ***Name***. FQDN должен содержать имя хоста и домен.
   - Выберите кластер из раскрывающегося списка в поле ***Cluster***.
   - Нажмите ***Create***.


Create host
Cancel
Create

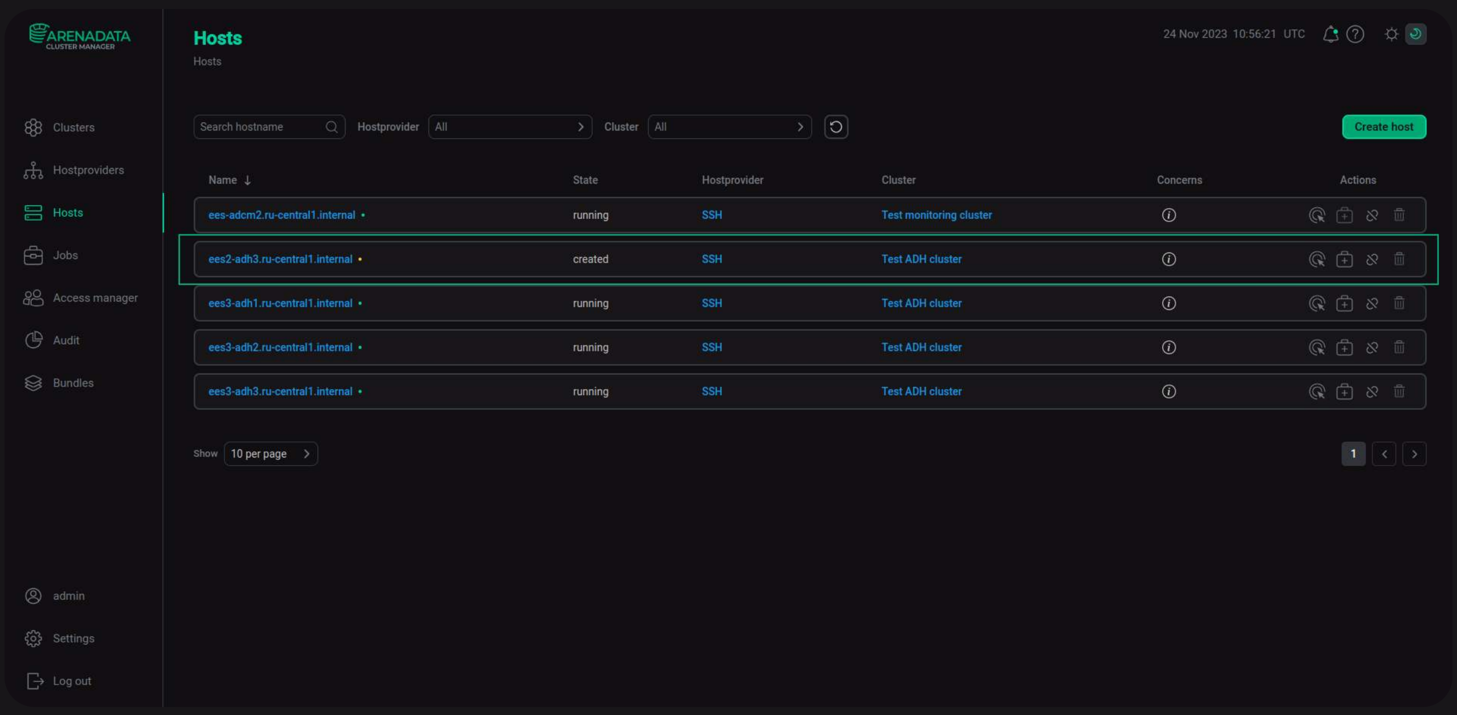
Hostprovider: SSH

Name: ees2-adh3.ru-central1.internal

Cluster: Test ADH cluster

Параметры добавляемого хоста

Новый хост отображается на странице ***Hosts***.


ARENADATA
CLUSTER MANAGER
Hosts
Hosts
24 Nov 2023 10:56:21 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search hostname
Hostprovider
All
Cluster
All
Create host
Show
10 per page
1

Name ↓	State	Hostprovider	Cluster	Concerns	Actions
ees-adcm2.ru-central1.internal •	running	SSH	Test monitoring cluster	ⓘ	
ees2-adh3.ru-central1.internal •	created	SSH	Test ADH cluster	ⓘ	
ees3-adh1.ru-central1.internal •	running	SSH	Test ADH cluster	ⓘ	
ees3-adh2.ru-central1.internal •	running	SSH	Test ADH cluster	ⓘ	
ees3-adh3.ru-central1.internal •	running	SSH	Test ADH cluster	ⓘ	

Страница "Hosts" с новым хостом

Проверка состояния хоста

Альберт Багдасарян, Евгения Кузина

Содержание

Назад к содержанию

Проверка состояния на странице Hosts

Проверка состояния хоста через его профиль

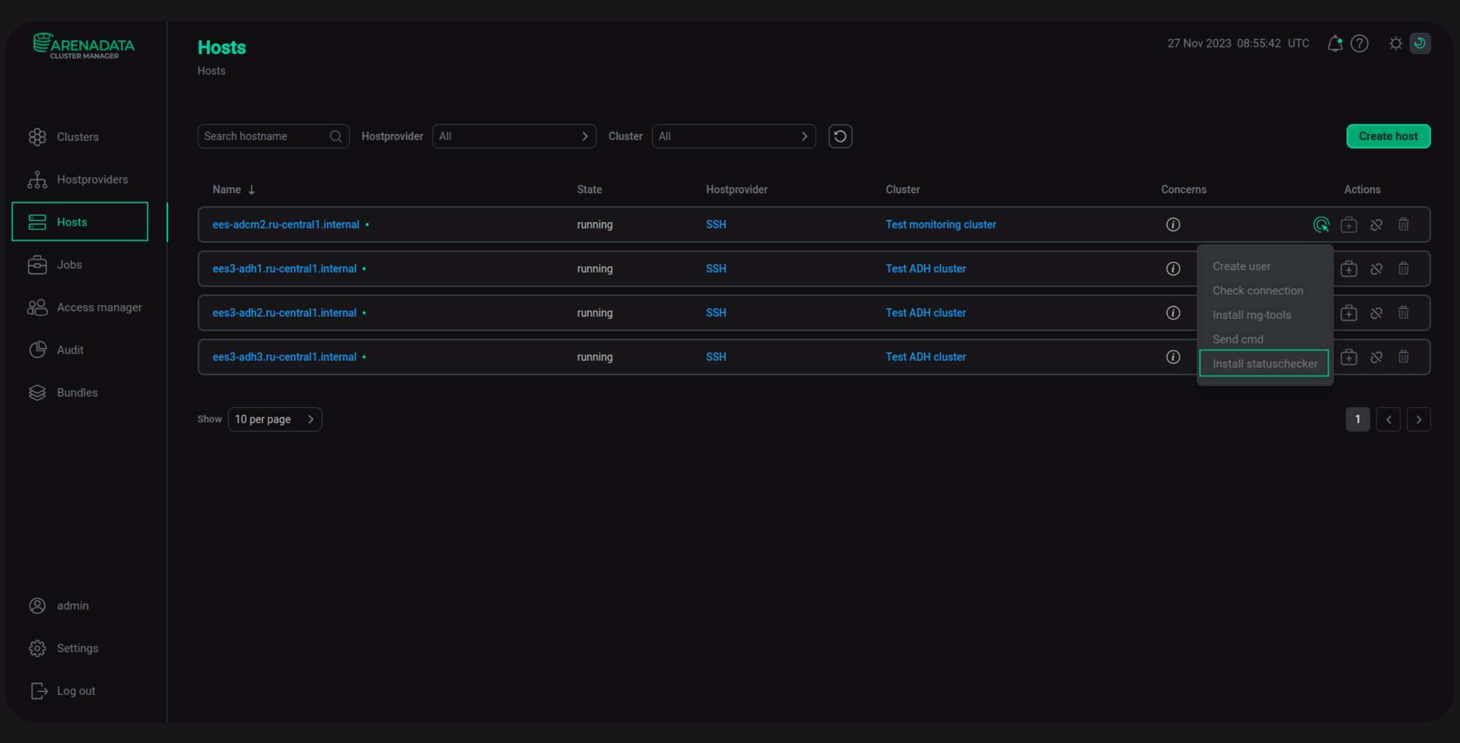
Вы можете проверить состояние хоста в ADCM, используя следующие возможности:

- общий список на странице ***Hosts***;
- страница с индивидуальным профилем хоста.

Проверка состояния на странице Hosts

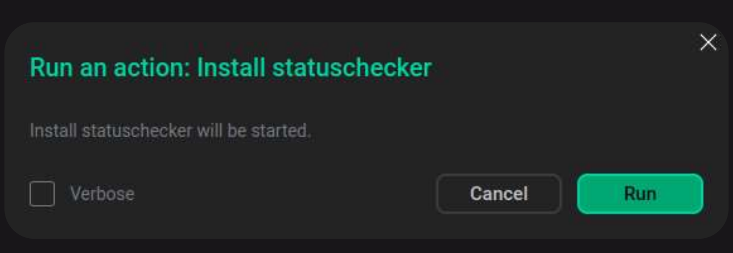
Выполните следующие действия, чтобы просмотреть состояние хоста:

1. Выберите пункт левого навигационного меню ***Hosts*** в ADCM.
2. Если вы ранее не устанавливали утилиту statuschecker, нажмите в столбце ***Actions*** и в раскрывающемся списке выберите действие ***Install statuschecker***.



Установка statuschecker

3. Кликните ***Run*** для подтверждения установки. Статус этой задачи вы можете отслеживать на странице ***Jobs***.



Run an action: Install statuschecker

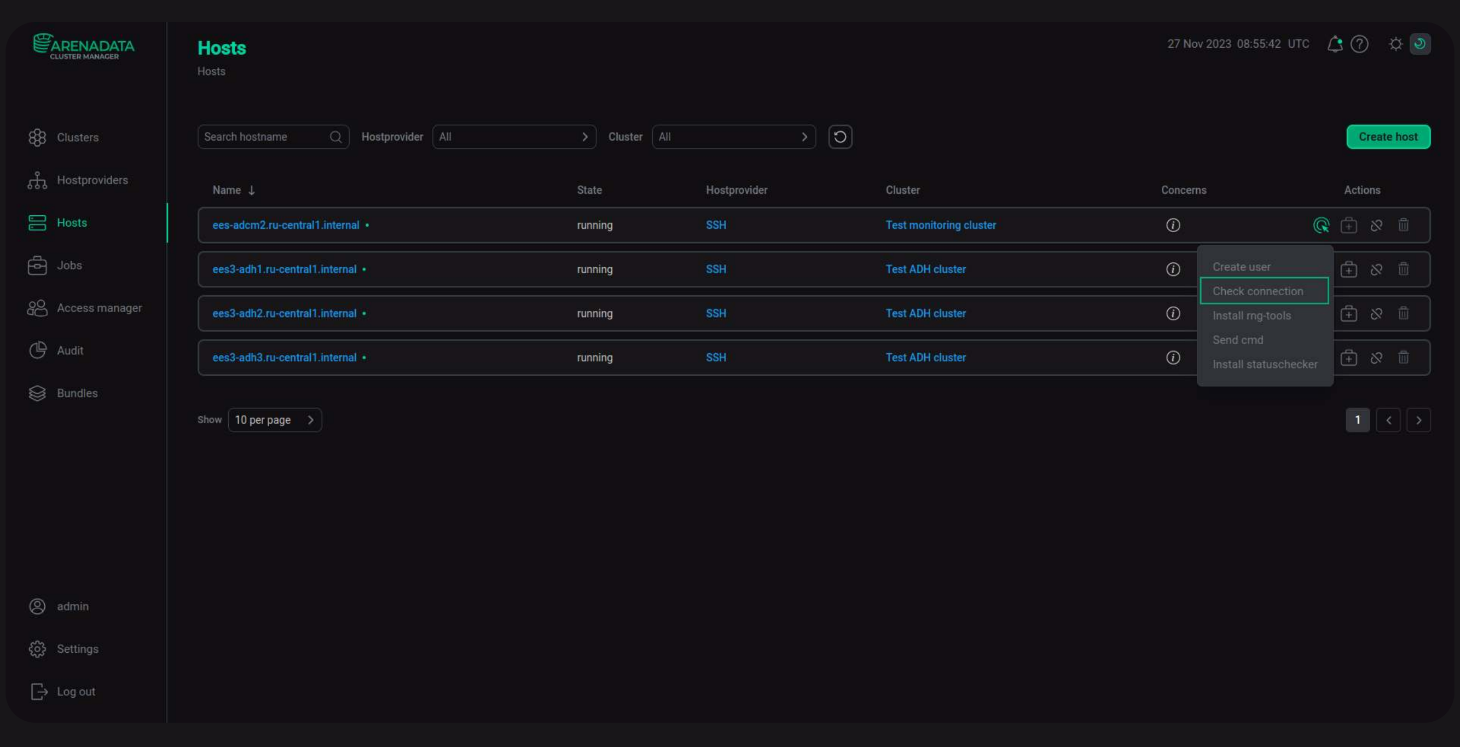
Install statuschecker will be started.

☐ Verbose

Cancel Run

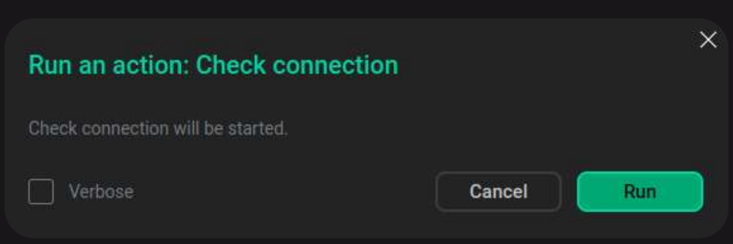
Подтверждение запроса на установку утилиты

4. Таким же образом запросите проверку соединения, кликнув и выбрав действие ***Check connection***.



Запуск проверки соединения

5. В открывшемся окне нажмите ***Run*** для подтверждения.



Run an action: Check connection

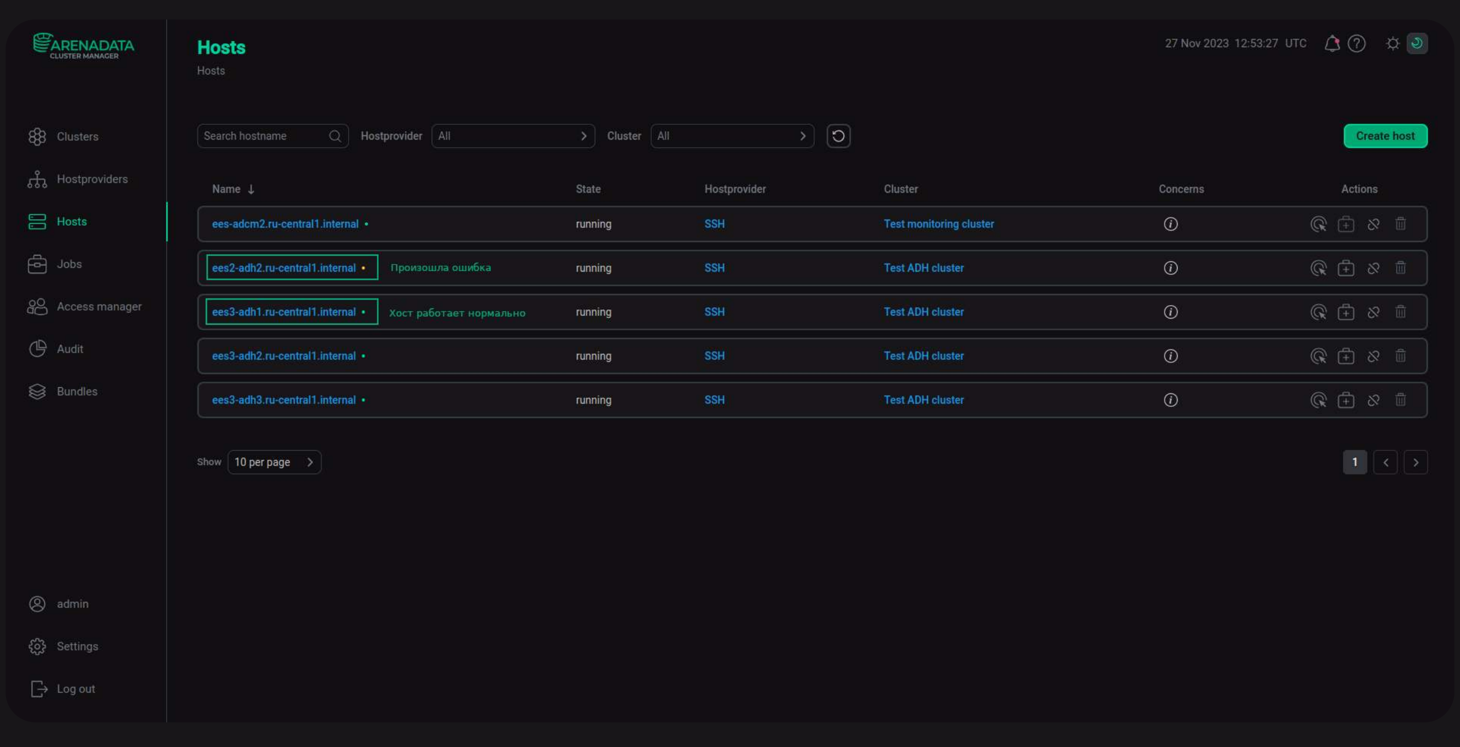
Check connection will be started.

☐ Verbose

Cancel Run

Подтверждение запроса на проверку соединения

6. Проверьте статус хоста. Если хост доступен, рядом с именем хоста отображается зеленая точка. Если в столбце ***Name*** присутствует желтая точка, это означает наличие ошибок. Подробности можно найти на странице ***Jobs***.



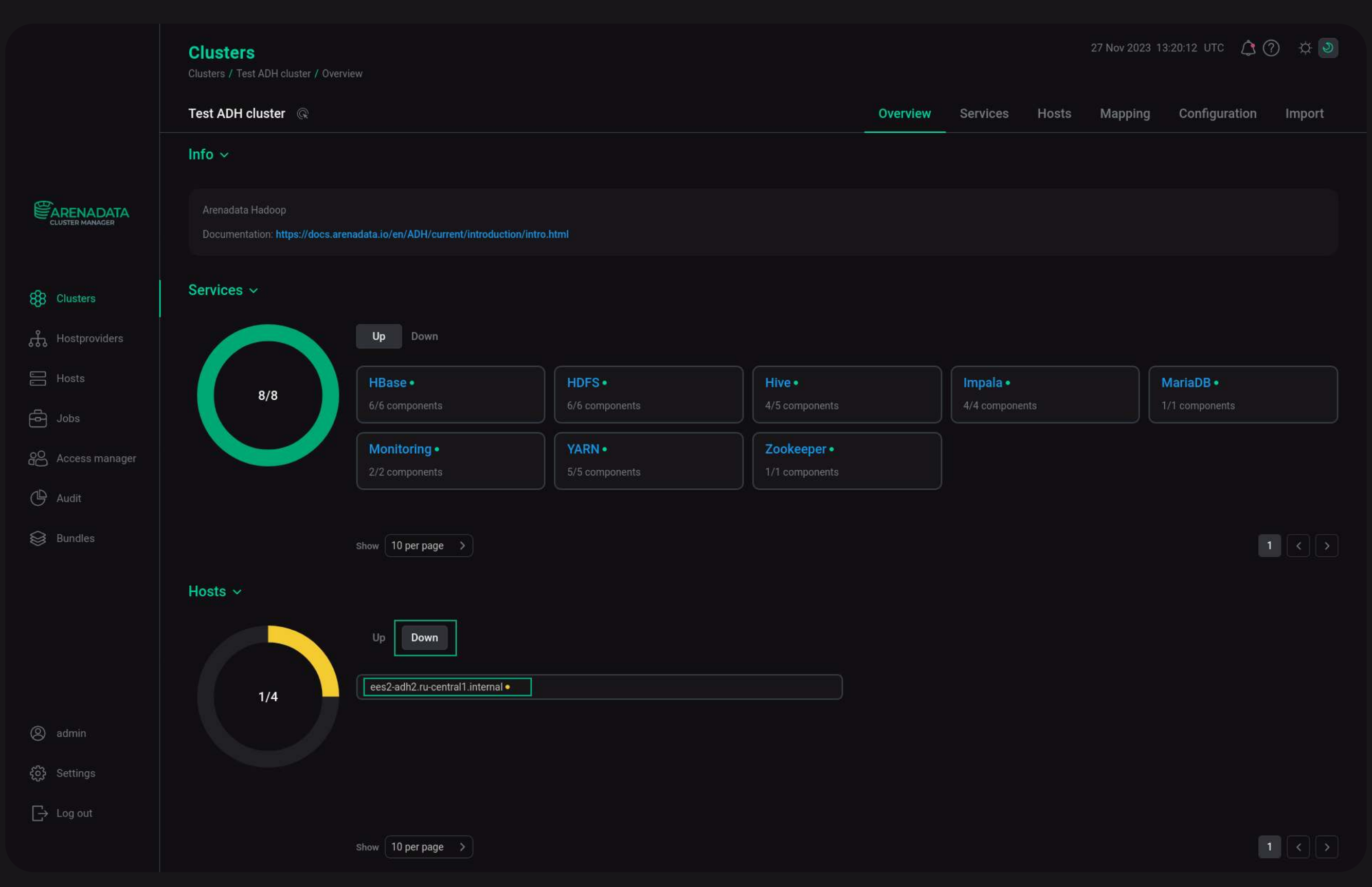
Обнаружение ошибок в работе хоста

Проверка состояния хоста через его профиль

Выполните следующие действия, чтобы просмотреть состояние хоста:

1. Кликните нужный хост рядом с диаграммой ***Hosts*** на вкладке кластера ***Overview***. Хосты кластера распределены по двум вкладкам:
   - ***Up*** — хосты, которые работают корректно;
   - ***Down*** — недоступные хосты.

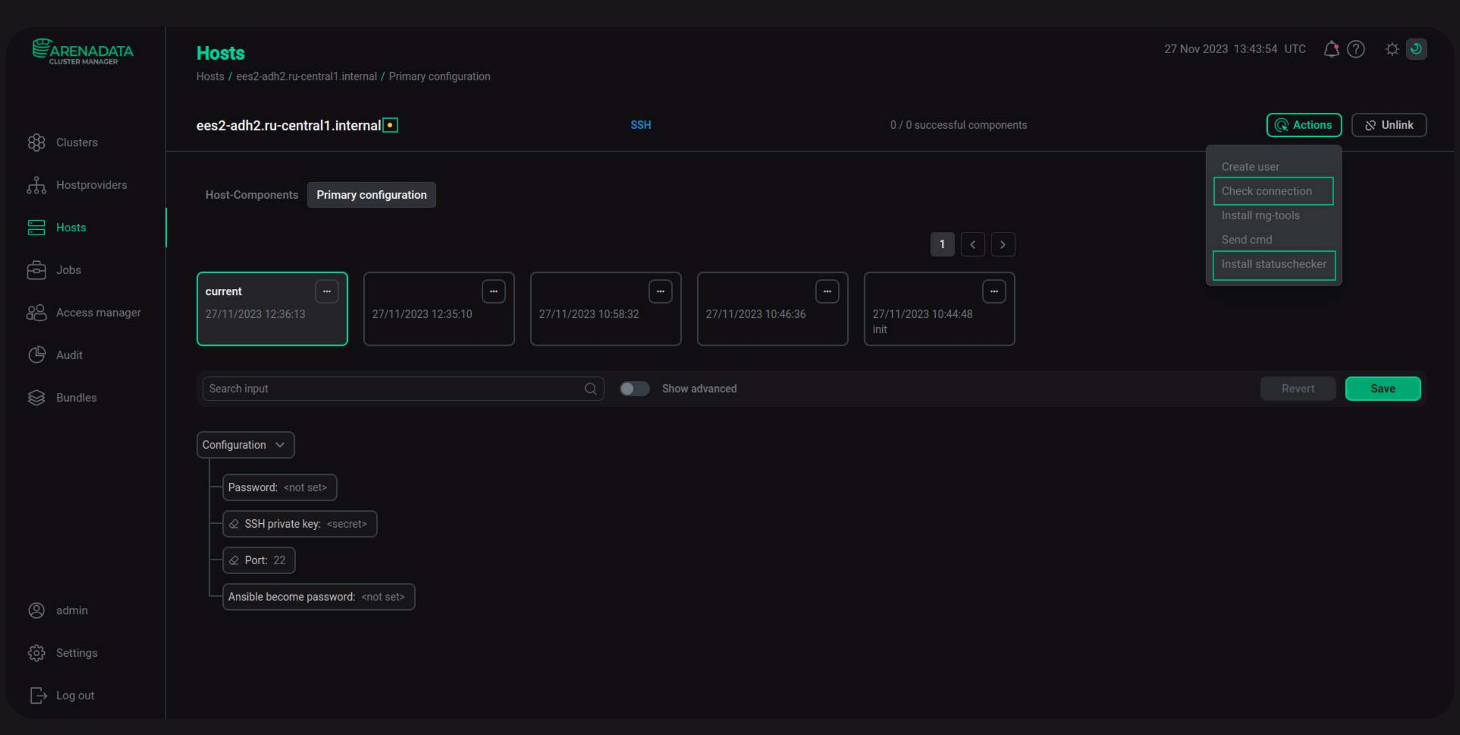
   Для просмотра хостов, которые работают неправильно, переключитесь на вкладку ***Down***.



Переход к настройкам хоста

   Откроется страница с настройками хоста.

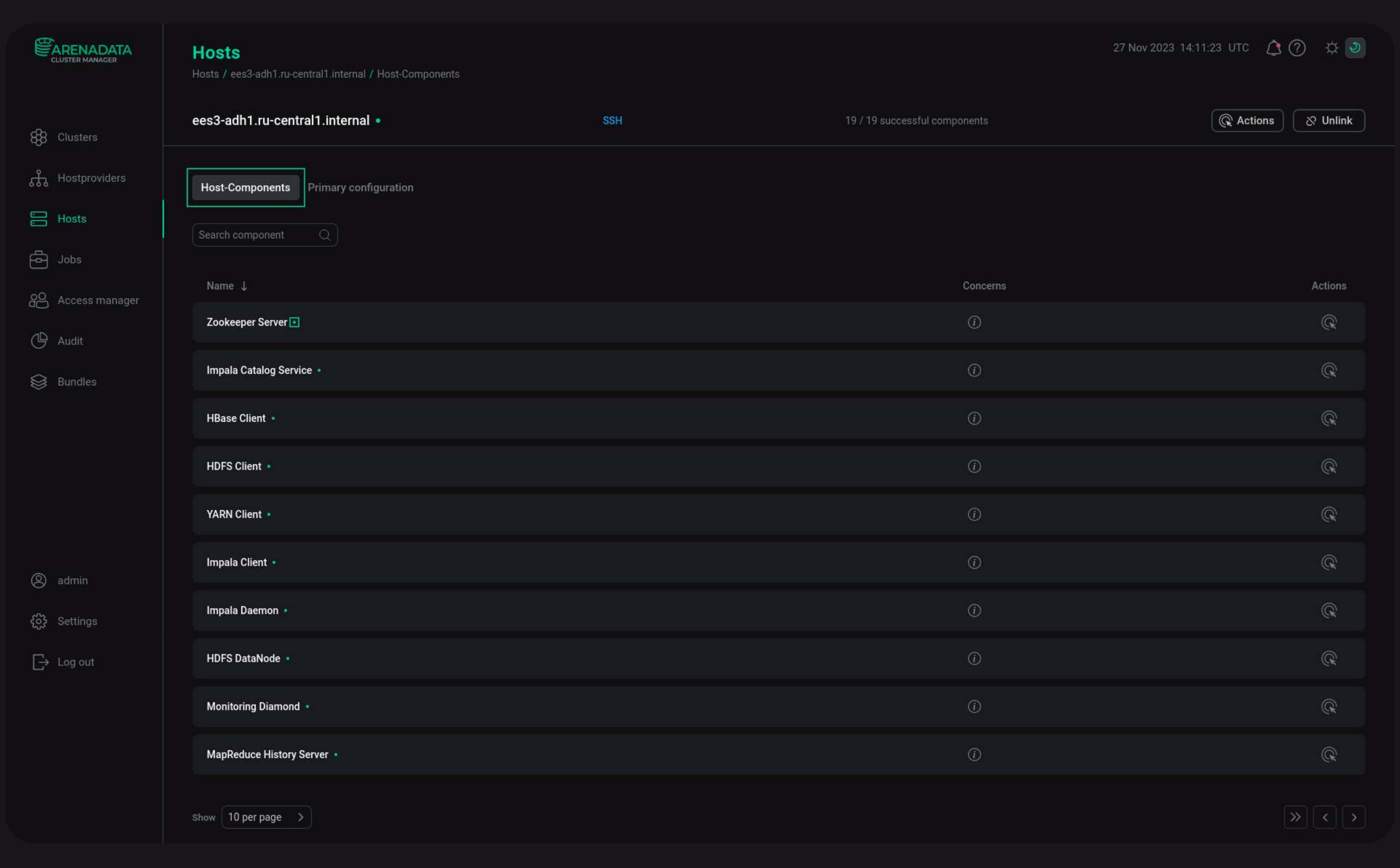
2. Если вы ранее не устанавливали утилиту statuschecker, нажмите ***Actions*** в верхнем правом углу экрана. Появившийся раскрывающийся список содержит те же действия, что и столбец ***Actions*** на странице ***Hosts***. Установите statuschecker и проверьте соединение, как описано выше.



Настройки хоста

   Точка, отображаемая рядом с именем хоста, определяет статус хоста. Зелёная точка указывает на то, что хост работает корректно, жёлтая — возникла проблема. Дополнительную информацию об этой проблеме можно найти на странице ***Jobs***.

3. На вкладке ***Host-Components*** можно получить дополнительные сведения о текущем хосте:
   - Все компоненты установлены на хосте.
   - Статус каждого компонента.



Компоненты хоста и их статусы

Зеленая точка в столбце ***Name*** показывает, что компонент работает корректно, желтая — возникла проблема.

Удаление хоста из кластера

Сергей Тихомиров

Содержание

Назад к содержанию

Шаг 1. Подготовка хоста

Шаг 2. Удаление компонентов с хоста

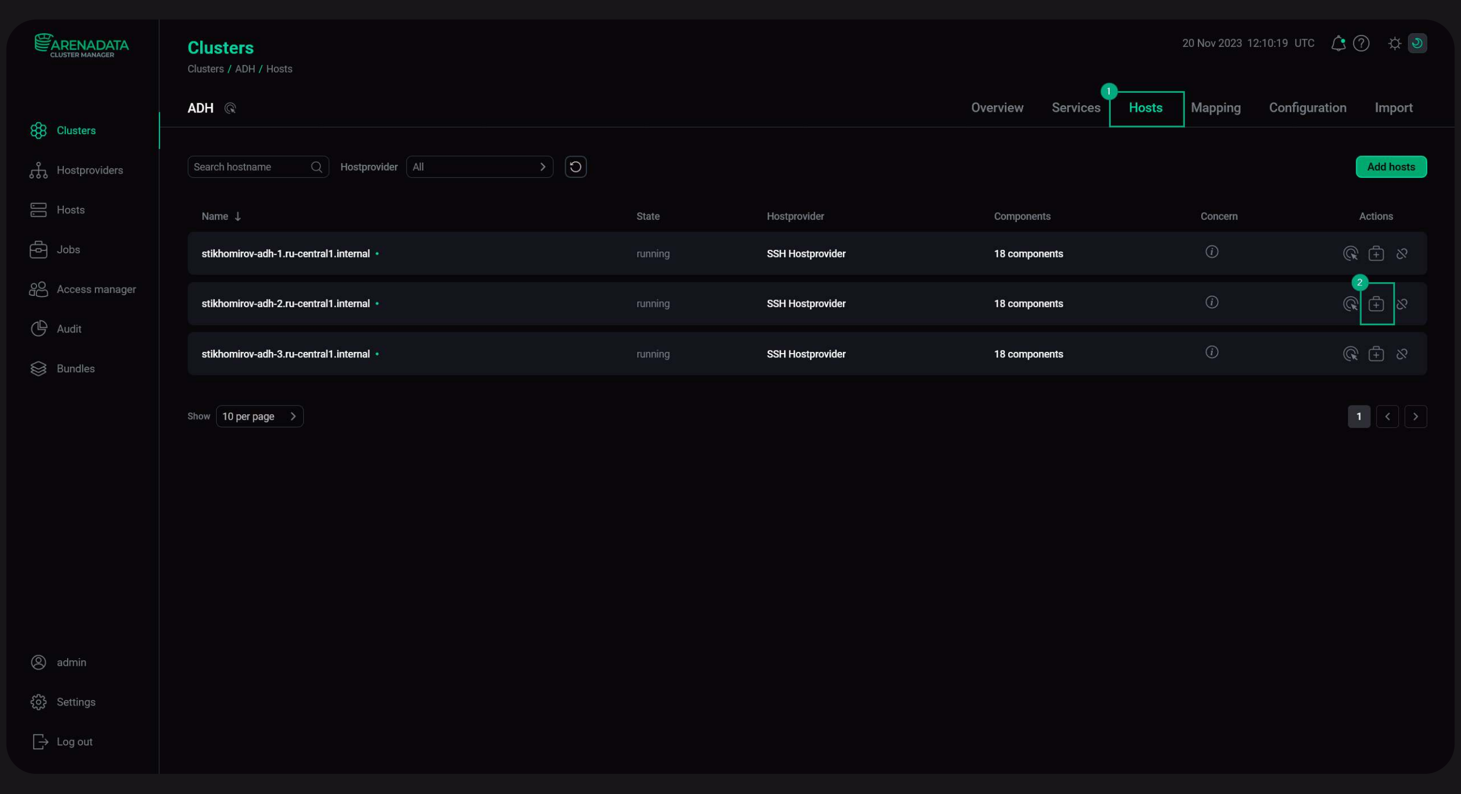
Шаг 3. Удаление хоста

Удаление хоста из кластера — нетривиальная операция, которую приходится выполнять, когда хост выходит из строя. Обычно при выполнении действий ***Remove***, ***Move*** и ***Decommiss*** на недоступных хостах появляются ошибки, сообщающие о недоступности хоста. Для игнорирования таких ошибок был разработан режим обслуживания (maintenance mode).

Чтобы удалить узел из кластера, следуйте шагам ниже.

Шаг 1. Подготовка хоста

1. Включите режим обслуживания на хосте, который хотите удалить. Это можно сделать на странице ***Hosts*** или на вкладке ***Hosts*** на странице выбранного кластера, кликнув на иконку ⊞ .



ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH / Hosts
20 Nov 2023 12:10:19 UTC
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
1
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
Search hostname
Hostprovider
All
Add hosts
Name
State
Hostprovider
Components
Concern
Actions
stikhomirov-adh-1.ru-central1.internal
running
SSH Hostprovider
18 components
stikhomirov-adh-2.ru-central1.internal
running
SSH Hostprovider
18 components
2
stikhomirov-adh-3.ru-central1.internal
running
SSH Hostprovider
18 components
Show
10 per page
admin
Settings
Log out

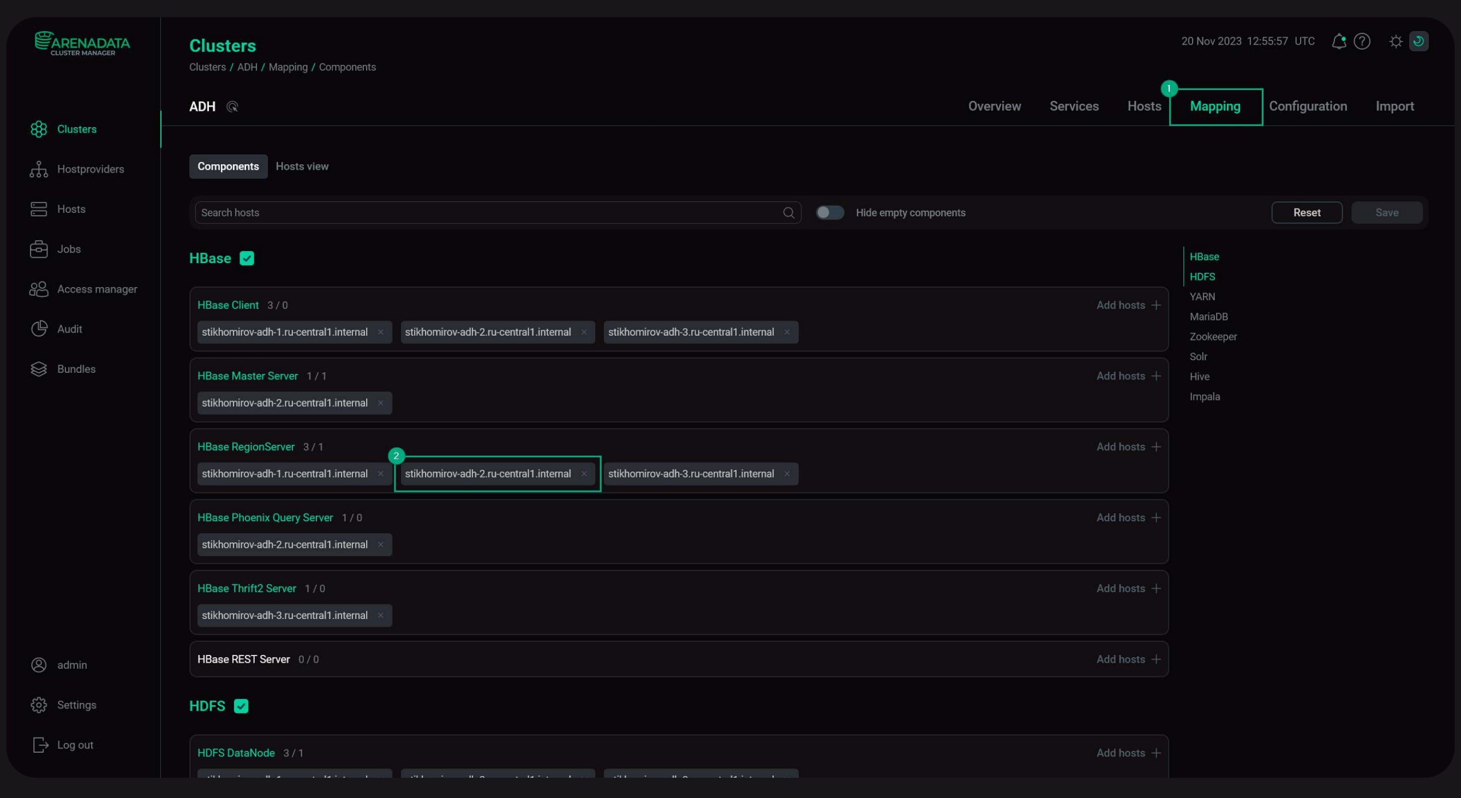
Включение режима обслуживания

2. Убедитесь, что фактор репликации в HDFS имеет значение не меньше, чем количество активных хостов с DataNode. В противном случае, добавьте новый хост с DataNode или уменьшите значение параметра `dfs.replication`. Если этого не сделать, то NameNode перейдет в безопасный режим (safe mode), из-за чего упадут DataNode и зависимые компоненты (HBase RegionServer, Tez и т.д.).

Шаг 2. Удаление компонентов с хоста

Для удаления всех компонентов с хоста повторите следующие шаги для каждого компонента:

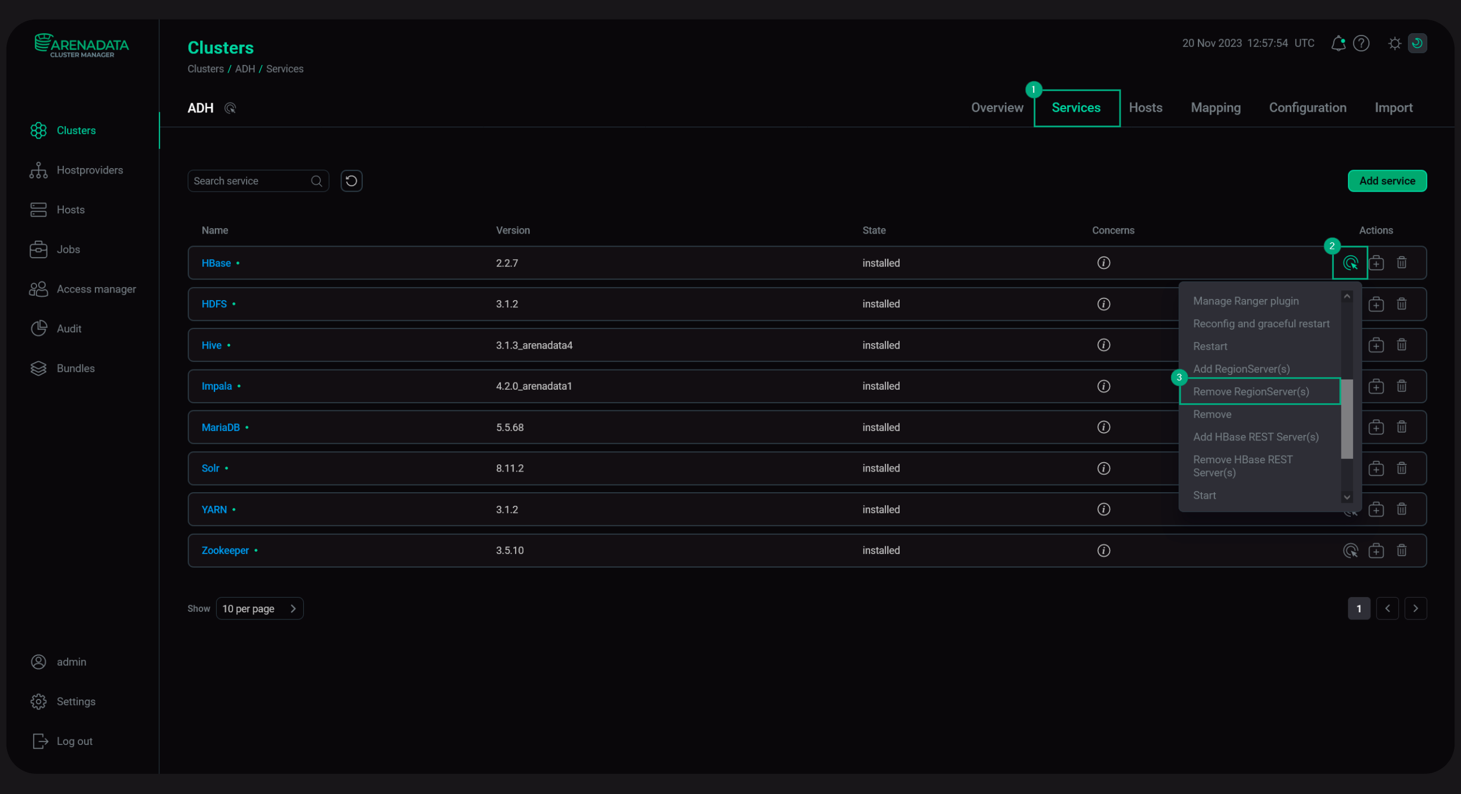
1. Проверьте в разделе ***Mapping***, есть ли компонент на хосте, вышедшем из строя.



Clusters
Clusters / ADH / Mapping / Components
20 Nov 2023 12:55:57 UTC
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
1
Components
Hosts view
Search hosts
Hide empty components
Reset
Save
HBase
HBase Client 3 / 0
HBase Master Server 1 / 1
HBase RegionServer 3 / 1
2
HBase Phoenix Query Server 1 / 0
HBase Thrift2 Server 1 / 0
HBase REST Server 0 / 0
HDFS
HDFS DataNode 3 / 1
Add hosts
stikhomirov-adh-1.ru-central1.internal
stikhomirov-adh-2.ru-central1.internal
stikhomirov-adh-3.ru-central1.internal
HBase
HDFS
YARN
MariaDB
Zookeeper
Solr
Hive
Impala

Раздел Mapping

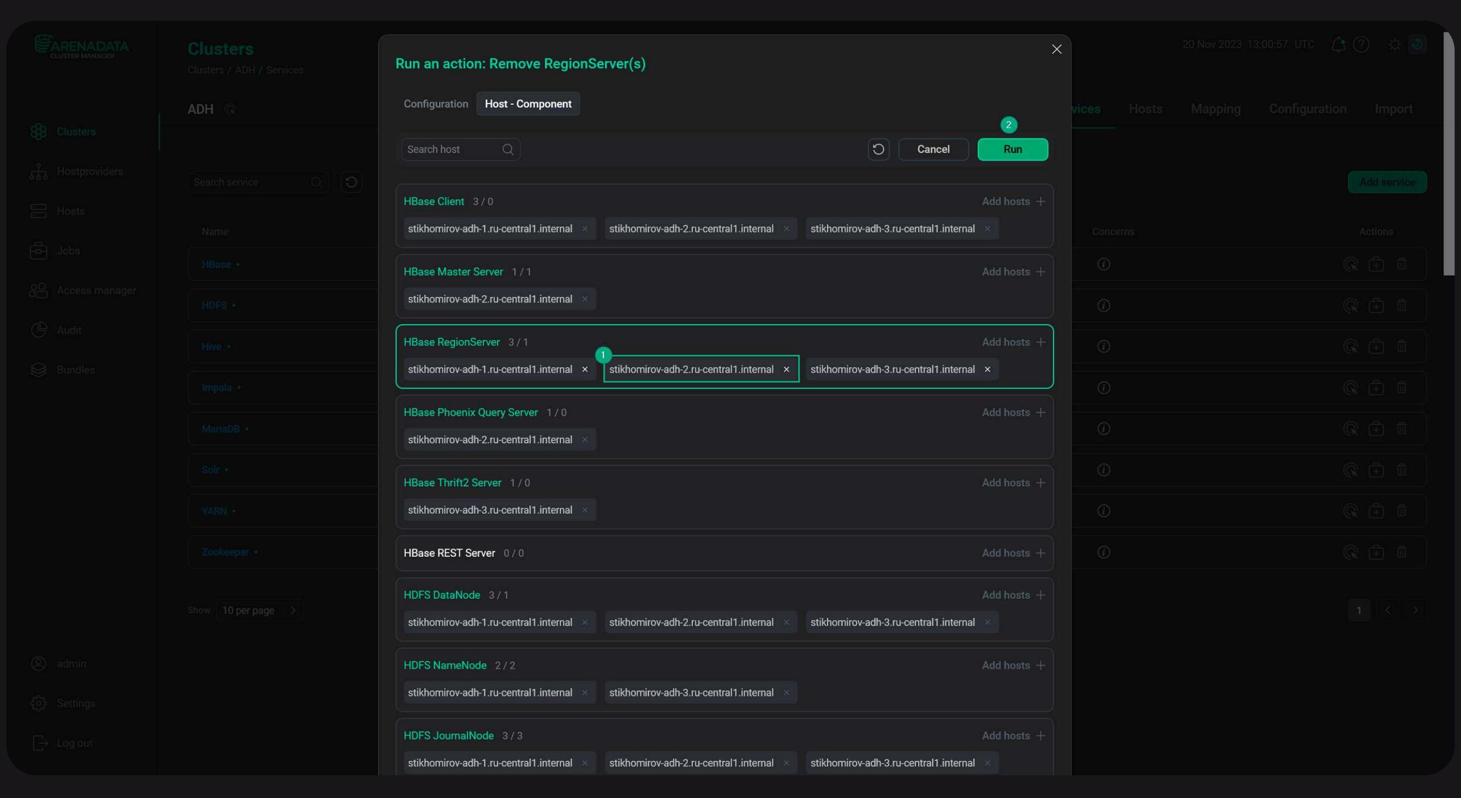
2. Если компонент размещен на недоступном хосте, перейдите в раздел ***Services***, найдите соответствующий компоненту сервис и запустите действие удаления компонента.



Clusters
Clusters / ADH / Services
20 Nov 2023 12:57:54 UTC
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
1
Search service
Add service
Name
Version
State
Concerns
Actions
HBase
2.2.7
HDFS
3.1.2
Hive
3.1.3_arenadata4
Impala
4.2.0_arenadata1
MariaDB
5.5.68
Solr
8.11.2
YARN
3.1.2
Zookeeper
3.5.10
installed
2
Manage Ranger plugin
Reconfig and graceful restart
Restart
Add RegionServer(s)
3
Remove RegionServer(s)
Remove
Add HBase REST Server(s)
Remove HBase REST Server(s)
Start
Show
10 per page

Меню действий сервиса

3. В появившемся окне нажмите на ✕ возле названия недоступного хоста. Затем кликните ***Run*** и дождитесь окончания операции.

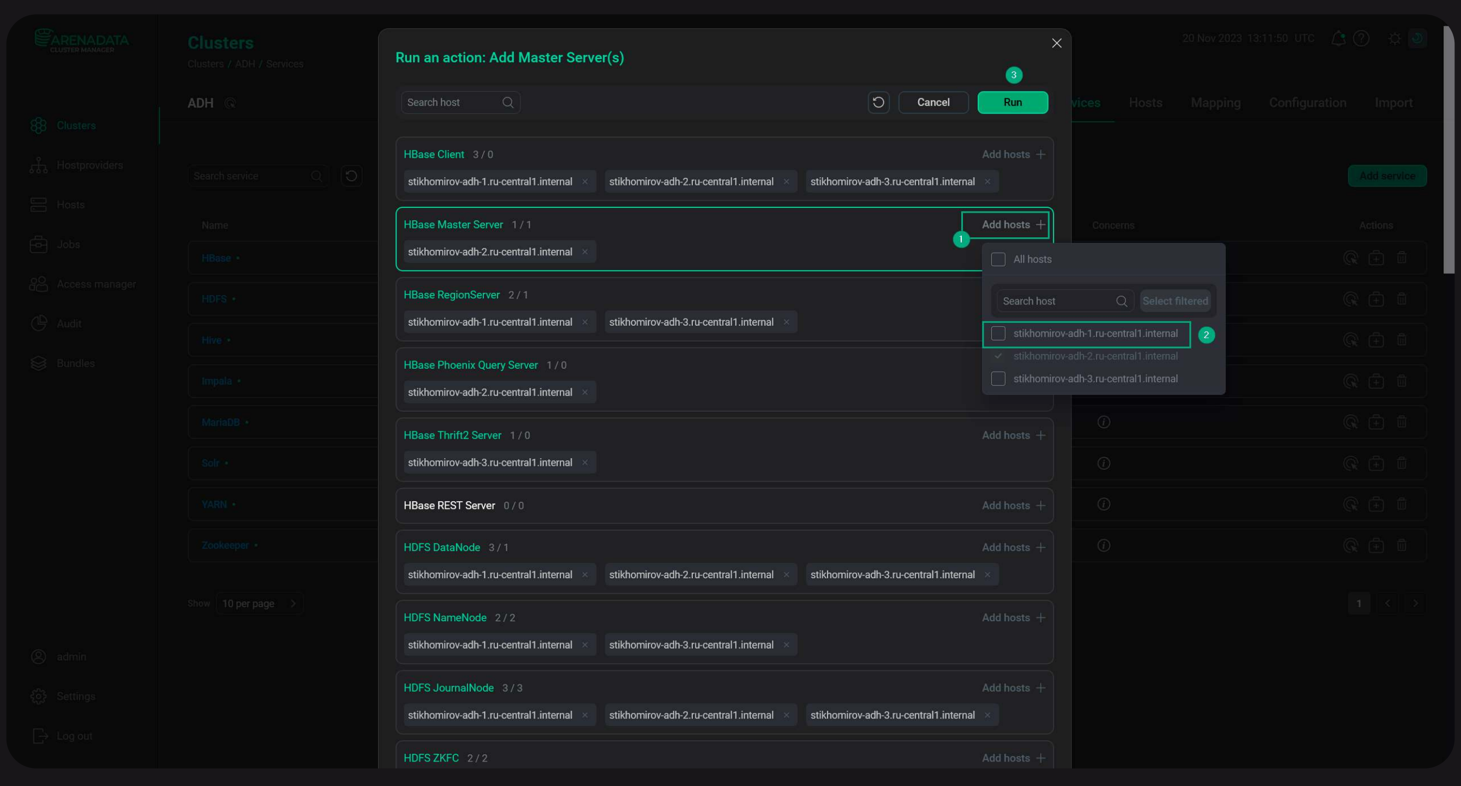


Run an action: Remove RegionServer(s)
Configuration
Host - Component
Search host
Cancel
Run
2
HBase Client 3 / 0
HBase Master Server 1 / 1
HBase RegionServer 3 / 1
1
HBase Phoenix Query Server 1 / 0
HBase Thrift2 Server 1 / 0
HBase REST Server 0 / 0
HDFS DataNode 3 / 1
HDFS NameNode 2 / 2
HDFS JournalNode 3 / 3
Add hosts
stikhomirov-adh-1.ru-central1.internal
stikhomirov-adh-2.ru-central1.internal
stikhomirov-adh-3.ru-central1.internal

Удаление компонента

Особые случаи

- Некоторые компоненты нельзя удалить пока они не были добавлены на другой хост (например, HBase Master Server). В таком случае для них следует запустить действие ***Add*** в меню действий соответствующего сервиса, выбрав дополнительный хост. После окончания операции можно удалить компонент с недоступного хоста.



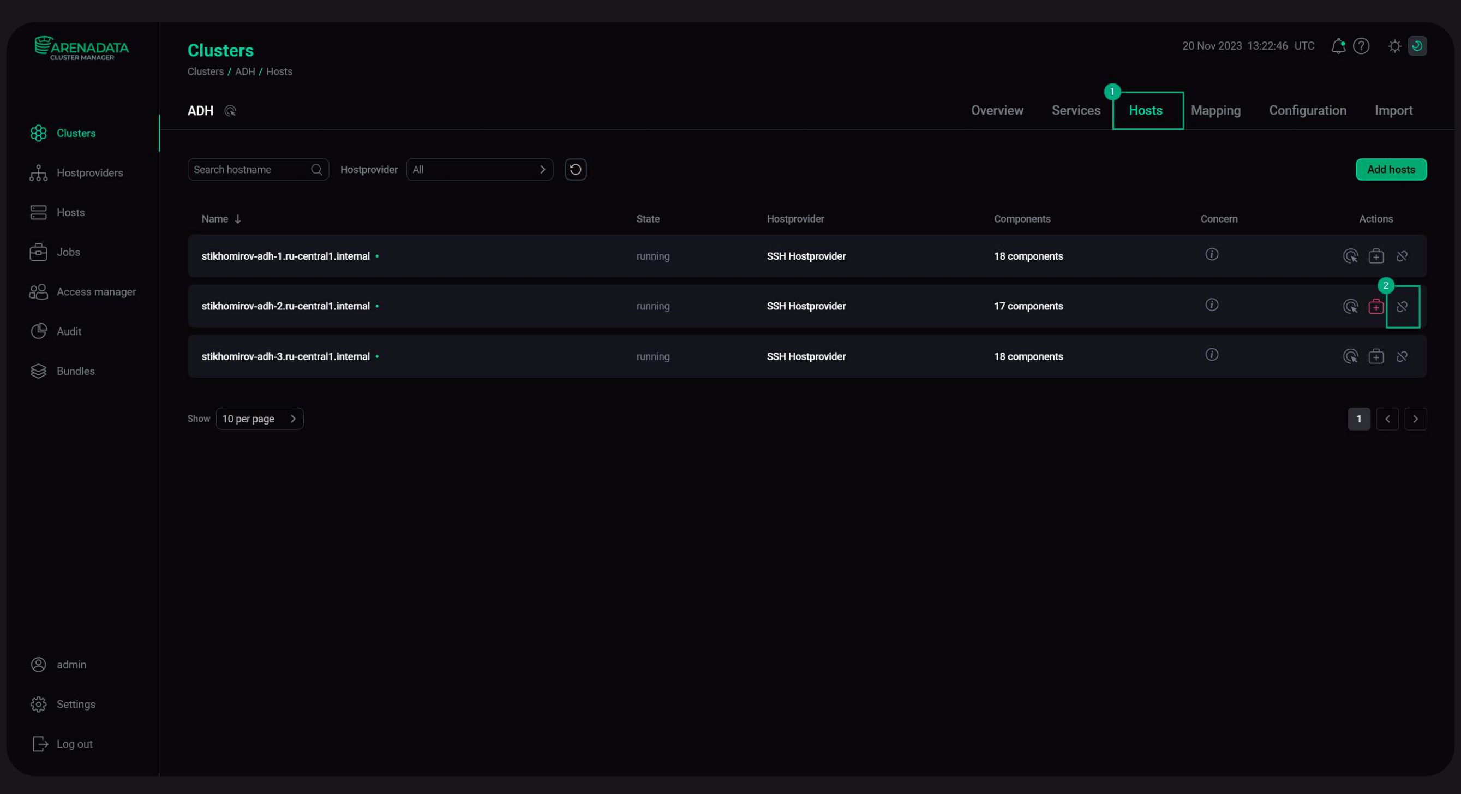
Run an action: Add Master Server(s)
Search host
Cancel
Run
3
HBase Client 3 / 0
HBase Master Server 1 / 1
Add hosts
1
All hosts
Search host
Select filtered
stikhomirov-adh-1.ru-central1.internal
2
stikhomirov-adh-2.ru-central1.internal
stikhomirov-adh-3.ru-central1.internal
HBase RegionServer 2 / 1
HBase Phoenix Query Server 1 / 0
HBase Thrift2 Server 1 / 0
HBase REST Server 0 / 0
HDFS DataNode 3 / 1
HDFS NameNode 2 / 2
HDFS JournalNode 3 / 3
HDFS ZKFC 2 / 2

Добавление компонента

- У некоторых компонентов допускается только один инстанс (instance) в кластере (например, MapReduce History Server). Если такой компонент оказался на недоступном хосте, то для него нужно выполнить действие ***Move*** в меню действий соответствующего сервиса. Когда хост находится в режиме обслуживания, для него допускается только удаление компонентов, поэтому для выполнения действия ***Move*** стоит вывести хост из режима обслуживания, а по завершении снова активировать его.
- Если удаляемый компонент это HDFS DataNode или YARN NodeManager, то перед удалением его необходимо вывести из эксплуатации. Это можно сделать запустив действия ***Maintenance/Decommission DataNodes*** и ***Decommission/Recommission NodeManagers*** в меню действий HDFS и YARN, соответственно.
- Часть сервисов требуют перезапуска после перемещения некоторых компонентов. Так, при перемещении компонентов YARN следует перезапустить Hive и Spark; после перемещения HDFS NameNode — HBase; после перемещения Hive Metastore — Spark; после перемещения, удаления или добавления Zookeeper Server — все HA-сервисы (Hive, YARN, Flink).

Шаг 3. Удаление хоста

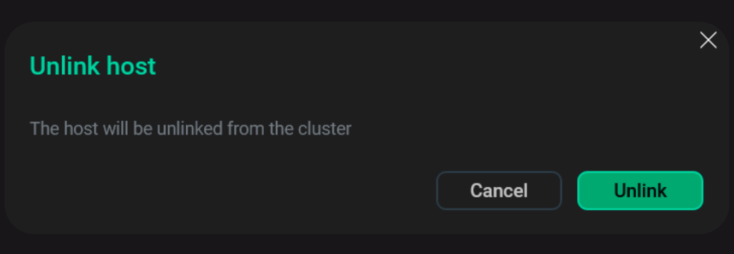
После освобождения хоста от компонентов его можно удалить из кластера в разделе ***Hosts***, нажав на иконку ⊘ .



Clusters
Clusters / ADH / Hosts
20 Nov 2023 13:22:46 UTC
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
1
Search hostname
Hostprovider
All
Add hosts
Name
State
Hostprovider
Components
Concern
Actions
stikhomirov-adh-1.ru-central1.internal
running
SSH Hostprovider
18 components
stikhomirov-adh-2.ru-central1.internal
running
SSH Hostprovider
17 components
2
stikhomirov-adh-3.ru-central1.internal
running
SSH Hostprovider
18 components
Show
10 per page

Удаление хоста

Затем нужно подтвердить действие, нажав ***Unlink*** в появившемся окне.



Unlink host

The host will be unlinked from the cluster

Cancel
Unlink

Подтверждение действия

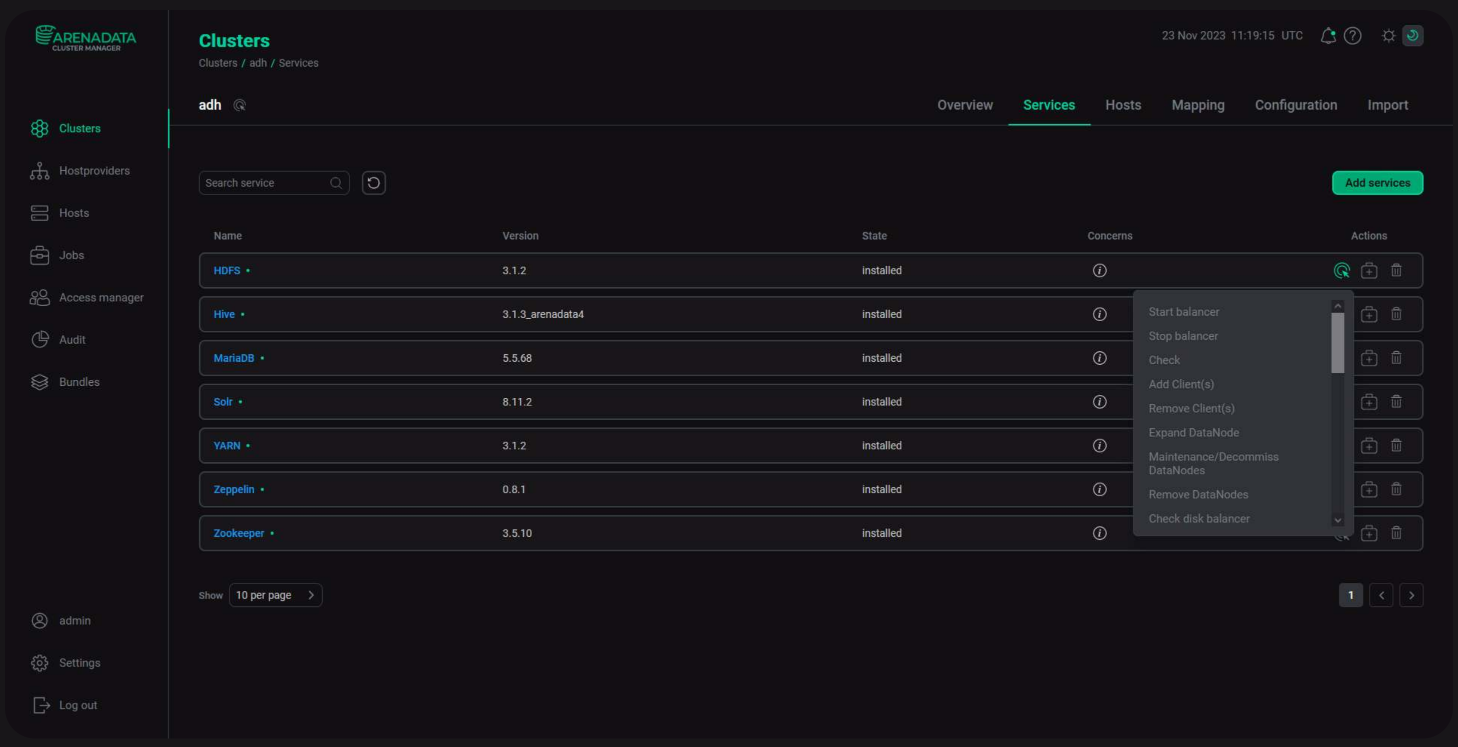
Действия с сервисами ADH

Константин Алпашкин

Назад к содержанию

ADCM позволяет взаимодействовать с кластером/хостами/сервисами, используя действия (actions). Чтобы выполнить действие для сервиса:

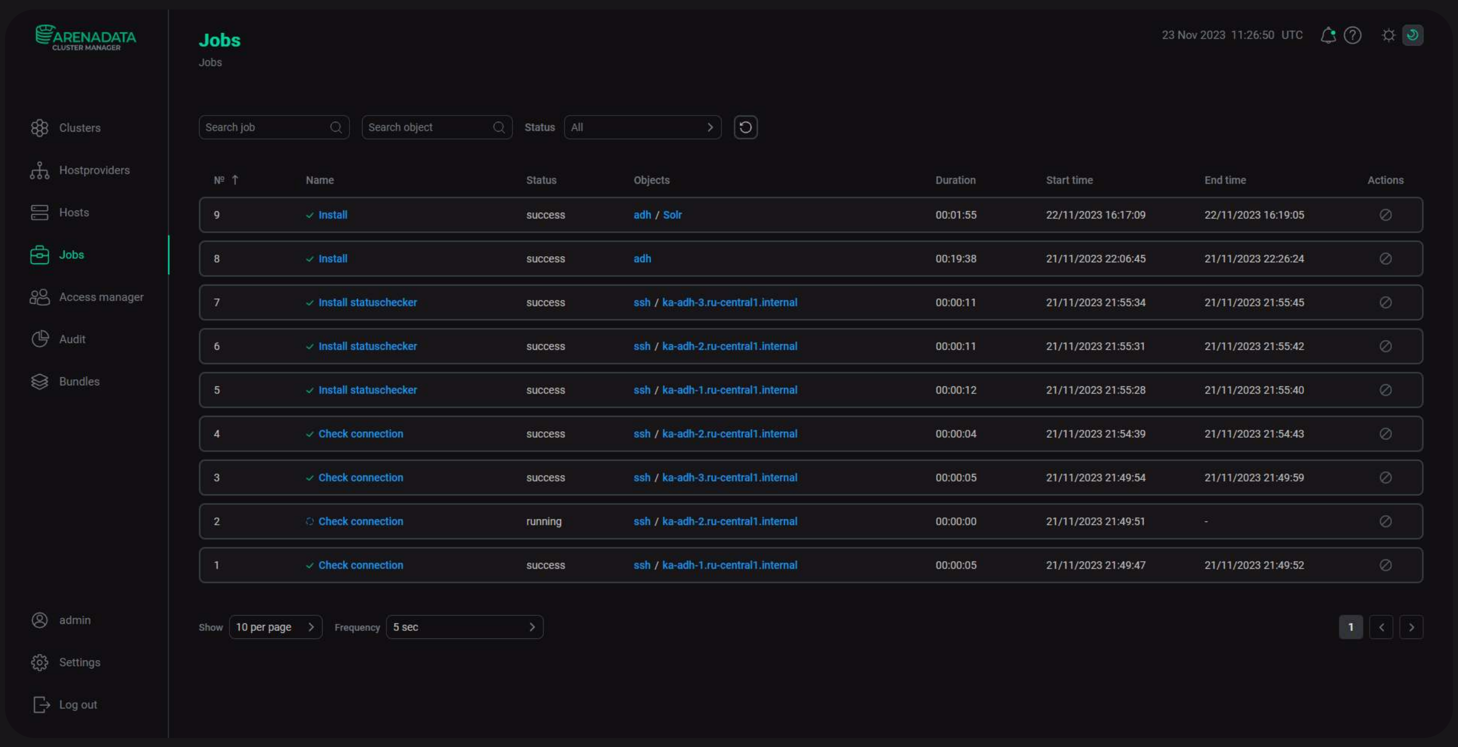
1. Выберите нужный кластер на странице ***Clusters***.
2. На открывшейся странице кликните ***Services***.
3. Кликните ***Actions*** в строке с нужным сервисом. В выпадающем меню отобразится список действий, доступных для выбранного сервиса.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters / adh / Services
23 Nov 2023 11:19:15 UTC
adh
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
Search service
Add services
Start balancer
Stop balancer
Check
Add Client(s)
Remove Client(s)
Expand DataNode
Maintenance/Decommiss DataNodes
Remove DataNodes
Check disk balancer
Show
10 per page
1
admin
Settings
Log out

Name	Version	State	Concerns	Actions
HDFS	3.1.2	installed		
Hive	3.1.3_arenadata4	installed		
MariaDB	5.5.68	installed		
Solr	8.11.2	installed		
YARN	3.1.2	installed		
Zeppelin	0.8.1	installed		
Zookeeper	3.5.10	installed		

Выполнение действия для сервиса

Прогресс и результаты всех действий доступны на странице ***Jobs*** как показано ниже.


ARENADATA
CLUSTER MANAGER
Jobs
Jobs
23 Nov 2023 11:26:50 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
Search job
Search object
Status
All
Show
10 per page
Frequency
5 sec
1
admin
Settings
Log out

№ ↑	Name	Status	Objects	Duration	Start time	End time	Actions
9	✓ Install	success	adh / Solr	00:01:55	22/11/2023 16:17:09	22/11/2023 16:19:05	⊘
8	✓ Install	success	adh	00:19:38	21/11/2023 22:06:45	21/11/2023 22:26:24	⊘
7	✓ Install statuschecker	success	ssh / ka-adh-3.ru-central1.internal	00:00:11	21/11/2023 21:55:34	21/11/2023 21:55:45	⊘
6	✓ Install statuschecker	success	ssh / ka-adh-2.ru-central1.internal	00:00:11	21/11/2023 21:55:31	21/11/2023 21:55:42	⊘
5	✓ Install statuschecker	success	ssh / ka-adh-1.ru-central1.internal	00:00:12	21/11/2023 21:55:28	21/11/2023 21:55:40	⊘
4	✓ Check connection	success	ssh / ka-adh-2.ru-central1.internal	00:00:04	21/11/2023 21:54:39	21/11/2023 21:54:43	⊘
3	✓ Check connection	success	ssh / ka-adh-3.ru-central1.internal	00:00:05	21/11/2023 21:49:54	21/11/2023 21:49:59	⊘
2	Check connection	running	ssh / ka-adh-2.ru-central1.internal	00:00:00	21/11/2023 21:49:51	-	⊘
1	✓ Check connection	success	ssh / ka-adh-1.ru-central1.internal	00:00:05	21/11/2023 21:49:47	21/11/2023 21:49:52	⊘

Страница Jobs

На этой странице доступны детали выполнения всех действий, включая подробный STDOUT/STDERR вывод Ansible. Чтобы увидеть детали выполнения действия, кликните на имя действия на странице ***Jobs***. Логи действий хранятся определенное время, которое можно изменять в настройках ADCM (раздел Data Retention Policy).

РЕКОМЕНДАЦИЯ

Для некоторых действий доступна опция ***Verbose***, которая позволяет генерировать более детальные логи выполнения действия.

Для получения информации о действиях, доступных для каждого сервиса ADH, перейдите по следующим ссылкам:

- Airflow
- Flink
- HBase
- HDFS
- Hive
- Impala
- Ozone
- Solr
- Spark3
- Trino
- YARN
- Zeppelin
- ZooKeeper

Обзор Arenadata Platform Security

Дарья Барышева

Назад к содержанию

Arenadata Platform Security (ADPS) — это специализированный фреймворк для управления безопасностью, построенный на базе сервисов Apache Ranger и Apache Knox. ADPS может использоваться в ADH для обеспечения авторизации и управления доступом к сервисам на основе преднастроенных политик.

Перед установкой ADPS необходимо убедиться, что выполнены следующие требования:

- Кластер Arenadata Hyperwave версии 2.1.3 или выше установлен и настроен согласно руководству Установка Arenadata Hyperwave.
- Соблюдены все требования к установке ADPS:
  - Требования к оборудованию
  - Требования к сети
  - Программные требования
- Сервисы ADPS не должны использовать хосты кластера ADH. Рекомендуется установка ADPS на отдельной виртуальной машине или выделенном сервере.
- Рекомендуется хранить аудит-данные как в HDFS, так и в Solr. В конфигурации по умолчанию для Ranger Audits в Solr используется общий экземпляр Solr.
- При использовании внешней СУБД для хранения метаданных Ranger (PostgreSQL/MySQL) она должна быть запущена и доступна. При установке Ranger создаются два новых пользователя (имена по умолчанию: `rangeradmin` и `rangerlogger`) и две новые базы данных (имена по умолчанию: `ranger` и `ranger_audit`).
- Необходимо настроить один из следующих типов аутентификации:
  - MIT Kerberos
  - MS Active Directory
  - FreeIPA

ПРИМЕЧАНИЕ

Подробное описание установки ADPS приведено в статьях руководства Установка Arenadata Platform Security.

После установки ADPS и его интеграции с ADH (на вкладке ***Import*** в меню кластера ADH в интерфейсе ADCM) можно перейти к работе с плагинами Ranger, доступными для ряда сервисов ADH: HDFS, HBase, Hive, YARN и др.

Плагины Ranger для ADH

Сергей Тихомиров

Содержание

Назад к содержанию

Обзор

Доступ к веб-интерфейсу Ranger

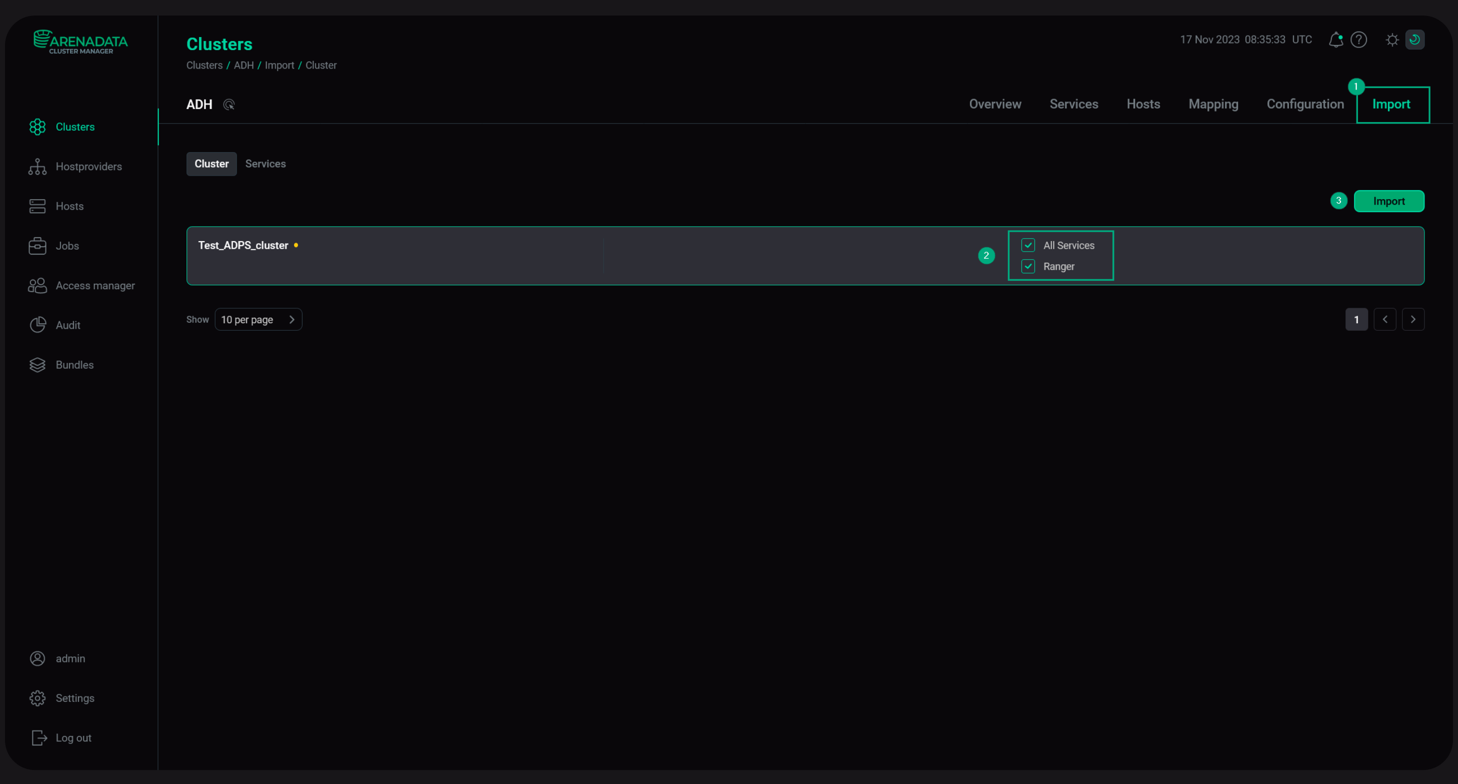
Обзор

Плагины Ranger доступны для следующих сервисов ADH:

- HBase
- HDFS
- Hive
- Impala
- Ozone
- Solr
- Spark3
- Trino
- YARN

Перед включением плагинов необходимо интегрировать кластер ADPS с кластером ADH.

1. На странице ***Clusters*** найдите ваш кластер ADH и кликните на его имя. В меню кластера перейдите на вкладку ***Import***.
2. На вкладке ***Import*** выберите опцию ***<ADPS_cluster_name>/Ranger***, чтобы связать ваш кластер ADPS с кластером ADH, и нажмите ***Import***.

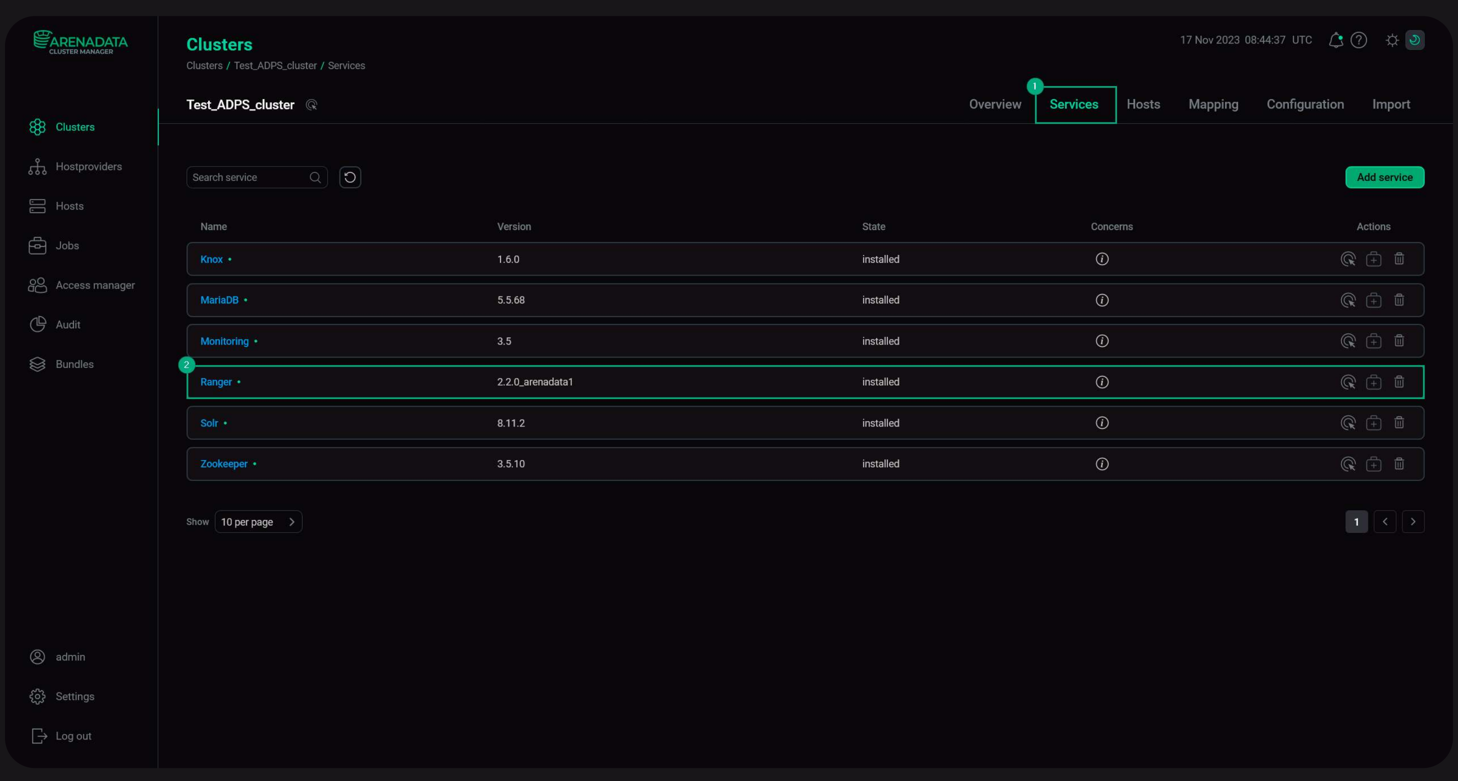

Clusters
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
Cluster
Services
Test_ADPS_cluster
All Services
Ranger

Импорт Ranger

Доступ к веб-интерфейсу Ranger

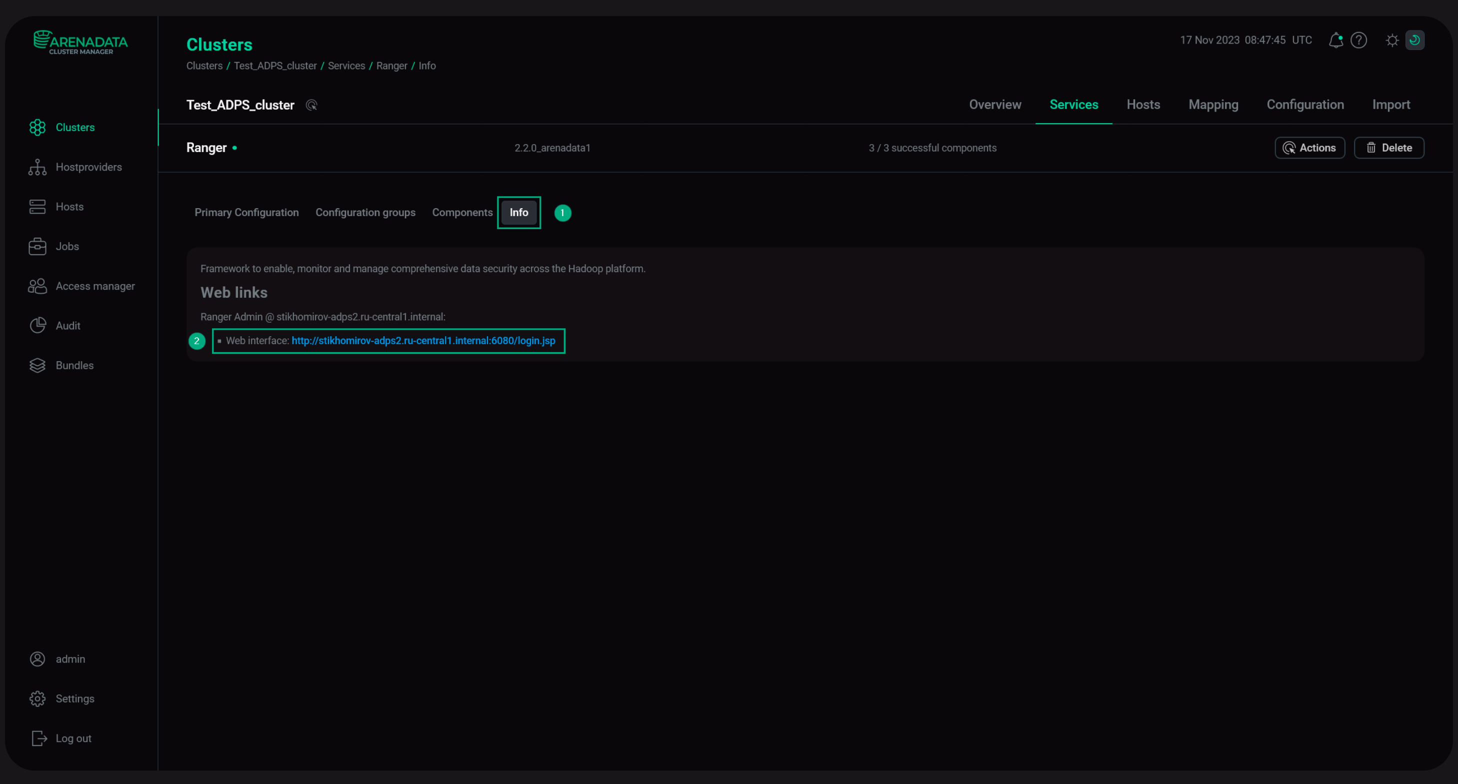
Ranger имеет веб-интерфейс, позволяющий управлять политиками и правами доступа. Следуйте шагам ниже, чтобы перейти на него.

1. На странице ***Clusters*** найдите кластер ADPS, связанный с вашим кластером ADH, кликните на него. Далее, перейдите на вкладку ***Services*** через кластерное меню и кликните на ***Ranger***.


Clusters
Test_ADPS_cluster
Overview
Services
Hosts
Mapping
Configuration
Import
Add service
Ranger

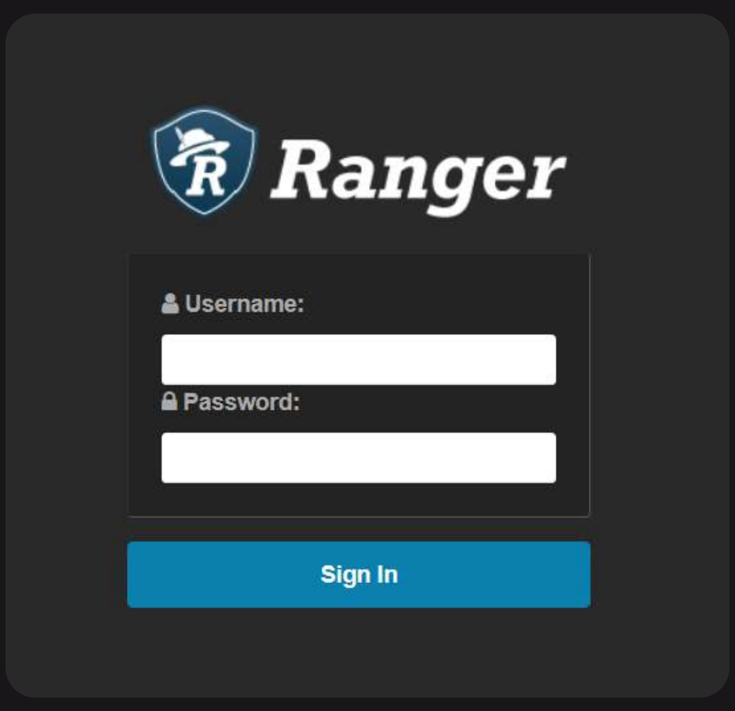
Сервис Ranger

2. На открывшейся странице перейдите по ссылке, помеченной как ***Web interface***.


Clusters
Test_ADPS_cluster
Ranger
Primary Configuration
Configuration groups
Components
Info
Web links

Ссылка на веб-интерфейс

3. Введите учетные данные администратора и кликните ***Sign in*** для авторизации в качестве администратора Ranger.


Ranger
Username:
Password:
Sign In

Вход в интерфейс Ranger

Пароли для доступа к веб-интерфейсу Ranger задаются перед установкой Ranger. Больше информации можно найти в секции ***Ranger → Credentials*** в статье Конфигурационные параметры.

Плагин Ranger HBase

Сергей Тихомиров

Содержание

Назад к содержанию

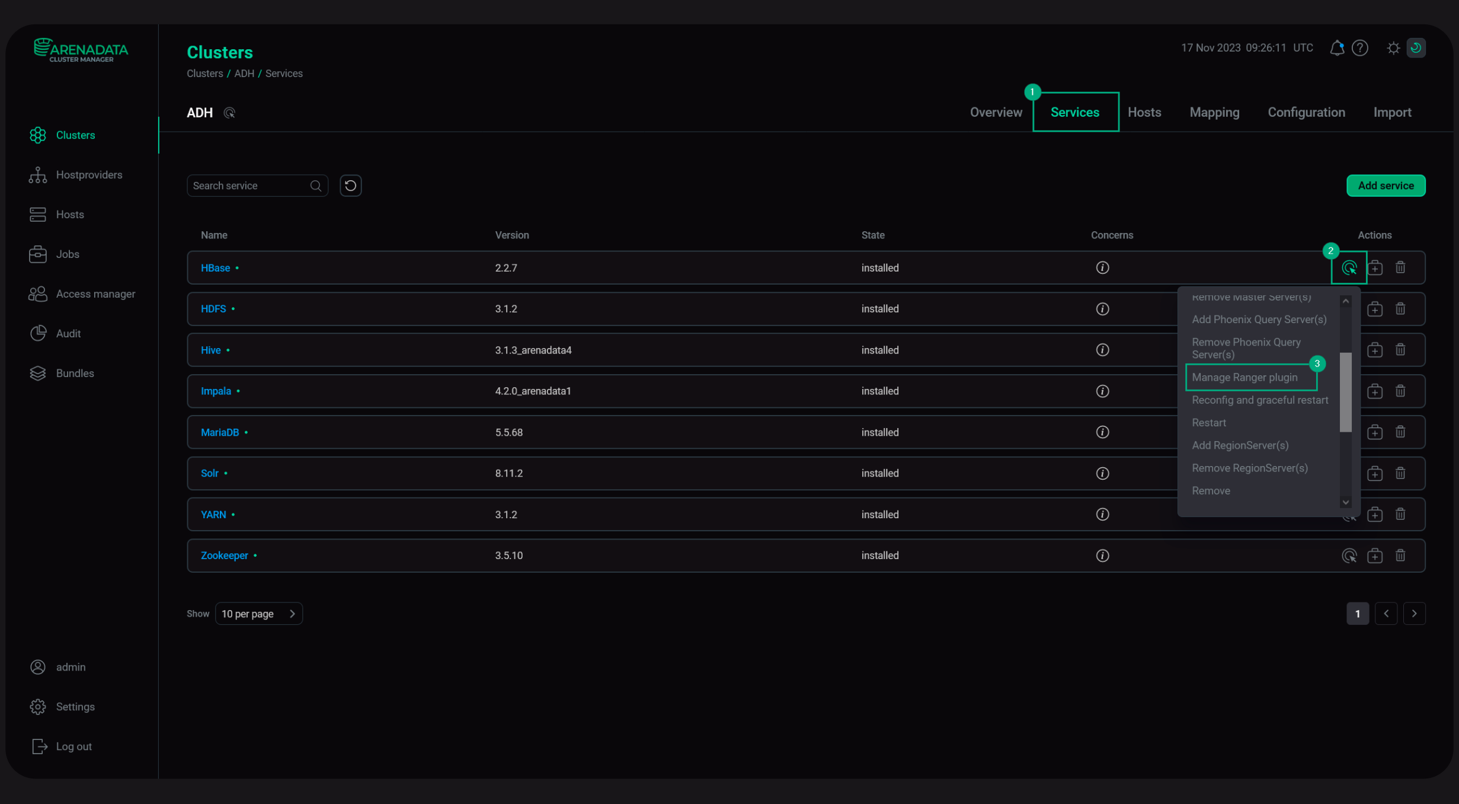
Включение плагина HBase

Добавление новой политики в Ranger

Включение плагина HBase

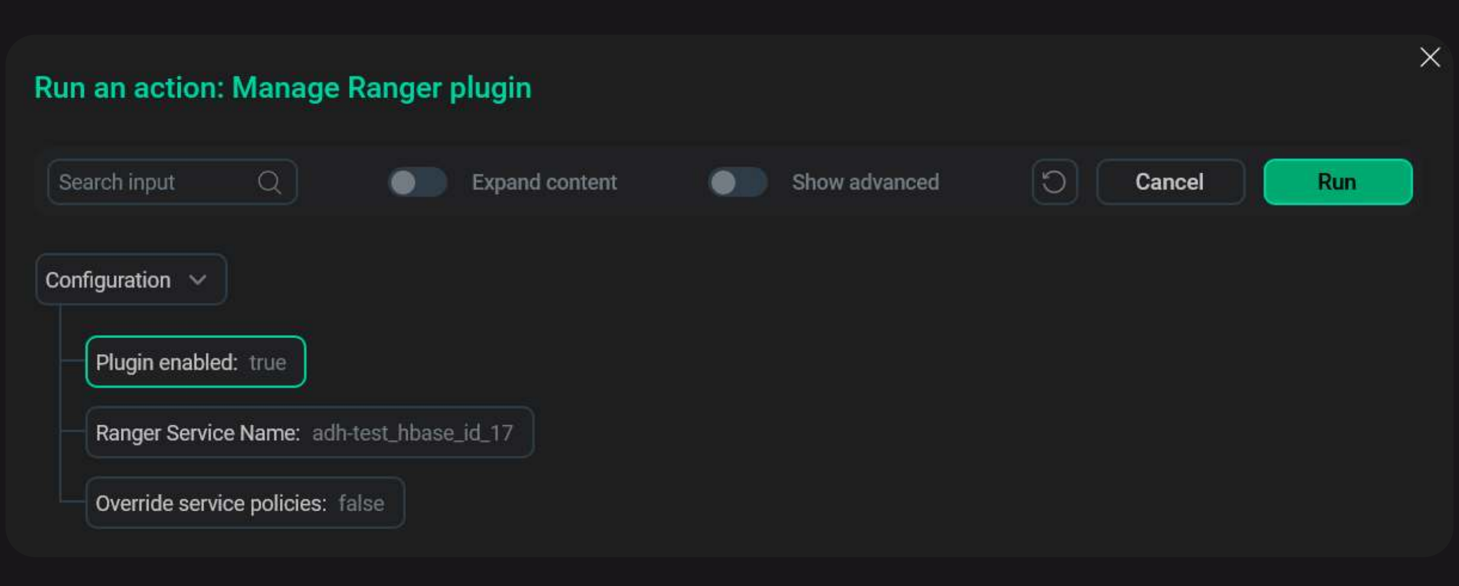
Чтобы включить плагин Ranger для HBase, выполните следующие действия.

1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку в строке HBase и выберите действие ***Manage Ranger plugin*** в выпадающем меню.



Сервисы кластера

3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.



Run an action: Manage Ranger plugin

Configuration

Plugin enabled: true

Ranger Service Name: adh-test_hbase_id_17

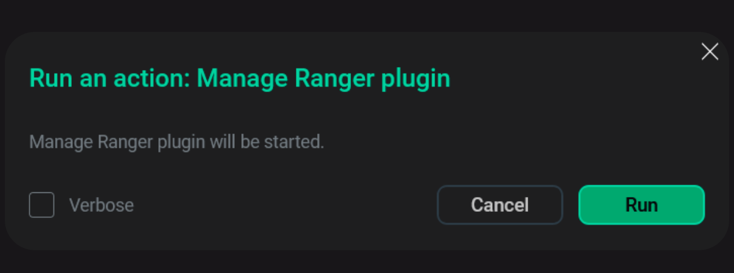
Override service policies: false

Состояние плагина

ПРИМЕЧАНИЕ

При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.



Run an action: Manage Ranger plugin

Manage Ranger plugin will be started.

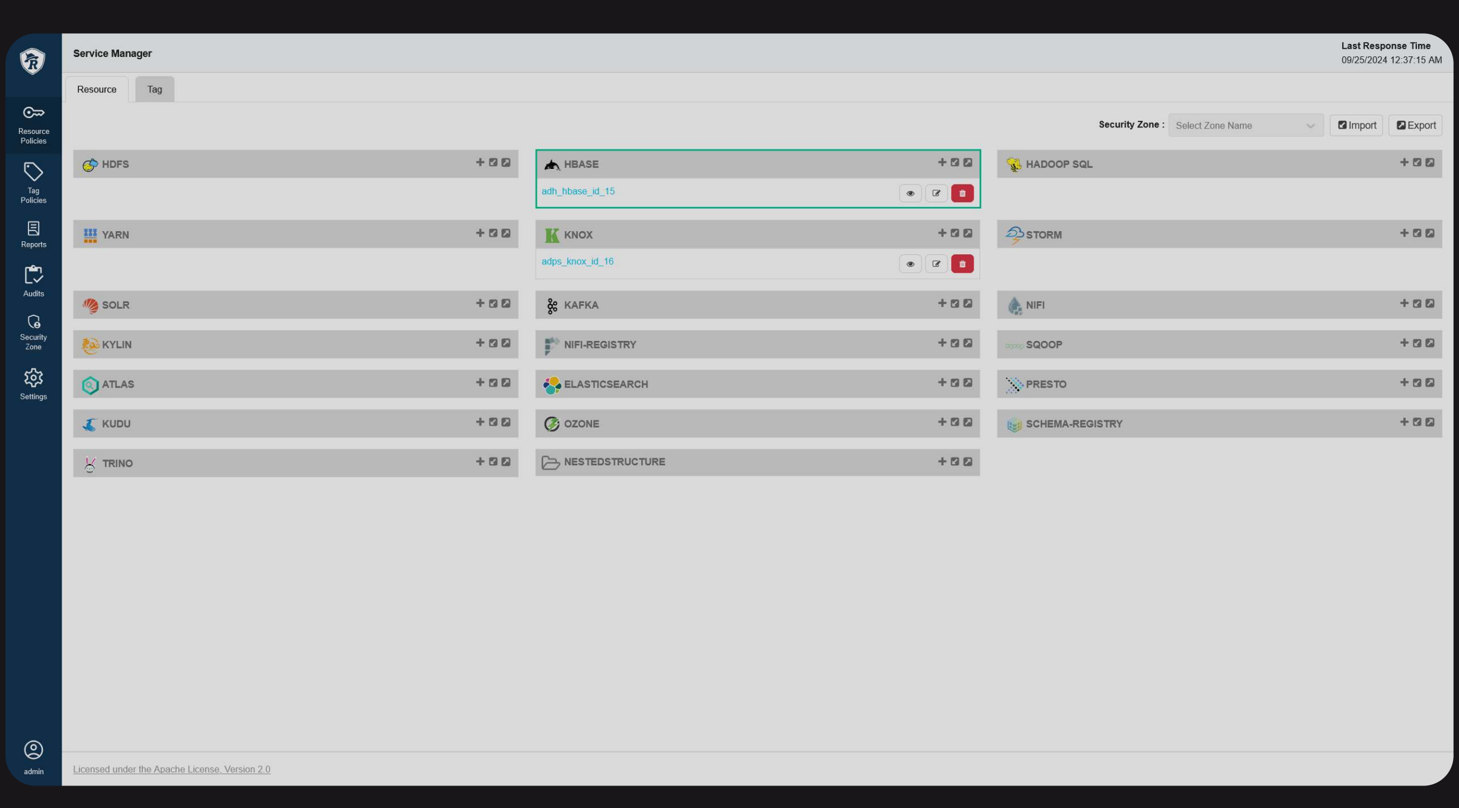
☐ Verbose

Подтверждение действия

Добавление новой политики в Ranger

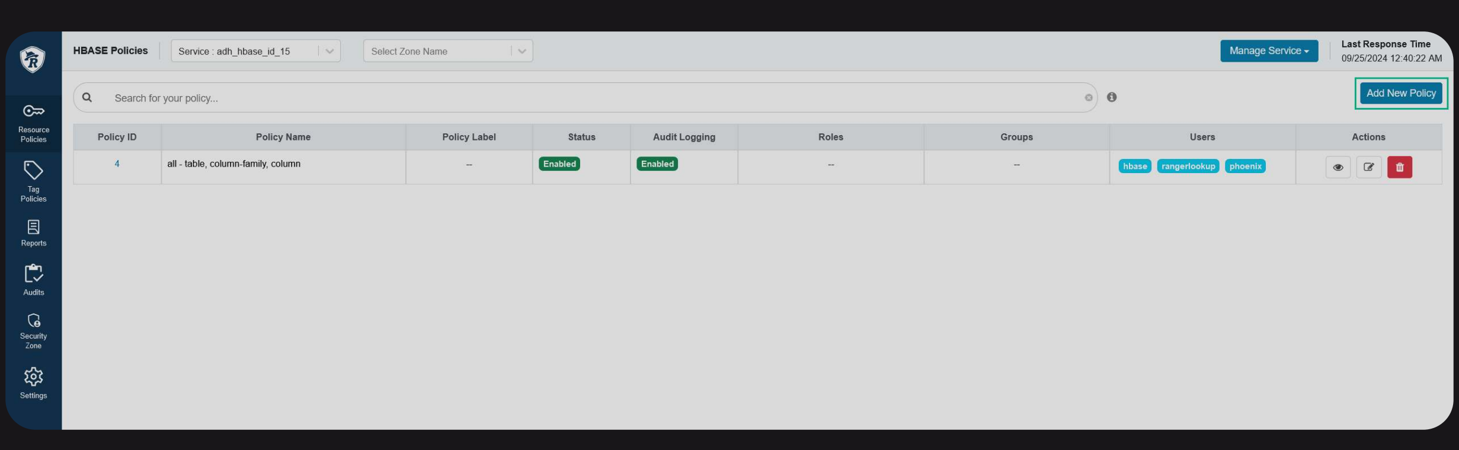
Чтобы добавить новую политику к существующему сервису HBase, необходимо выполнить следующие действия:

1. На странице ***Service Manager*** нажмите на существующий сервис в панели HBase.



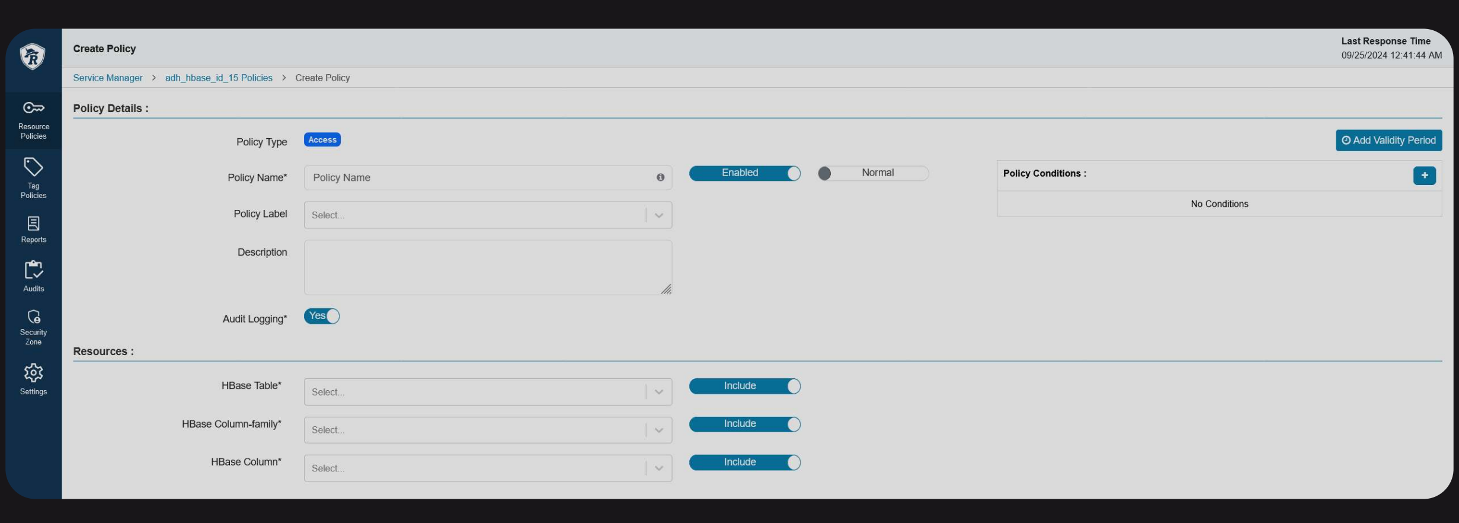
Страница Service Manager

2. На странице политик HBase нажмите ***Add New Policy***.



Добавление новой политики

3. На открывшейся странице ***Create Policy*** заполните необходимые поля.

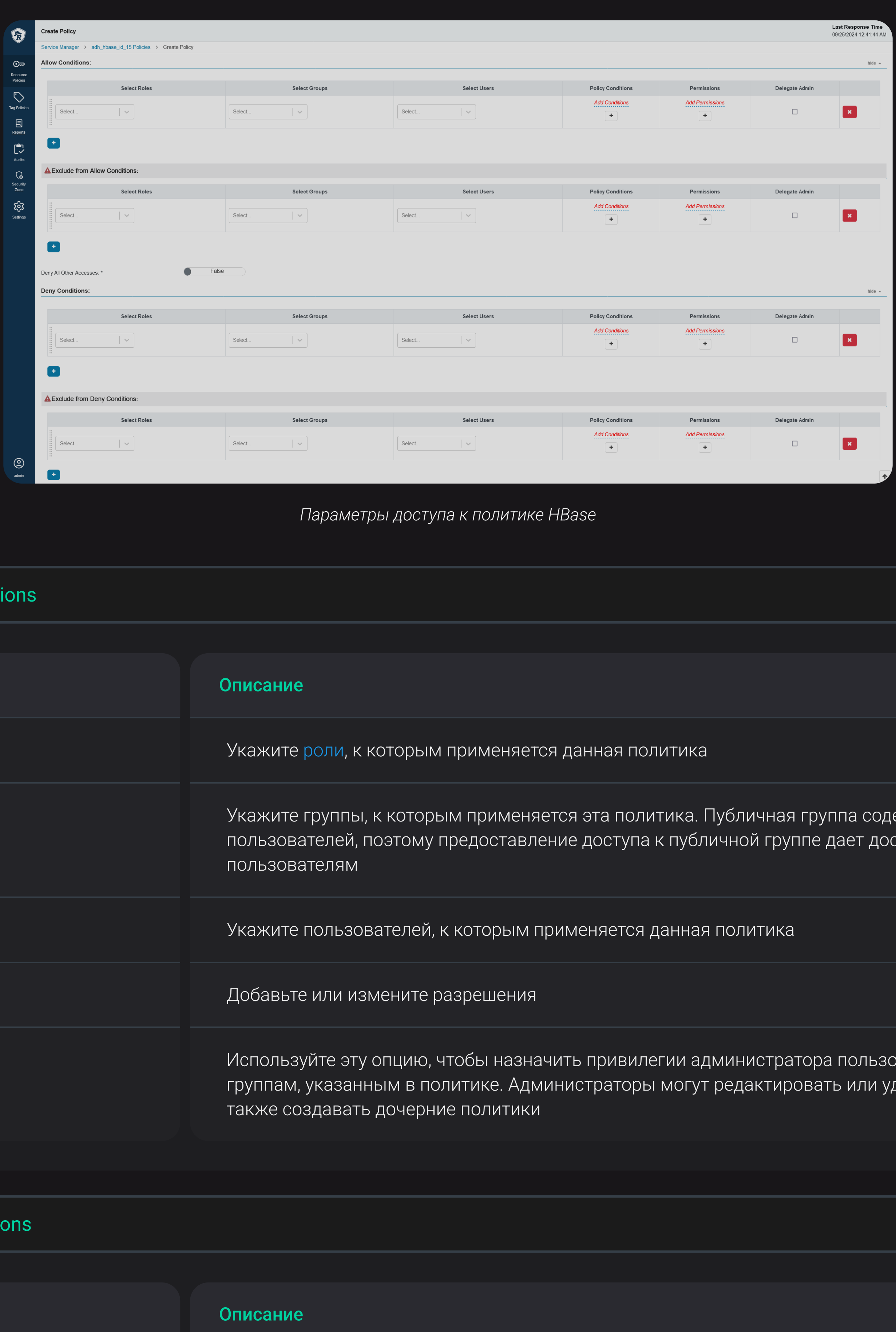


Параметры политики HBase

Параметры политики

Параметр	Описание
Policy Name	Название политики. Должно быть уникальным в системе
Enabled	Флаг, позволяющий применить политику сразу после создания
Normal/Override	Позволяет применить override-политику. Если выбран режим ***override***, то права доступа новой политики переписывают права доступа существующих политик
Policy Label	Позволяет группировать несколько политик с метками, что дает возможность искать политики по меткам. Поиском можно воспользоваться на страницах ***Policy listing*** и ***Reports***. Также упрощает экспорт/импорт политик — если требуется экспортировать некоторые политики, то можно провести поиск по меткам и экспортировать результат
HBase Table	Название HBase-таблицы
HBase Column-family	Название семейства столбцов HBase
HBase Column	Название столбца HBase
Description	Описание назначения политик
Audit Logging	Включает аудит для политики
Add Validity Period	Позволяет указать время начала и окончания действия политики

4. Установите параметры разрешений/запретов. Секция ***Allow Conditions*** позволяет разрешать доступ определенным ролям/группам/пользователям. Имеет смысл использовать эту секцию, когда требуется разрешить доступ небольшому числу пользователей, а остальным — запретить. Секция ***Deny Conditions*** позволяет запрещать доступ определенным ролям/группам/пользователями. Её удобно использовать, когда требуется запретить доступ небольшому числу пользователей, а остальным — разрешить. Также можно комбинировать правила из обеих секций. Для добавления дополнительных условий кликните +. Условия интерпретируются в порядке, указанном в политике. Сначала применяется условие, находящееся в начале списка, затем второе, затем третье и так далее.



Параметры доступа к политике HBase

Параметры Allow Conditions

Параметр	Описание
Select Role	Укажите роли, к которым применяется данная политика
Select Group	Укажите группы, к которым применяется эта политика. Публичная группа содержит всех пользователей, поэтому предоставление доступа к публичной группе дает доступ всем пользователям
Select User	Укажите пользователей, к которым применяется данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

Параметры Deny Conditions

Параметр	Описание
Select Role	Укажите роли, к которым не будет применяться данная политика
Select Group	Укажите группы, к которым не будет применяться эта политика. Публичная группа содержит всех пользователей, поэтому запрет доступа к публичной группе запрещает доступ всем пользователям
Select User	Укажите пользователей, к которым не будет применяться данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

5. Кликните ***Add*** внизу страницы.

Плагин Ranger HDFS

Сергей Тихомиров

Содержание

Назад к содержанию

Обзор

Включение плагина HDFS

Добавление новой политики в Ranger

Обзор

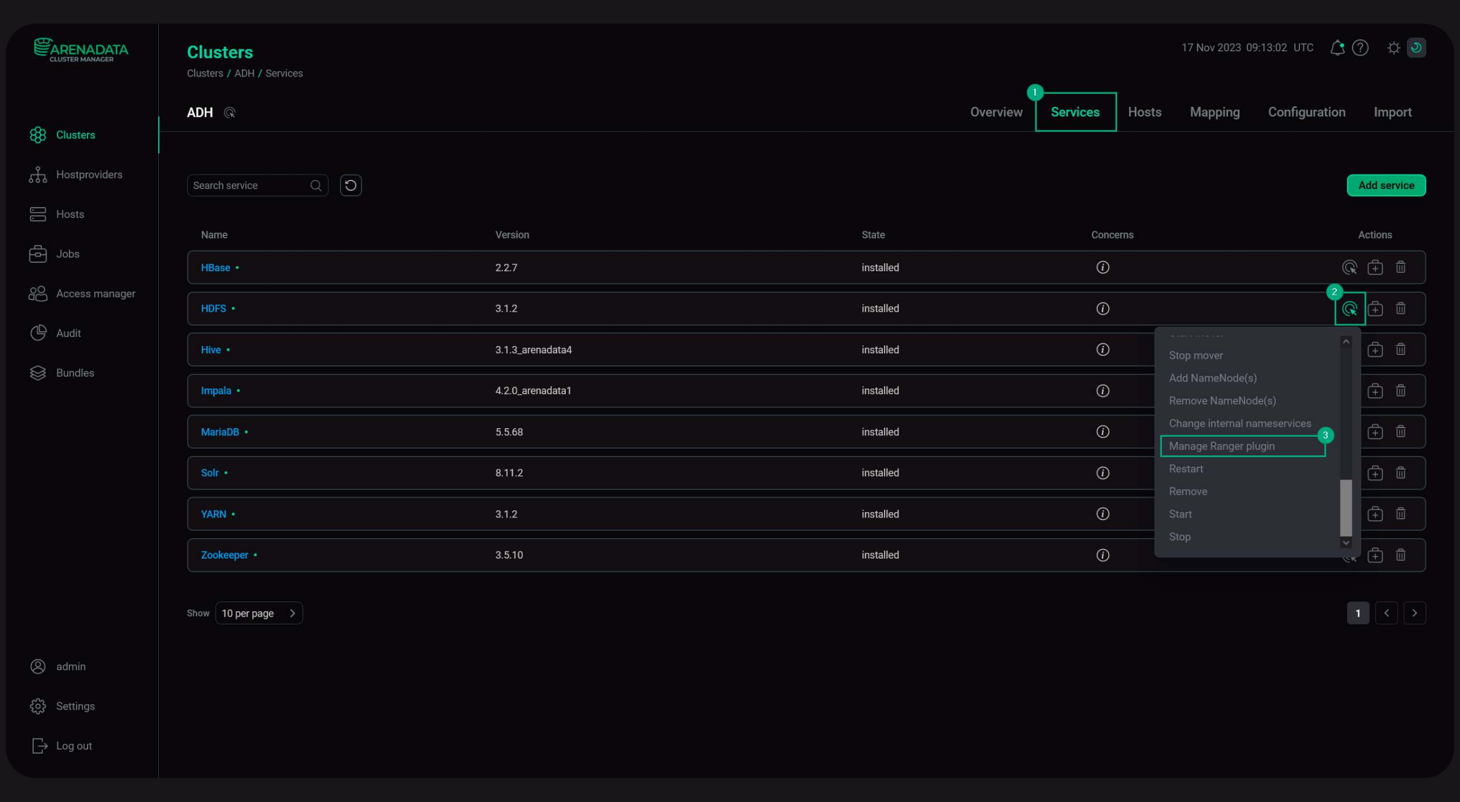
Конфигурация Apache Ranger позволяет применять как политики Ranger, так и разрешения HDFS. Когда узел NameNode получает запрос пользователя, плагин Ranger проверяет политики, установленные через Ranger Service Manager, и, если их нет, проверяет разрешения, установленные в HDFS.

> **ПРИМЕЧАНИЕ**
>
> Рекомендуется создавать разрешения в Ranger Service Manager и иметь ограниченные разрешения на уровне HDFS.

Включение плагина HDFS

Чтобы включить плагин Ranger для HDFS, выполните следующие действия.

1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку в строке HDFS и выберите действие ***Manage Ranger plugin*** в выпадающем меню.

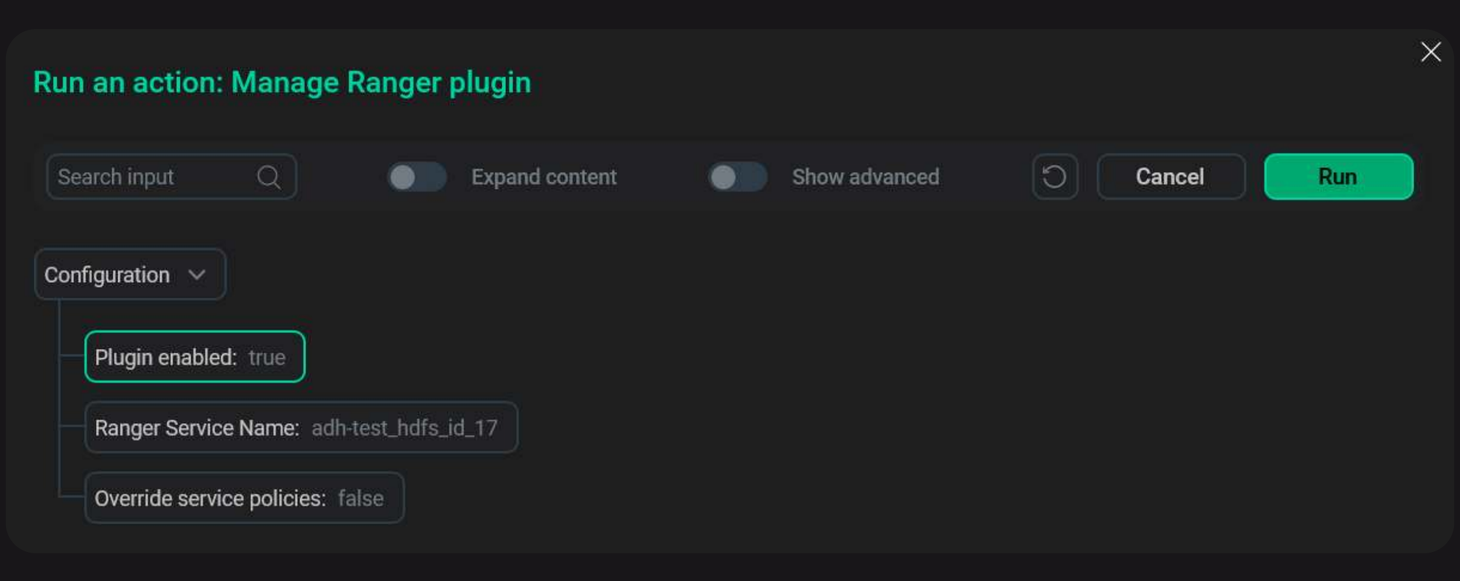


Name	Version	State
HBase	2.2.7	installed
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
Impala	4.2.0_arenadata1	installed
MariaDB	5.5.68	installed
Solr	8.11.2	installed
YARN	3.1.2	installed
Zookeeper	3.5.10	installed

ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH / Services
17 Nov 2023 09:13:02 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add service
Concerns
Actions
Stop mover
Add NameNode(s)
Remove NameNode(s)
Change internal nameservices
Manage Ranger plugin
Restart
Remove
Start
Stop
Show 10 per page
admin
Settings
Log out
1
2
3

Сервисы кластера

3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.



Run an action: Manage Ranger plugin

Configuration

Plugin enabled: true

Ranger Service Name: adh-test_hdfs_id_17

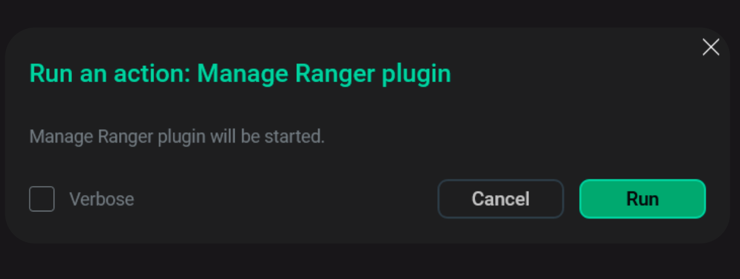
Override service policies: false

Search input
Expand content
Show advanced
Cancel
Run

Состояние плагина

> **ПРИМЕЧАНИЕ**
>
> При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.



Run an action: Manage Ranger plugin

Manage Ranger plugin will be started.

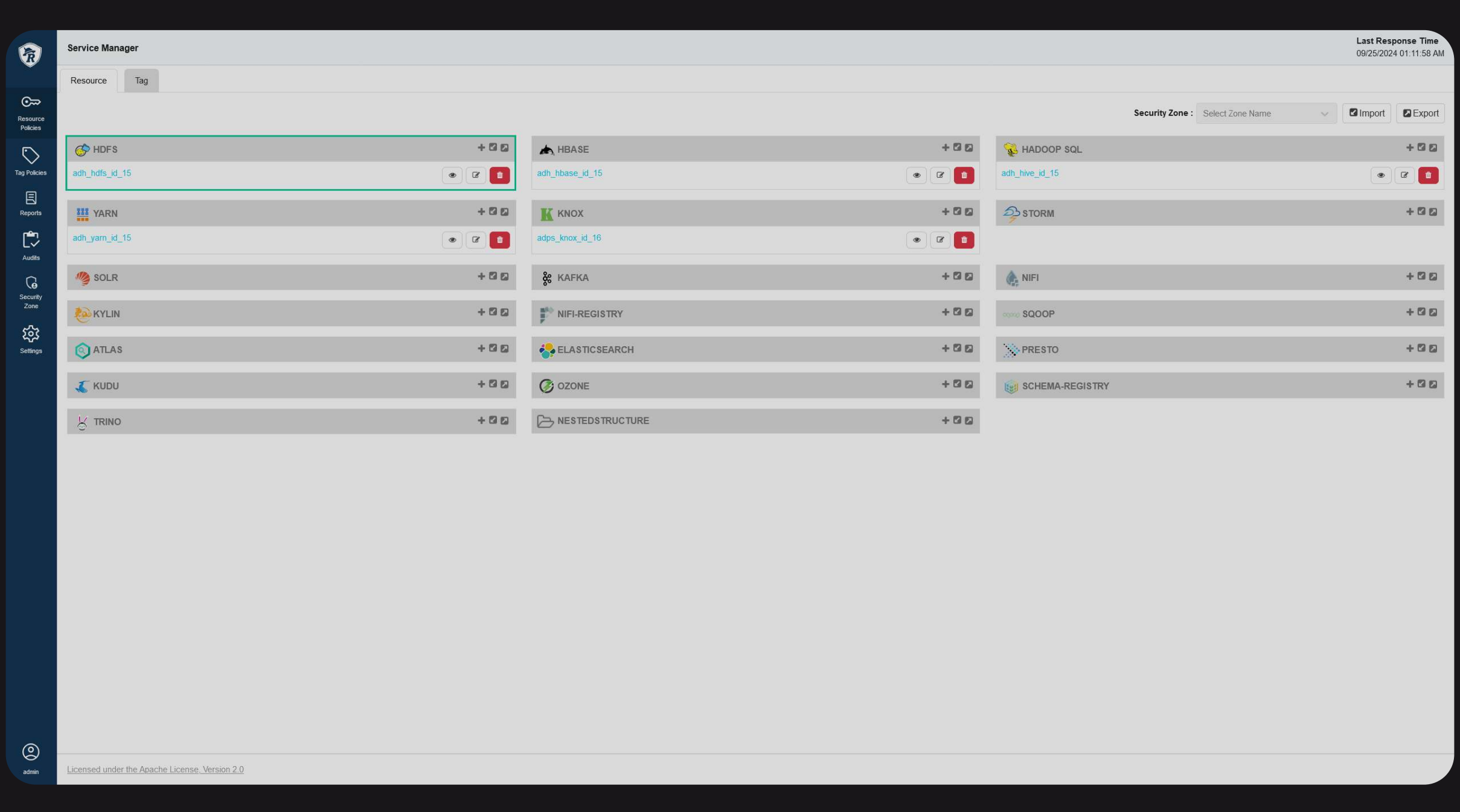
☐ Verbose
Cancel
Run

Подтверждение действия

Добавление новой политики в Ranger

Чтобы добавить новую политику к существующему сервису HDFS, необходимо выполнить следующие действия:

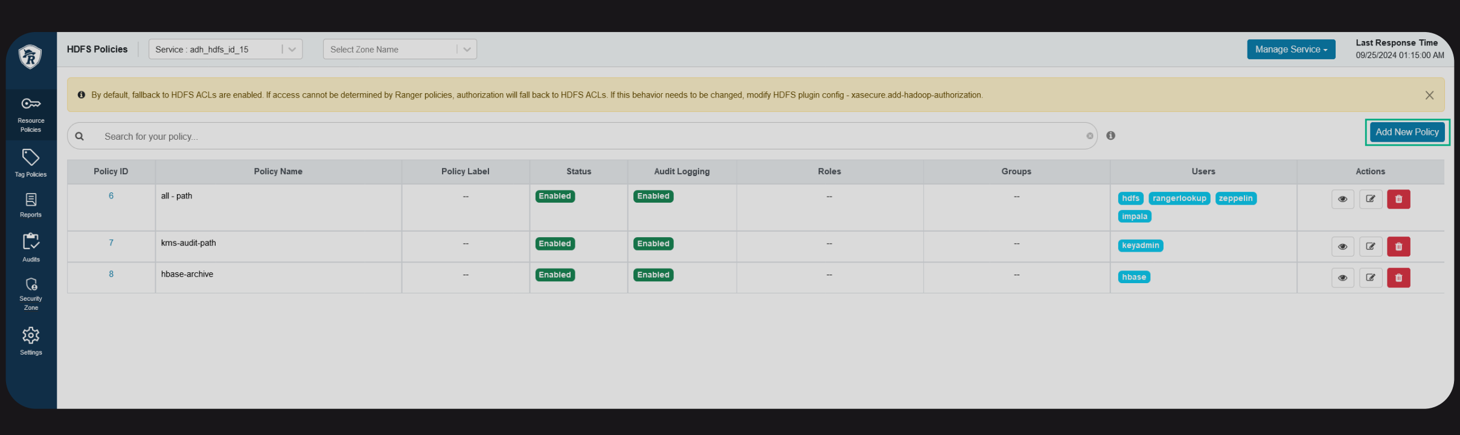
1. На странице ***Service Manager*** нажмите на существующий сервис в панели HDFS.



Service Manager
Last Response Time
09/25/2024 01:11:58 AM
Resource
Tag
Security Zone :
Select Zone Name
Import
Export
HDFS
adh_hdfs_id_15
HBASE
adh_hbase_id_15
HADOOP SQL
adh_hive_id_15
YARN
adh_yarn_id_15
KNOX
adps_knox_id_16
STORM
SOLR
KAFKA
NIFI
KYLIN
NIFI-REGISTRY
SQOOP
ATLAS
ELASTICSEARCH
PRESTO
KUDU
OZONE
SCHEMA-REGISTRY
TRINO
NESTEDSTRUCTURE
Licensed under the Apache License, Version 2.0

Страница Service Manager

2. На странице политик HDFS нажмите ***Add New Policy***.



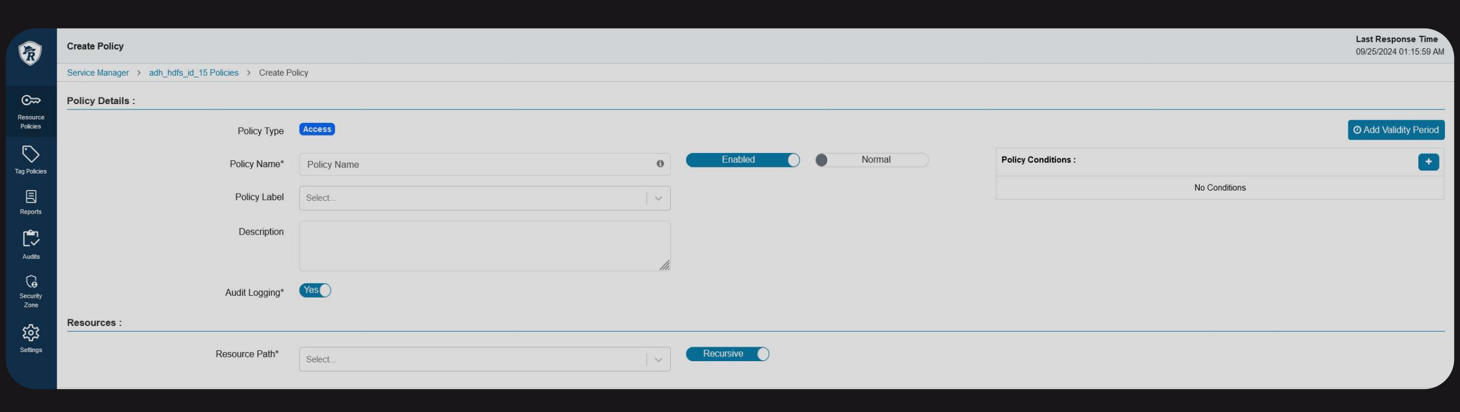
By default, fallback to HDFS ACLs are enabled. If access cannot be determined by Ranger policies, authorization will fall back to HDFS ACLs. If this behavior needs to be changed, modify HDFS plugin config - xasecure.add-hadoop-authorization.

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users
6	all - path	--	Enabled	Enabled	--	--	hdfs rangerlookup zeppelin impala
7	kms-audit-path	--	Enabled	Enabled	--	--	keyadmin
8	hbase-archive	--	Enabled	Enabled	--	--	hbase

HDFS Policies
Service : adh_hdfs_id_15
Select Zone Name
Manage Service
Last Response Time
09/25/2024 01:15:00 AM
Search for your policy...
Add New Policy
Actions

Добавление новой политики

3. На открывшейся странице ***Create Policy*** заполните необходимые поля.



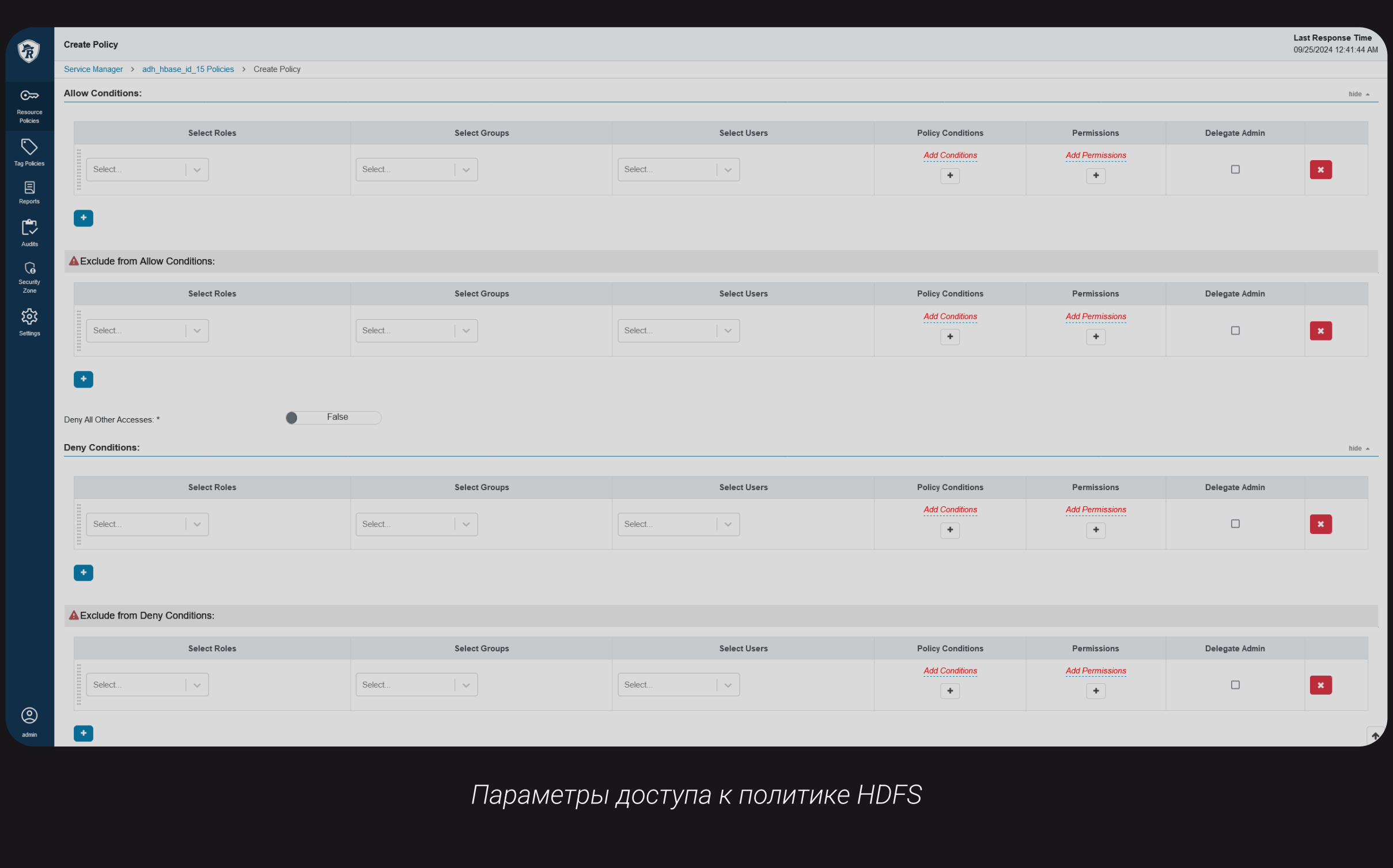
Create Policy
Last Response Time
09/25/2024 01:15:59 AM
Service Manager > adh_hdfs_id_15 Policies > Create Policy
Policy Details :
Policy Type
Access
Add Validity Period
Policy Name*
Policy Name
Enabled
Normal
Policy Conditions :
No Conditions
Policy Label
Select...
Description
Audit Logging*
Yes
Resources :
Resource Path*
Select...
Recursive

Параметры политики HDFS

Параметры политики

Параметр	Описание
Policy Name	Название политики. Должно быть уникальным в системе
Enabled	Флаг, позволяющий применить политику сразу после создания
Normal/Override	Позволяет применить override-политику. Если выбран режим ***override***, то права доступа новой политики переписывают права доступа существующих политик
Policy Label	Позволяет группировать несколько политик с метками, что дает возможность искать политики по меткам. Поиском можно воспользоваться на страницах ***Policy listing*** и ***Reports***. Также упрощает экспорт/импорт политик — если требуется экспортировать некоторые политики, то можно провести поиск по меткам и экспортировать результат
Resource Path	Путь к ресурсу для папки/файла политики. Чтобы избежать необходимости указывать полный путь или включать политику для всех вложенных папок или файлов, вы можете заполнить это поле с помощью подстановок (wildcards) (например, `/home`) или указать, что политика должна быть рекурсивной. В путь к ресурсу, имя базы данных, таблицу или столбец могут быть включены символы подстановки: `\*` — обозначает ноль или более символов; `?` — обозначает один символ
Description	Описание назначения политик
Audit Logging	Включает аудит для политики
Add Validity Period	Позволяет указать время начала и окончания действия политики

4. Установите параметры разрешений/запретов. Секция ***Allow Conditions*** позволяет разрешать доступ определенным ролям/группам/пользователям. Имеет смысл использовать эту секцию, когда требуется разрешить доступ небольшому числу пользователей, а остальным — запретить. Секция ***Deny Conditions*** позволяет запрещать доступ определенным ролям/группам/пользователями. Её удобно использовать, когда требуется запретить доступ небольшому числу пользователей, а остальным — разрешить. Также можно комбинировать правила из обеих секций. Для добавления дополнительных условий кликните **+**. Условия интерпретируются в порядке, указанном в политике. Сначала применяется условие, находящееся в начале списка, затем второе, затем третье и так далее.



Create Policy
Last Response Time
09/25/2024 12:41:44 AM
Service Manager > adh_hbase_id_15 Policies > Create Policy
Allow Conditions:
Exclude from Allow Conditions:
Deny All Other Accesses: *
False
Deny Conditions:
Exclude from Deny Conditions:
Select Roles
Select Groups
Select Users
Policy Conditions
Permissions
Delegate Admin
Select...
Add Conditions
Add Permissions

Параметры доступа к политике HDFS

Параметры Allow Conditions

Параметр	Описание
Select Role	Укажите роли, к которым применяется данная политика
Select Group	Укажите группы, к которым применяется эта политика. Публичная группа содержит всех пользователей, поэтому предоставление доступа к публичной группе дает доступ всем пользователям
Select User	Укажите пользователей, к которым применяется данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

Параметры Deny Conditions

Параметр	Описание
Select Role	Укажите роли, к которым не будет применяться данная политика
Select Group	Укажите группы, к которым не будет применяться эта политика. Публичная группа содержит всех пользователей, поэтому запрет доступа к публичной группе запрещает доступ всем пользователям
Select User	Укажите пользователей, к которым не будет применяться данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

5. Кликните ***Add*** внизу страницы.

Плагин Ranger Hive

Сергей Тихомиров

Содержание

Назад к содержанию

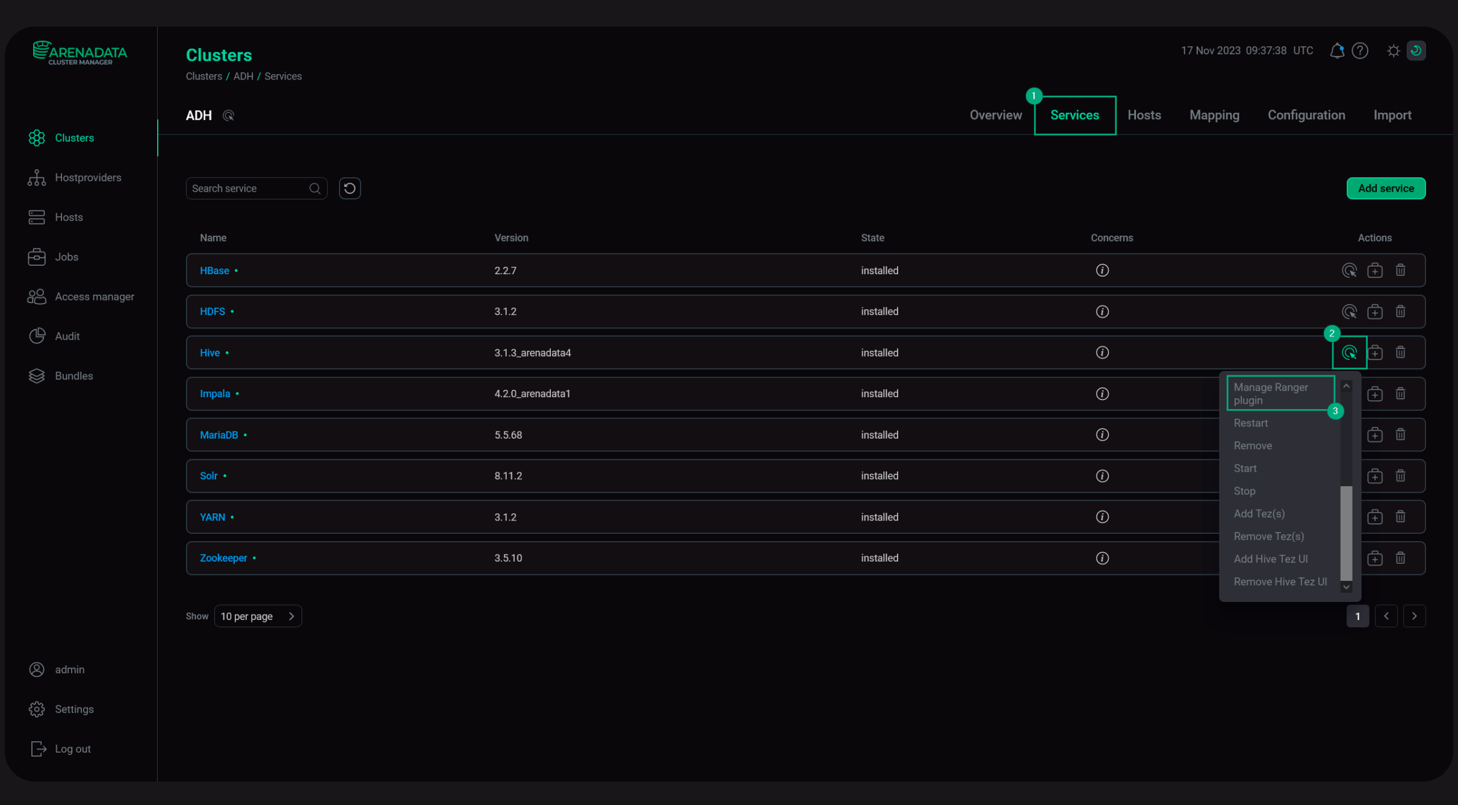
Включение плагина Hive

Добавление новой политики в Ranger

Включение плагина Hive

Чтобы включить плагин Ranger для Hive, выполните следующие действия.

1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку в строке Hive и выберите действие ***Manage Ranger plugin*** в выпадающем меню.

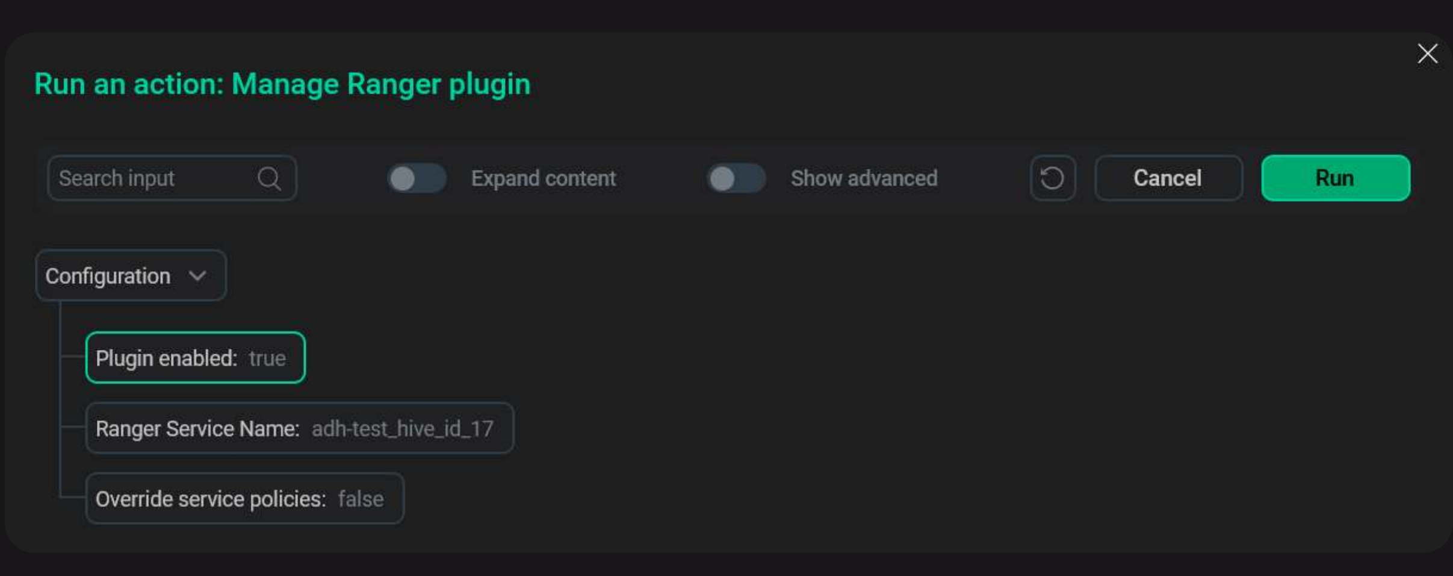


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH / Services
17 Nov 2023 09:37:38 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add service
Manage Ranger plugin
Restart
Remove
Start
Stop
Add Tez(s)
Remove Tez(s)
Add Hive Tez UI
Remove Hive Tez UI
Show 10 per page
1
2
3

Name	Version	State
HBase	2.2.7	installed
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
Impala	4.2.0_arenadata1	installed
MariaDB	5.5.68	installed
Solr	8.11.2	installed
YARN	3.1.2	installed
Zookeeper	3.5.10	installed

Сервисы кластера

3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.



Run an action: Manage Ranger plugin
Search input
Expand content
Show advanced
Cancel
Run
Configuration

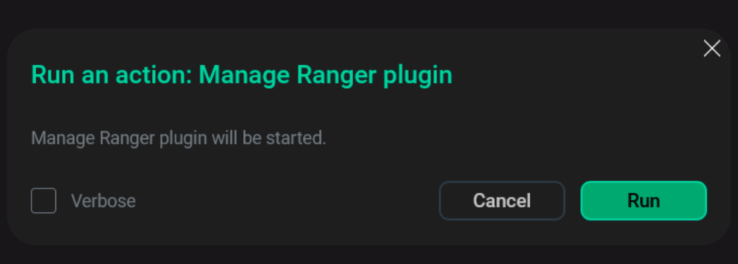
Plugin enabled: true
Ranger Service Name: adh-test_hive_id_17
Override service policies: false

Состояние плагина

ПРИМЕЧАНИЕ

При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.



Run an action: Manage Ranger plugin
Verbose
Cancel
Run

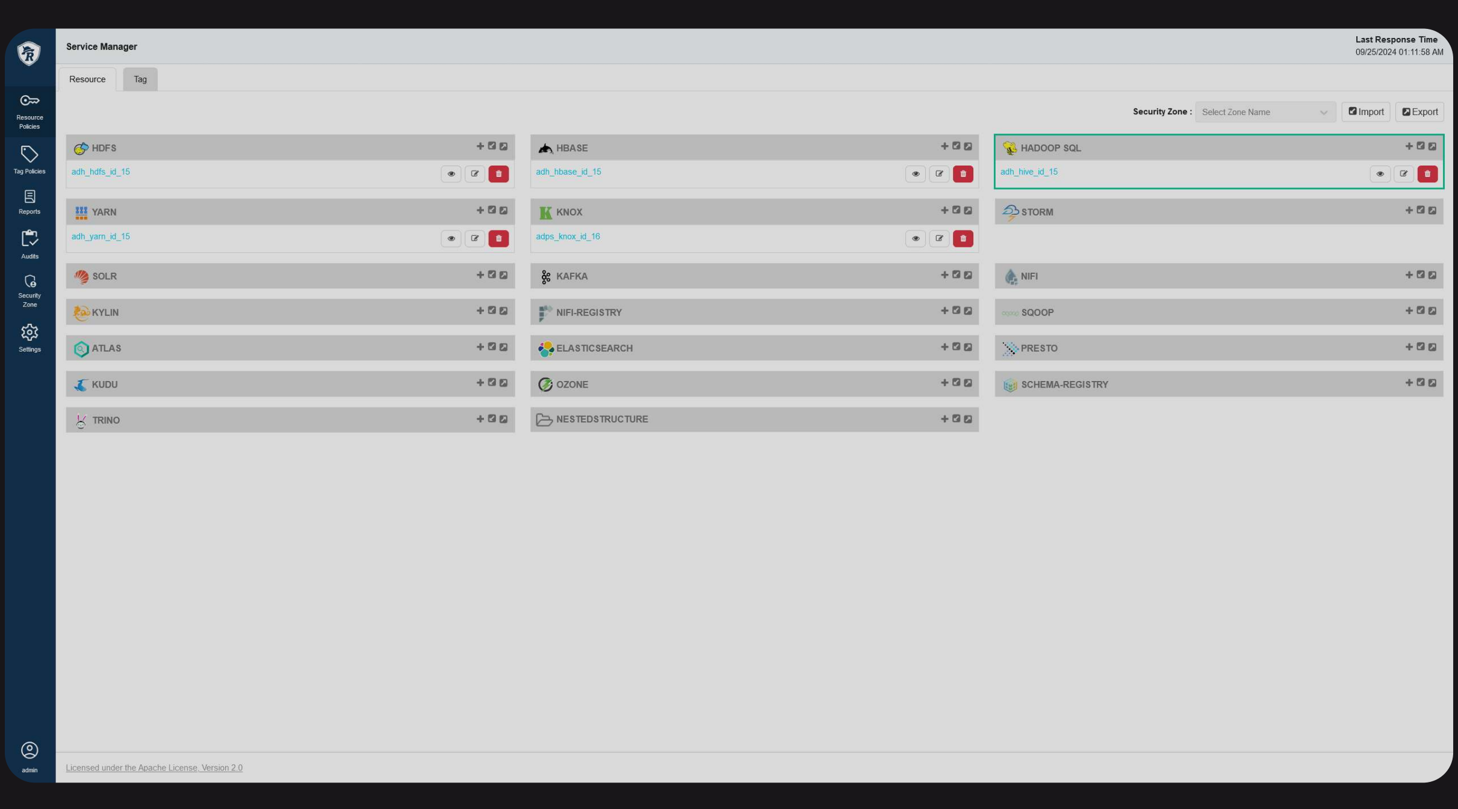
Manage Ranger plugin will be started.

Подтверждение действия

Добавление новой политики в Ranger

Чтобы добавить новую политику к существующему сервису Hive, необходимо выполнить следующие действия:

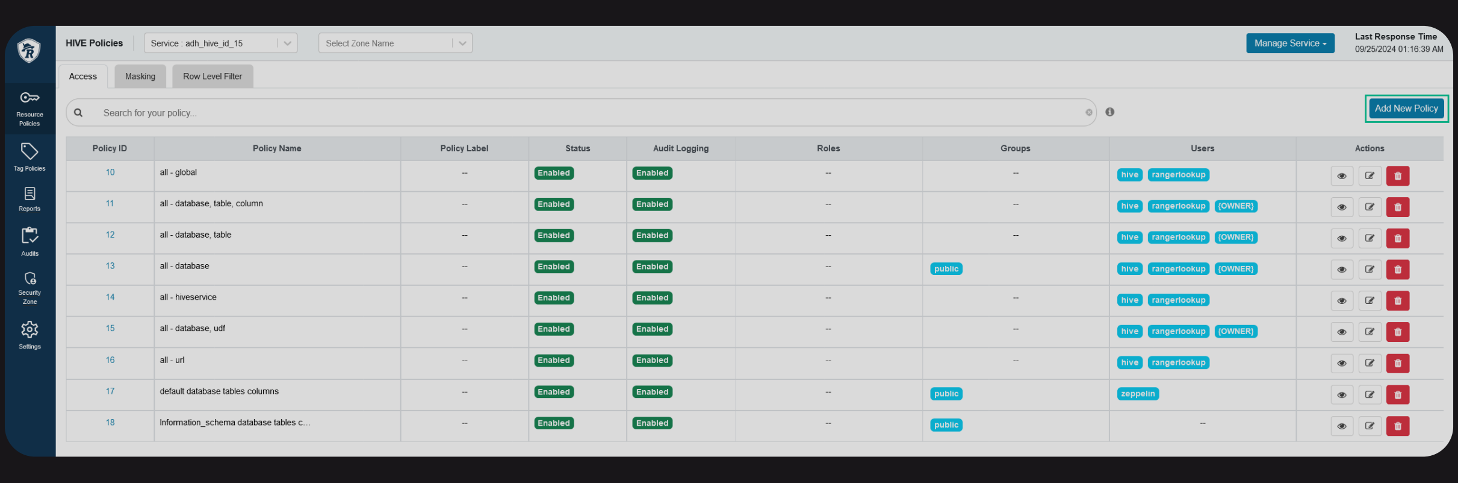
1. На странице ***Service Manager*** нажмите на существующий сервис Hive в панели Hadoop SQL.



Service Manager
Last Response Time
Resource
Tag
Security Zone :
Select Zone Name
Import
Export
HDFS
adh_hdfs_id_15
HBASE
adh_hbase_id_15
HADOOP SQL
adh_hive_id_15
YARN
adh_yarn_id_15
KNOX
adps_knox_id_16
STORM
SOLR
KAFKA
NIFI
KYLIN
NIFI-REGISTRY
SQOOP
ATLAS
ELASTICSEARCH
PRESTO
KUDU
OZONE
SCHEMA-REGISTRY
TRINO
NESTEDSTRUCTURE
Licensed under the Apache License, Version 2.0

Страница Service Manager

2. На странице политик Hive нажмите ***Add New Policy***.

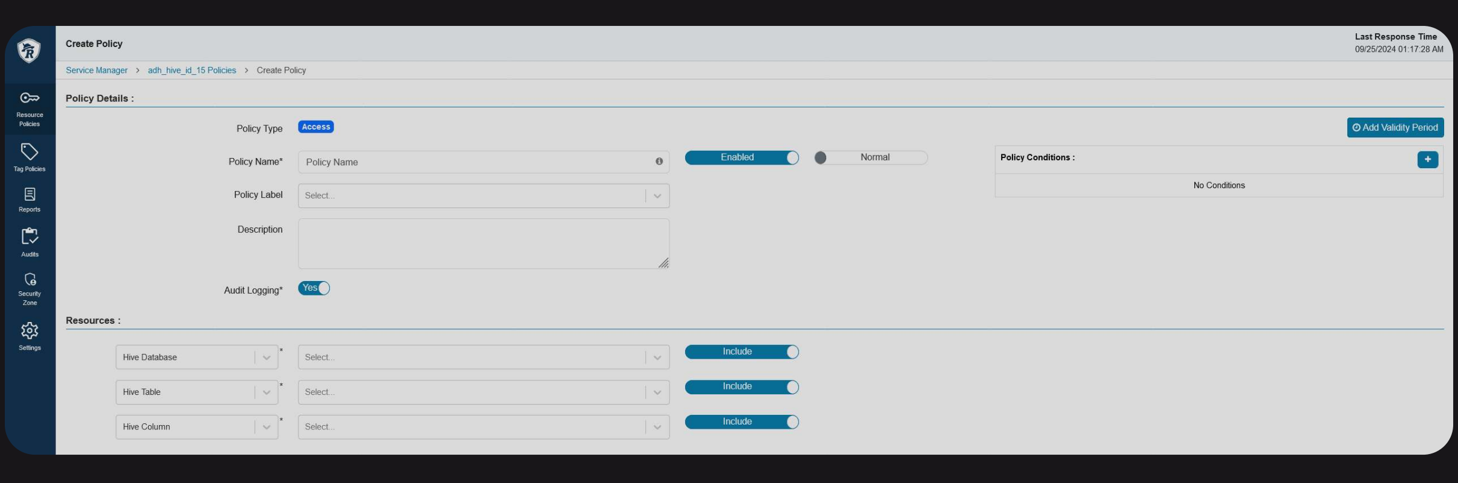


HIVE Policies
Service : adh_hive_id_15
Select Zone Name
Manage Service
Last Response Time
Access
Masking
Row Level Filter
Search for your policy...
Add New Policy

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users
10	all - global	--	Enabled	Enabled	--	--	hive rangerlookup
11	all - database, table, column	--	Enabled	Enabled	--	--	hive rangerlookup {OWNER}
12	all - database, table	--	Enabled	Enabled	--	--	hive rangerlookup {OWNER}
13	all - database	--	Enabled	Enabled	--	public	hive rangerlookup {OWNER}
14	all - hiveservice	--	Enabled	Enabled	--	--	hive rangerlookup
15	all - database, udf	--	Enabled	Enabled	--	--	hive rangerlookup {OWNER}
16	all - url	--	Enabled	Enabled	--	--	hive rangerlookup
17	default database tables columns	--	Enabled	Enabled	--	public	zeppelin
18	Information_schema database tables c...	--	Enabled	Enabled	--	public	--

Добавление новой политики

3. На открывшейся странице ***Create Policy*** заполните необходимые поля.



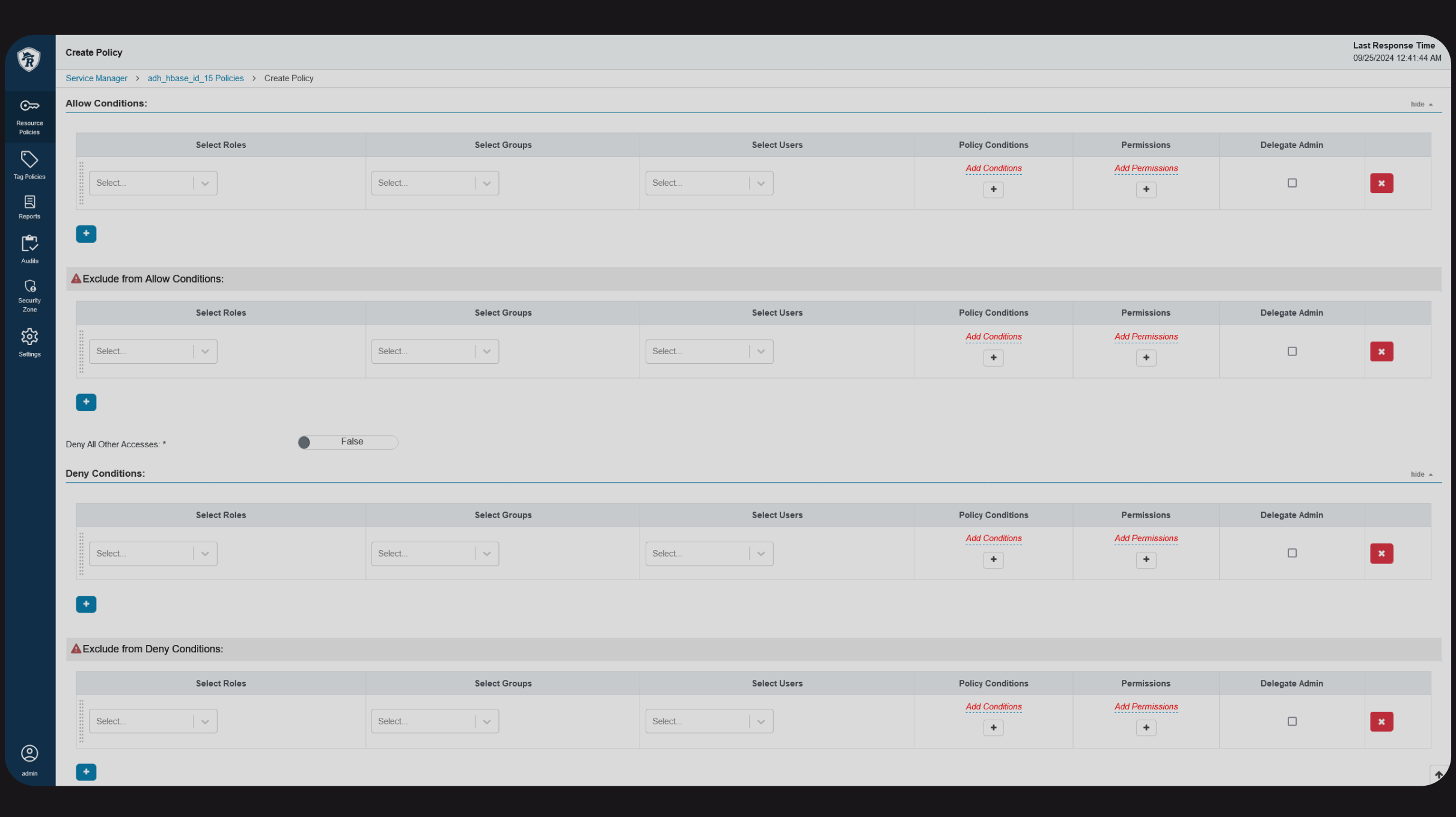
Create Policy
Service Manager > adh_hive_id_15 Policies > Create Policy
Policy Details :
Policy Type
Access
Policy Name*
Policy Label
Description
Audit Logging*
Yes
Enabled
Normal
Add Validity Period
Policy Conditions :
No Conditions
Resources :
Hive Database
Hive Table
Hive Column
Include

Параметры политики Hive

Параметры политики

Параметр	Описание
Policy Name	Название политики. Должно быть уникальным в системе
Enabled	Флаг, позволяющий применить политику сразу после создания
Normal/Override	Позволяет применить override-политику. Если выбран режим ***override***, то права доступа новой политики переписывают права доступа существующих политик
Policy Label	Позволяет группировать несколько политик с метками, что дает возможность искать политики по меткам. Поиском можно воспользоваться на страницах ***Policy listing*** и ***Reports***. Также упрощает экспорт/импорт политик — если требуется экспортировать некоторые политики, то можно провести поиск по меткам и экспортировать результат
database/url/hiveservice/global	Если выбран параметр ***database***, то введите имя соответствующей базы данных. Функция автозаполнения отображает доступные базы данных на основе введенного текста. Если выбран параметр ***url***, то введите путь к облачному хранилищу (например, *s3a://dev-admin/demo/campaigns.txt*), где необходимо разрешение конечного пользователя для чтения/записи данных Hive. Опция ***hiveservice*** позволяет пользователю, имеющему права Service Admin в Ranger, запустить API остановки запроса: `kill query <queryID>`. Опция ***global*** используется для создания глобальных политик, которые используют шаблоны соответствия и определяют необходимый доступ по умолчанию
table/udf/none	Определяет политику на основе таблицы или UDF
column/none	Название колонки Hive. Неверная установка этого параметра приведет к сообщению об ошибке `HiveAccessControlException`
Description	Описание назначения политик
Audit Logging	Включает аудит для политики
Add Validity Period	Позволяет указать время начала и окончания действия политики

4. Установите параметры разрешений/запретов. Секция ***Allow Conditions*** позволяет разрешать доступ определенным ролям/группам/пользователям. Имеет смысл использовать эту секцию, когда требуется разрешить доступ небольшому числу пользователей, а остальным — запретить. Секция ***Deny Conditions*** позволяет запрещать доступ определенным ролям/группам/пользователями. Её удобно использовать, когда требуется запретить доступ небольшому числу пользователей, а остальным — разрешить. Также можно комбинировать правила из обеих секций. Для добавления дополнительных условий кликните **+**. Условия интерпретируются в порядке, указанном в политике. Сначала применяется условие, находящееся в начале списка, затем второе, затем третье и так далее.



Create Policy
Last Response Time
Service Manager > adh_hbase_id_15 Policies > Create Policy
Allow Conditions:
Exclude from Allow Conditions:
Deny All Other Accesses: *
False
Deny Conditions:
Exclude from Deny Conditions:
Select Roles
Select Groups
Select Users
Policy Conditions
Permissions
Delegate Admin
Add Conditions
Add Permissions

Параметры доступа к политике Hive

Параметры Allow Conditions

Параметр	Описание
Select Role	Укажите роли, к которым применяется данная политика
Select Group	Укажите группы, к которым применяется эта политика. Публичная группа содержит всех пользователей, поэтому предоставление доступа к публичной группе дает доступ всем пользователям
Select User	Укажите пользователей, к которым применяется данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

Параметры Deny Conditions

Параметр	Описание
Select Role	Укажите роли, к которым не будет применяться данная политика
Select Group	Укажите группы, к которым не будет применяться эта политика. Публичная группа содержит всех пользователей, поэтому запрет доступа к публичной группе запрещает доступ всем пользователям
Select User	Укажите пользователей, к которым не будет применяться данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

5. Кликните ***Add*** внизу страницы.

Плагин Ranger Impala

Сергей Тихомиров

Содержание

Назад к содержанию

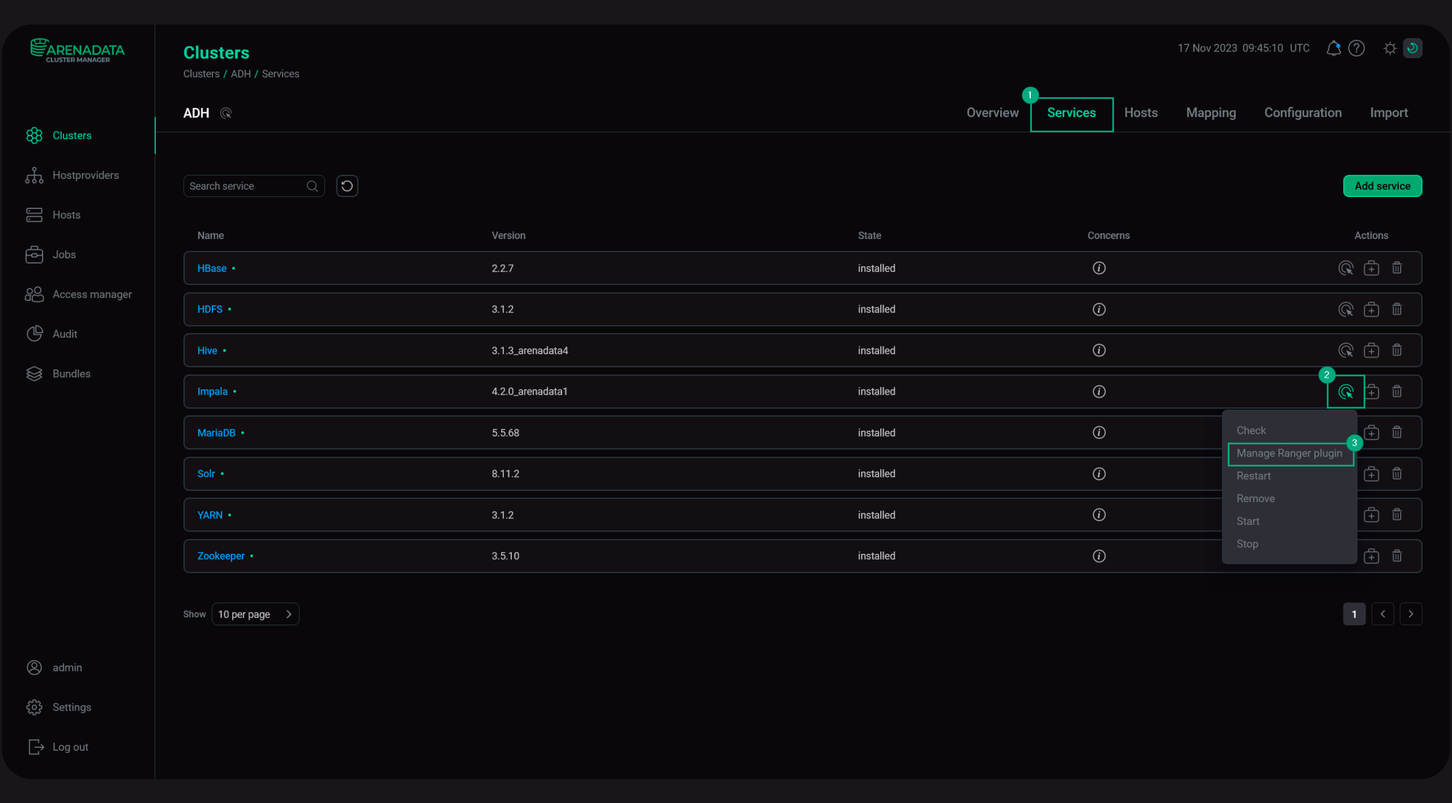
Включение плагина Impala

Добавление новой политики в Ranger

Включение плагина Impala

Чтобы включить плагин Ranger для Impala, выполните следующие действия.

1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку ◎ в строке Impala и выберите действие ***Manage Ranger plugin*** в выпадающем меню.

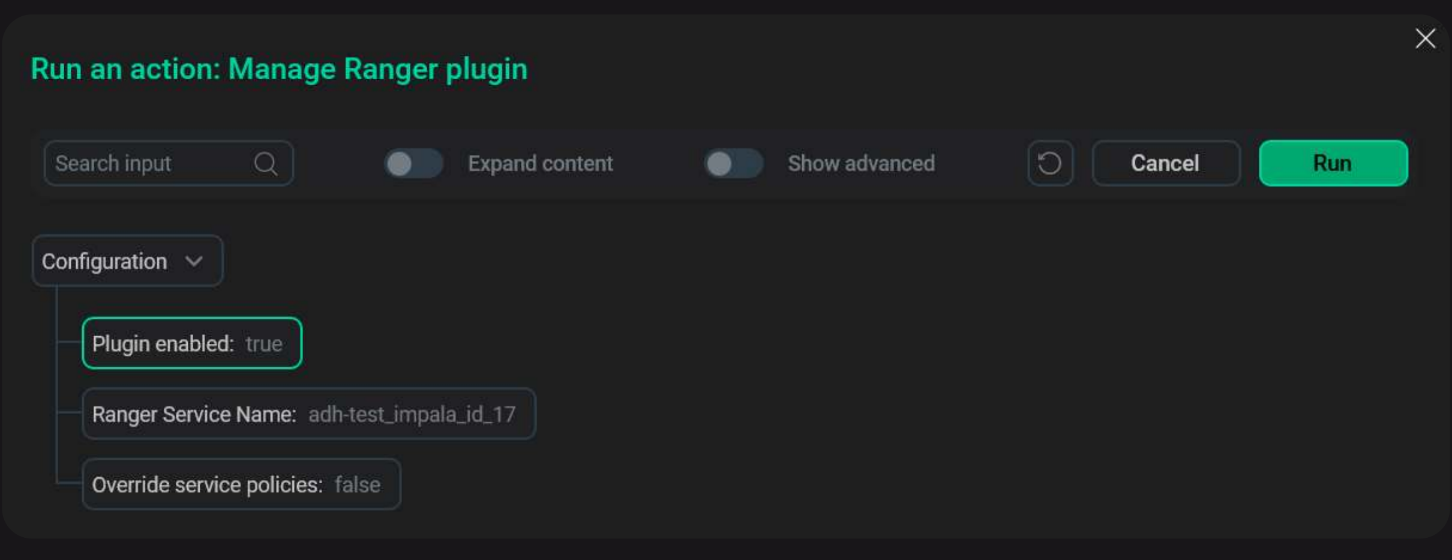


ARENADATA
Clusters
Clusters / ADH / Services
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add service
Check
Manage Ranger plugin
Restart
Remove
Start
Stop

Name	Version	State
HBase	2.2.7	installed
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
Impala	4.2.0_arenadata1	installed
MariaDB	5.5.68	installed
Solr	8.11.2	installed
YARN	3.1.2	installed
Zookeeper	3.5.10	installed

Сервисы кластера

3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.



Run an action: Manage Ranger plugin
Search input
Expand content
Show advanced
Cancel
Run
Configuration

Plugin enabled: true

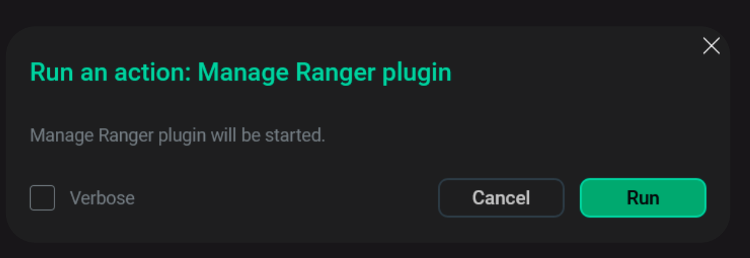
Ranger Service Name: adh-test_impala_id_17

Override service policies: false

Состояние плагина

> **ПРИМЕЧАНИЕ**
>
> При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.



Run an action: Manage Ranger plugin
Verbose
Cancel
Run

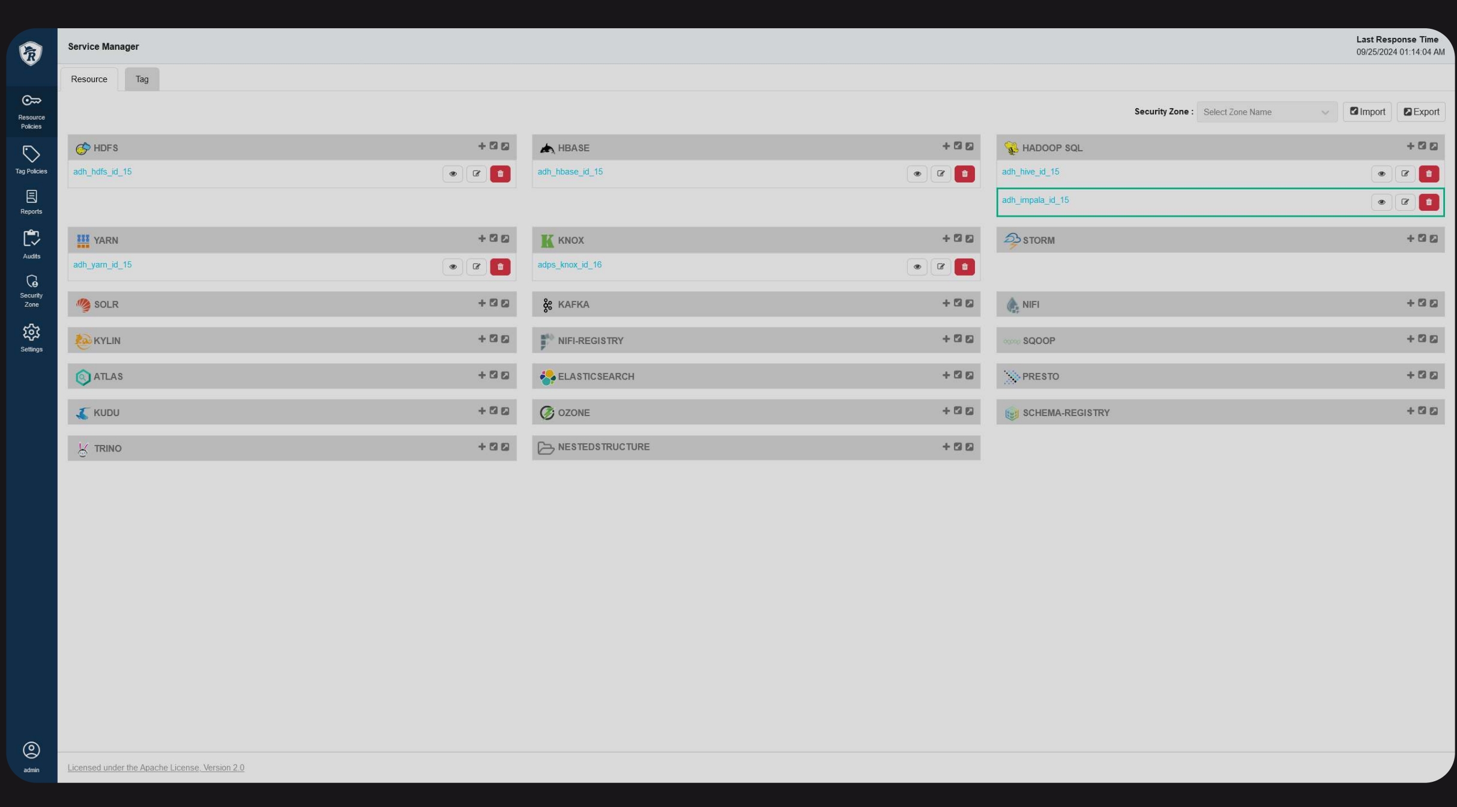
Manage Ranger plugin will be started.

Подтверждение действия

Добавление новой политики в Ranger

Чтобы добавить новую политику к существующему сервису Impala, необходимо выполнить следующие действия:

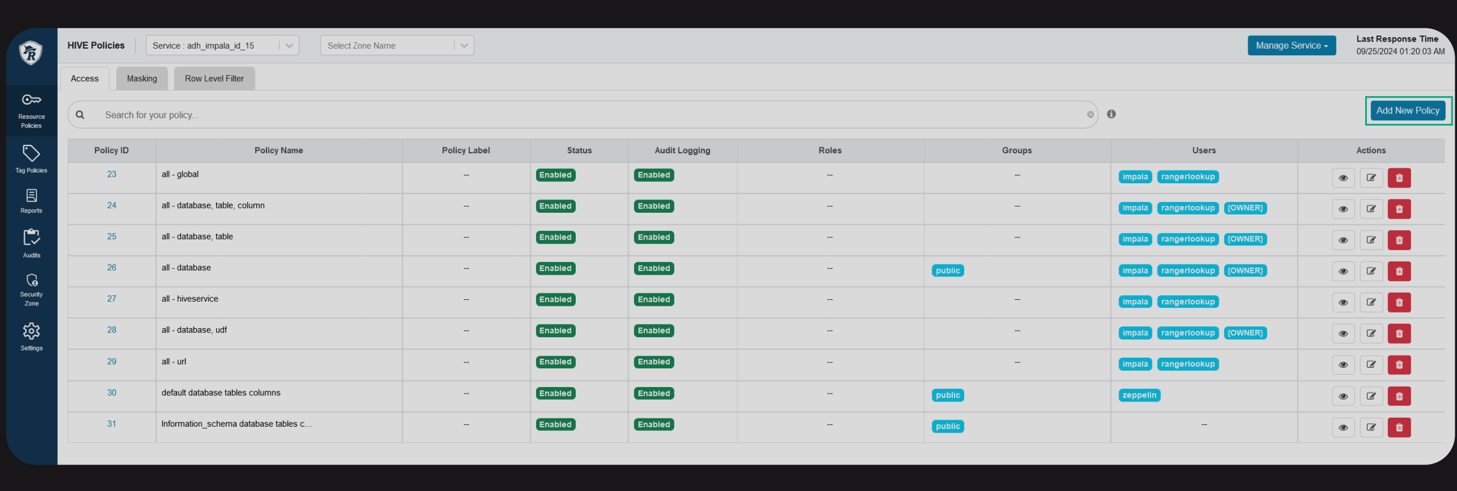
1. На странице ***Service Manager*** нажмите на существующий сервис Impala в панели Hadoop SQL.



Service Manager
Resource
Tag
Security Zone :
Import
Export
HDFS
HBASE
HADOOP SQL
adh_hdfs_id_15
adh_hbase_id_15
adh_hive_id_15
adh_impala_id_15
YARN
KNOX
STORM
adh_yarn_id_15
adps_knox_id_16
SOLR
KAFKA
NIFI
KYLIN
NIFI-REGISTRY
SQOOP
ATLAS
ELASTICSEARCH
PRESTO
KUDU
OZONE
SCHEMA-REGISTRY
TRINO
NESTEDSTRUCTURE

Страница Service Manager

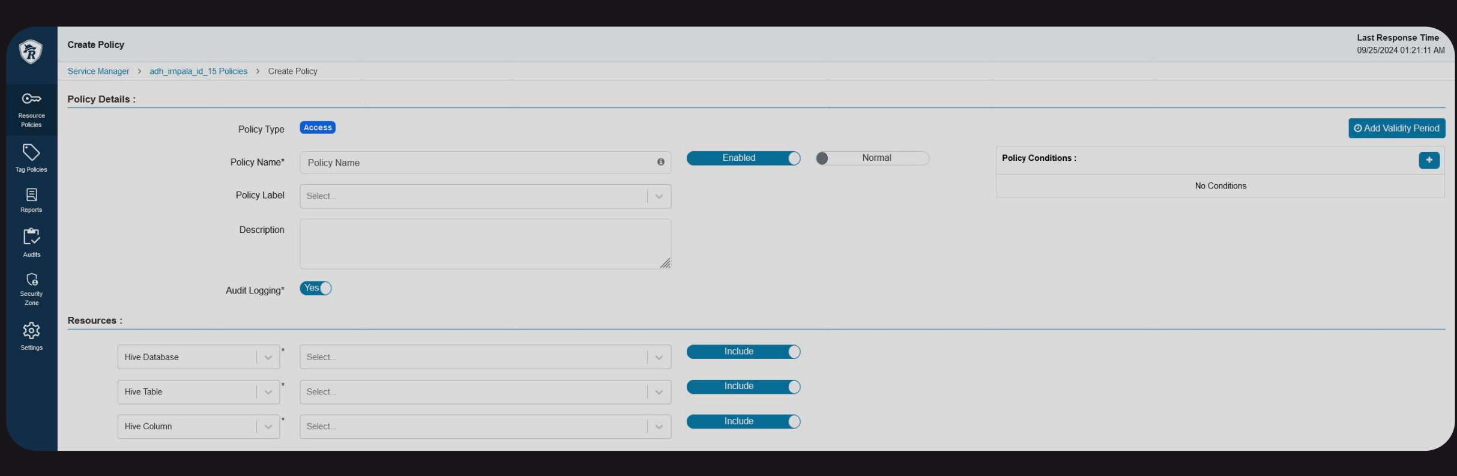
2. На странице политик Impala нажмите ***Add New Policy***.



HIVE Policies
Service : adh_impala_id_15
Manage Service
Access
Masking
Row Level Filter
Search for your policy...
Add New Policy

Добавление новой политики

3. На открывшейся странице ***Create Policy*** заполните необходимые поля.



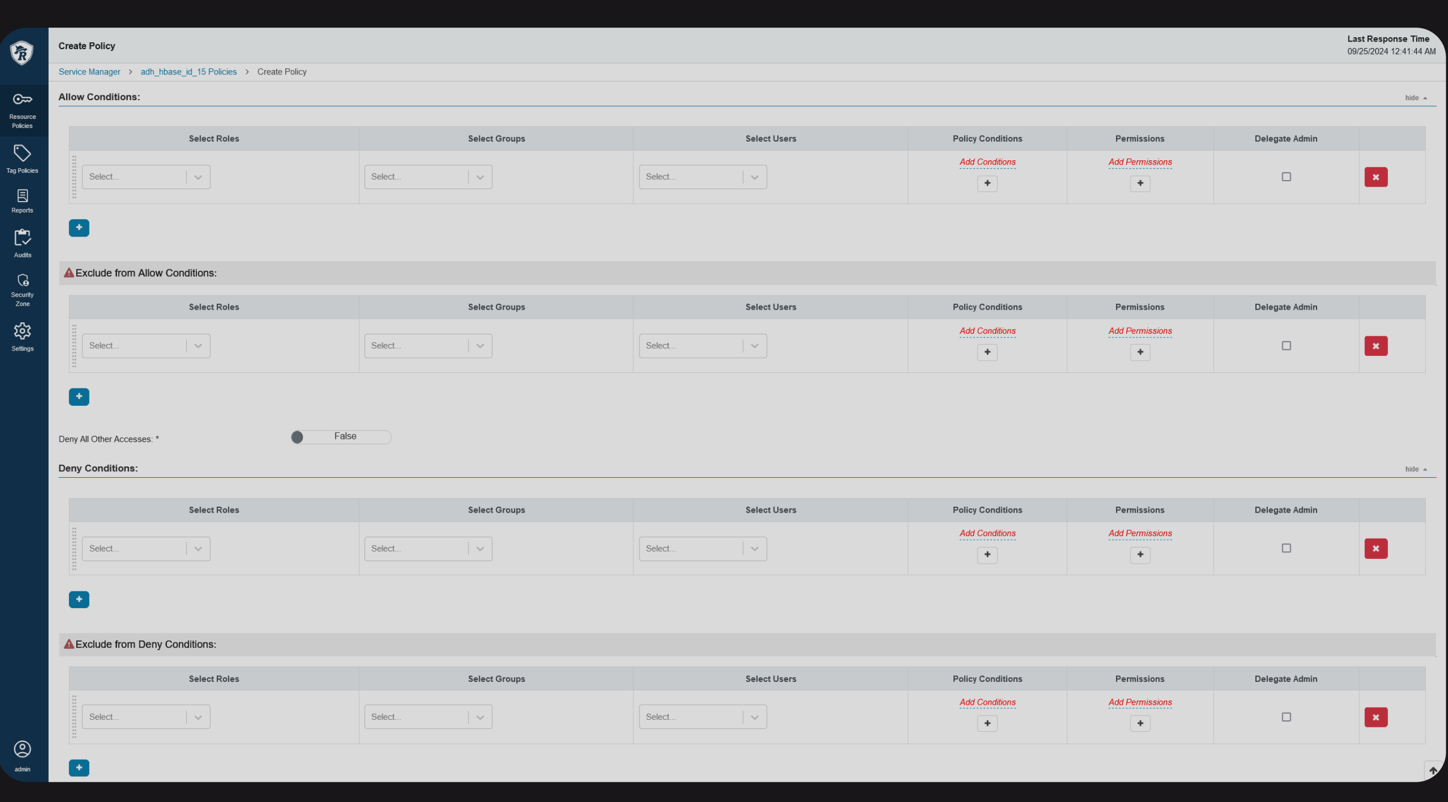
Create Policy
Service Manager › adh_impala_id_15 Policies › Create Policy
Policy Details :
Policy Type
Access
Policy Name*
Policy Label
Description
Audit Logging*
Enabled
Normal
Add Validity Period
Policy Conditions :
No Conditions
Resources :
Hive Database
Hive Table
Hive Column
Include

Параметры политики Impala

Параметры политики

Параметр	Описание
Policy Name	Название политики. Должно быть уникальным в системе
Enabled	Флаг, позволяющий применить политику сразу после создания
Normal/Override	Позволяет применить override-политику. Если выбран режим ***override***, то права доступа новой политики переписывают права доступа существующих политик
Policy Label	Позволяет группировать несколько политик с метками, что дает возможность искать политики по меткам. Поиском можно воспользоваться на страницах ***Policy listing*** и ***Reports***. Также упрощает экспорт/импорт политик — если требуется экспортировать некоторые политики, то можно провести поиск по меткам и экспортировать результат
database/url/hiveservice/global	Если выбран параметр ***database***, то введите имя соответствующей базы данных. Функция автозаполнения отображает доступные базы данных на основе введенного текста. Если выбран параметр ***url***, то введите путь к облачному хранилищу (например, *s3a://dev-admin/demo/campaigns.txt*), где необходимо разрешение конечного пользователя для чтения/записи данных Impala. Опция ***hiveservice*** позволяет пользователю, имеющему права Service Admin в Ranger, запустить API остановки запроса: `kill query <queryID>`. Опция ***global*** используется для создания глобальных политик, которые используют шаблоны соответствия и определяют необходимый доступ по умолчанию
table/udf/none	Определяет политику на основе таблицы или UDF
column/none	Название колонки Impala
Description	Описание назначения политик
Audit Logging	Включает аудит для политики
Add Validity Period	Позволяет указать время начала и окончания действия политики

4. Установите параметры разрешений/запретов. Секция ***Allow Conditions*** позволяет разрешать доступ определенным ролям/группам/пользователям. Имеет смысл использовать эту секцию, когда требуется разрешить доступ небольшому числу пользователей, а остальным — запретить. Секция ***Deny Conditions*** позволяет запрещать доступ определенным ролям/группам/пользователями. Её удобно использовать, когда требуется запретить доступ небольшому числу пользователей, а остальным — разрешить. Также можно комбинировать правила из обеих секций. Для добавления дополнительных условий кликните **+**. Условия интерпретируются в порядке, указанном в политике. Сначала применяется условие, находящееся в начале списка, затем второе, затем третье и так далее.



Create Policy
Service Manager › adh_hbase_id_15 Policies › Create Policy
Allow Conditions:
Select Roles
Select Groups
Select Users
Policy Conditions
Permissions
Delegate Admin
Add Conditions
Add Permissions
Exclude from Allow Conditions:
Deny All Other Accesses: *
False
Deny Conditions:
Exclude from Deny Conditions:

Параметры доступа к политике Impala

Параметры Allow Conditions

Параметр	Описание
Select Role	Укажите роли, к которым применяется данная политика
Select Group	Укажите группы, к которым применяется эта политика. Публичная группа содержит всех пользователей, поэтому предоставление доступа к публичной группе дает доступ всем пользователям
Select User	Укажите пользователей, к которым применяется данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

Параметры Deny Conditions

Параметр	Описание
Select Role	Укажите роли, к которым не будет применяться данная политика
Select Group	Укажите группы, к которым не будет применяться эта политика. Публичная группа содержит всех пользователей, поэтому запрет доступа к публичной группе запрещает доступ всем пользователям
Select User	Укажите пользователей, к которым не будет применяться данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

5. Кликните ***Add*** внизу страницы.

Плагин Ranger Ozone

Сергей Тихомиров

Содержание

Назад к содержанию

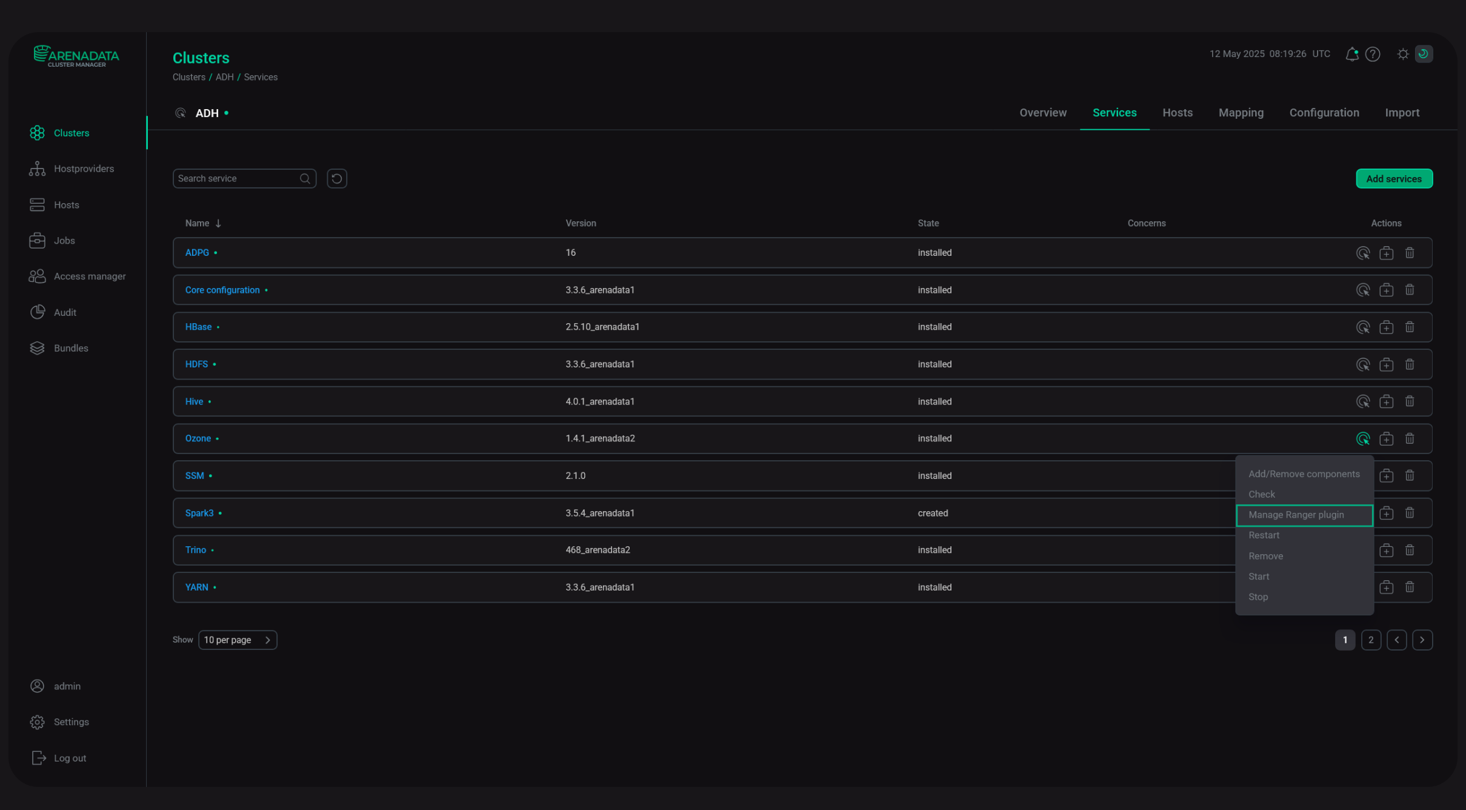
Включение плагина Ozone

Добавление новой политики в Ranger

Включение плагина Ozone

Чтобы включить плагин Ranger для Ozone, выполните следующие действия.

1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку в строке Ozone и выберите действие ***Manage Ranger plugin*** в выпадающем меню.



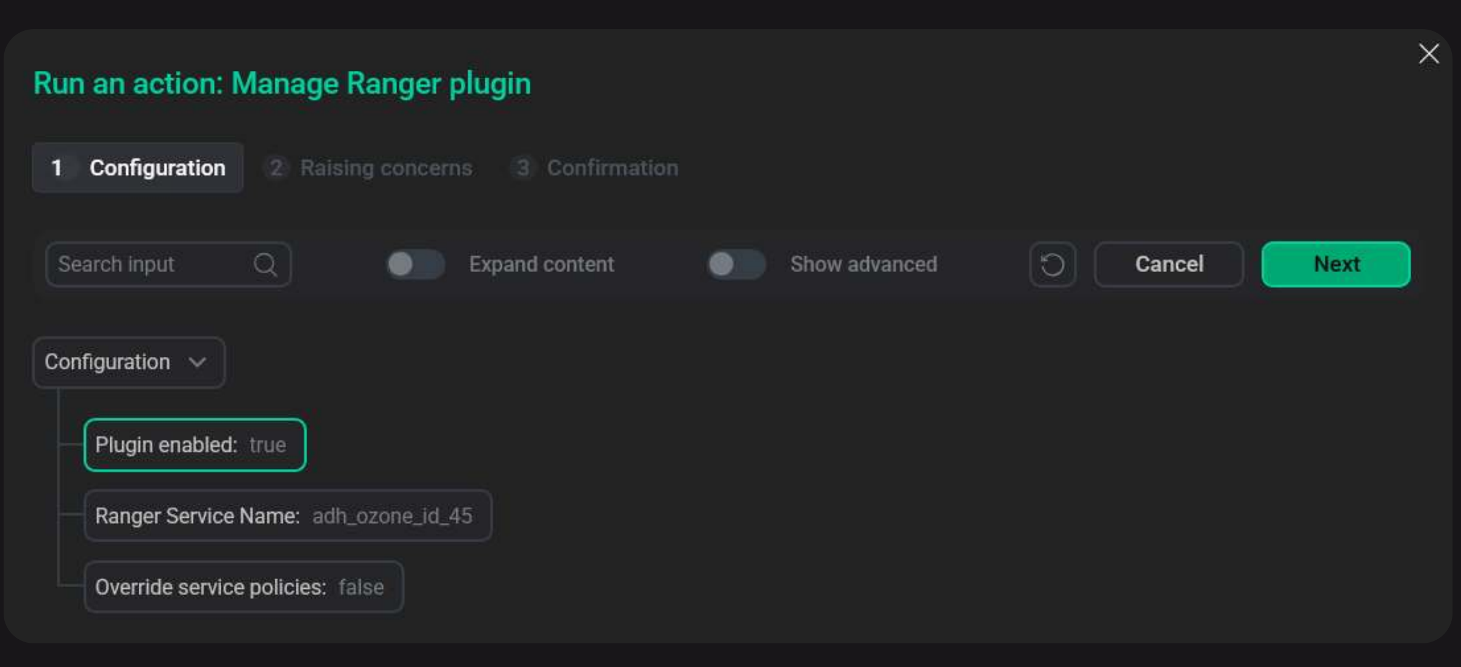
Clusters / ADH / Services

Name	Version	State	Concerns	Actions
ADPG	16	installed		
Core configuration	3.3.6_arenadata1	installed		
HBase	2.5.10_arenadata1	installed		
HDFS	3.3.6_arenadata1	installed		
Hive	4.0.1_arenadata1	installed		
Ozone	1.4.1_arenadata2	installed		
SSM	2.1.0	installed		
Spark3	3.5.4_arenadata1	created		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		

ARENADATA CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
12 May 2025 08:19:26 UTC
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add services
Add/Remove components
Check
Manage Ranger plugin
Restart
Remove
Start
Stop
Show 10 per page
admin
Settings
Log out

Сервисы кластера

3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.



Run an action: Manage Ranger plugin

Configuration

Plugin enabled: true

Ranger Service Name: adh_ozone_id_45

Override service policies: false

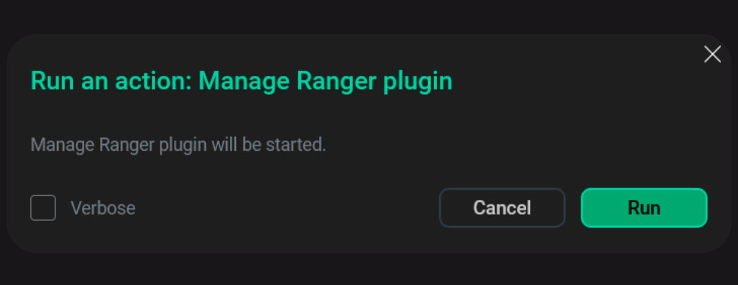
1 Configuration
2 Raising concerns
3 Confirmation
Search input
Expand content
Show advanced
Cancel
Next

Состояние плагина

ПРИМЕЧАНИЕ

При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.



Run an action: Manage Ranger plugin

Manage Ranger plugin will be started.

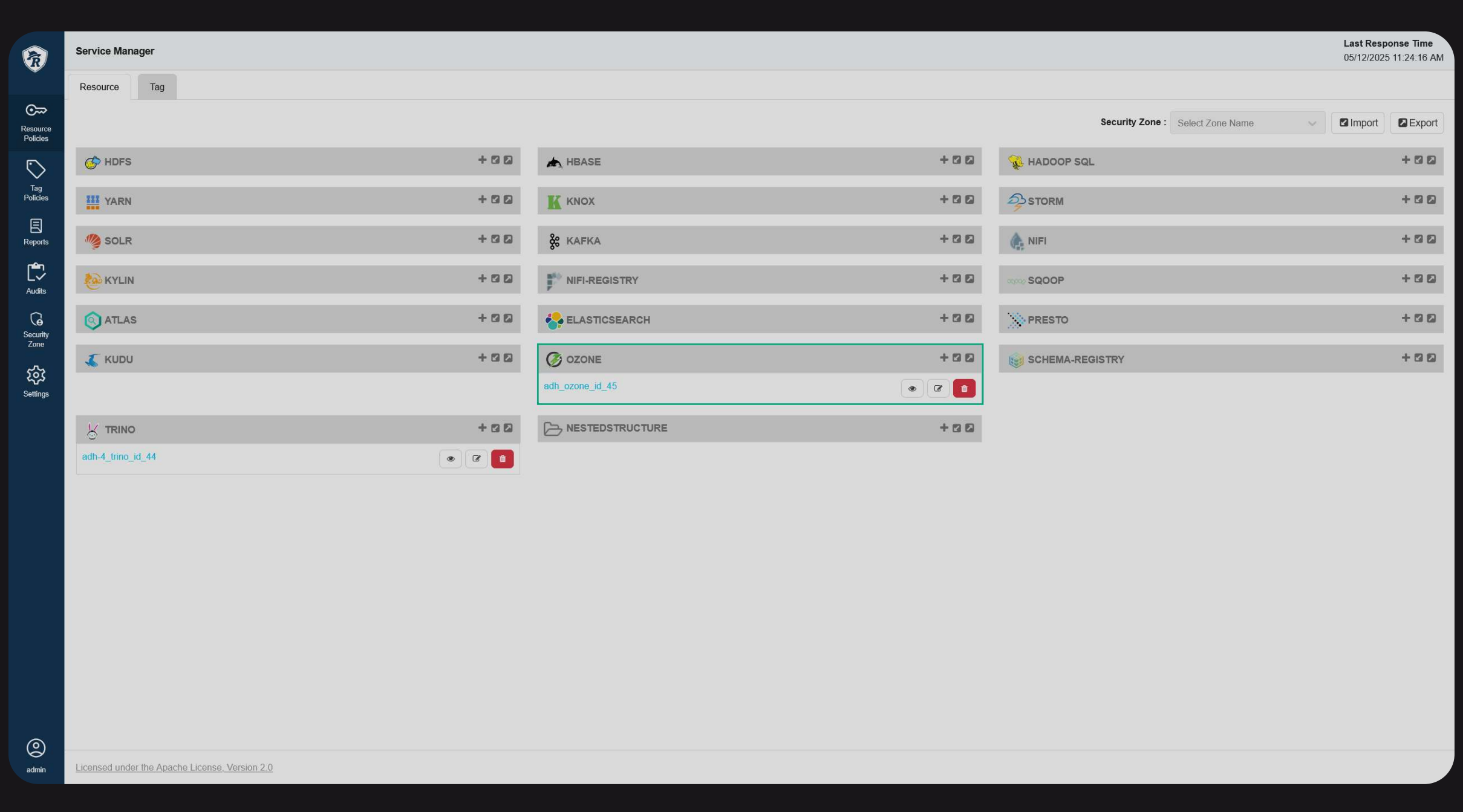
☐ Verbose
Cancel
Run

Подтверждение действия

Добавление новой политики в Ranger

Чтобы добавить новую политику к существующему сервису Ozone, необходимо выполнить следующие действия:

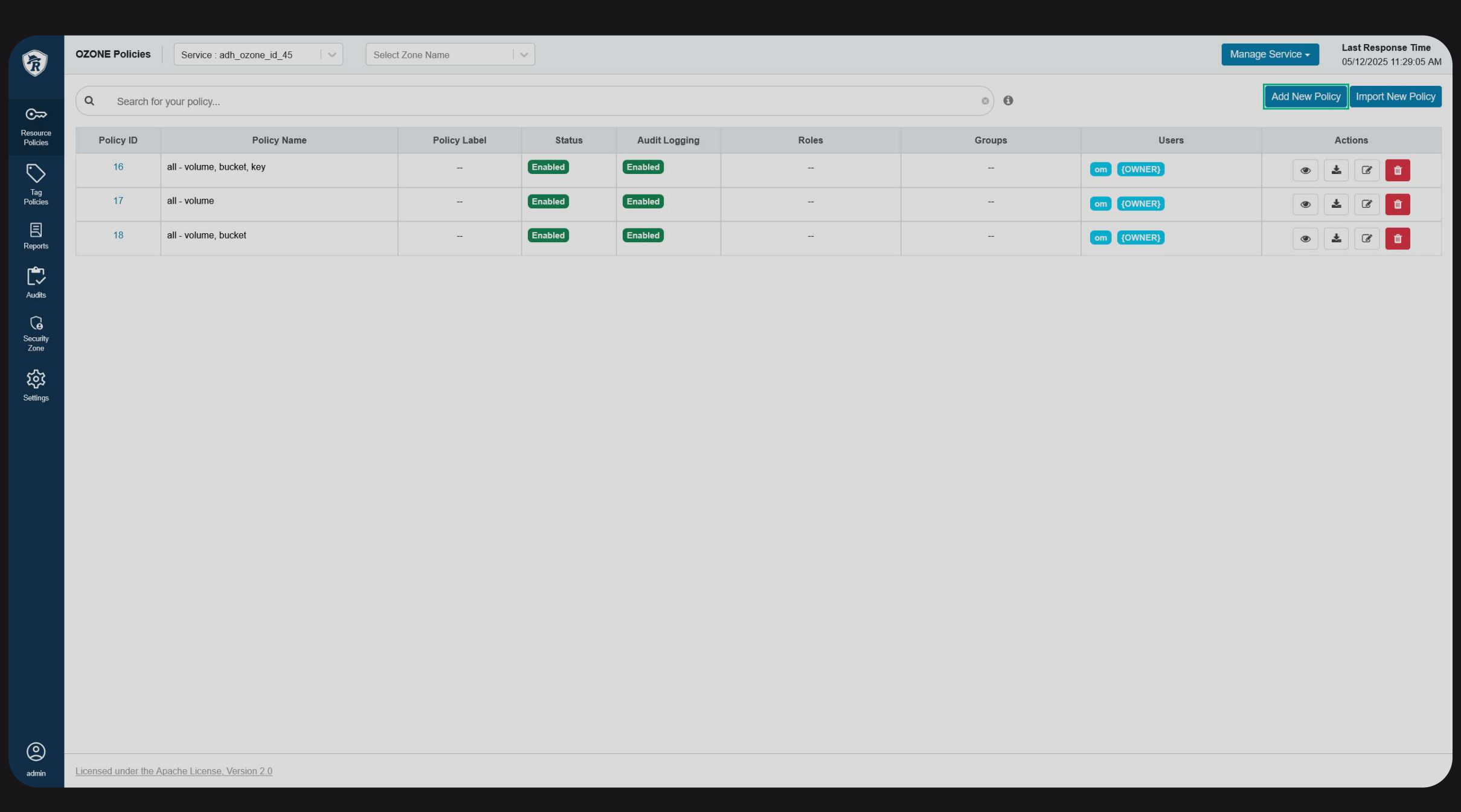
1. На странице ***Service Manager*** нажмите на существующий сервис Ozone в панели Ozone.



Service Manager
Resource
Tag
Security Zone :
Select Zone Name
Import
Export
HDFS
YARN
SOLR
KYLIN
ATLAS
KUDU
TRINO
adh-4_trino_id_44
HBASE
KNOX
KAFKA
NIFI-REGISTRY
ELASTICSEARCH
OZONE
adh_ozone_id_45
NESTEDSTRUCTURE
HADOOP SQL
STORM
NIFI
SQOOP
PRESTO
SCHEMA-REGISTRY
Last Response Time
05/12/2025 11:24:16 AM
Licensed under the Apache License, Version 2.0

Страница Service Manager

2. На странице политик Ozone нажмите ***Add New Policy***.



OZONE Policies

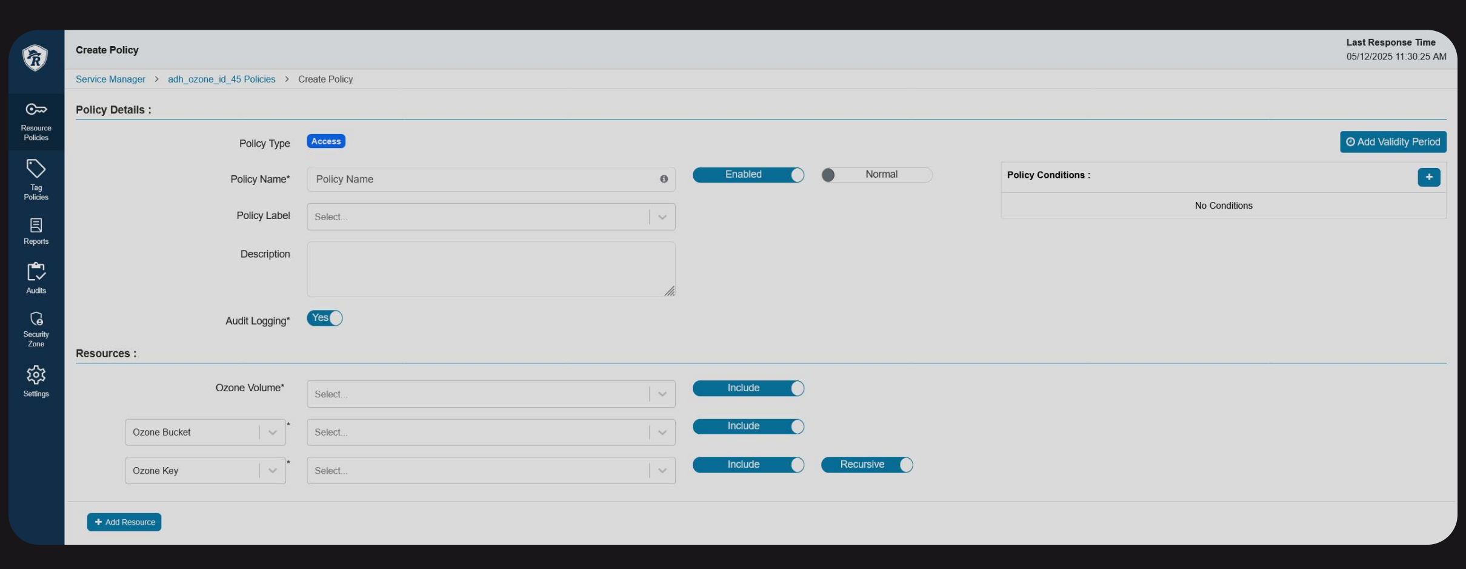
Service : adh_ozone_id_45

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users	Actions
16	all - volume, bucket, key	–	Enabled	Enabled	–	–	om {OWNER}	
17	all - volume	–	Enabled	Enabled	–	–	om {OWNER}	
18	all - volume, bucket	–	Enabled	Enabled	–	–	om {OWNER}	

Select Zone Name
Manage Service
Last Response Time
05/12/2025 11:29:05 AM
Search for your policy...
Add New Policy
Import New Policy
Licensed under the Apache License, Version 2.0

Добавление новой политики

3. На открывшейся странице ***Create Policy*** заполните необходимые поля.



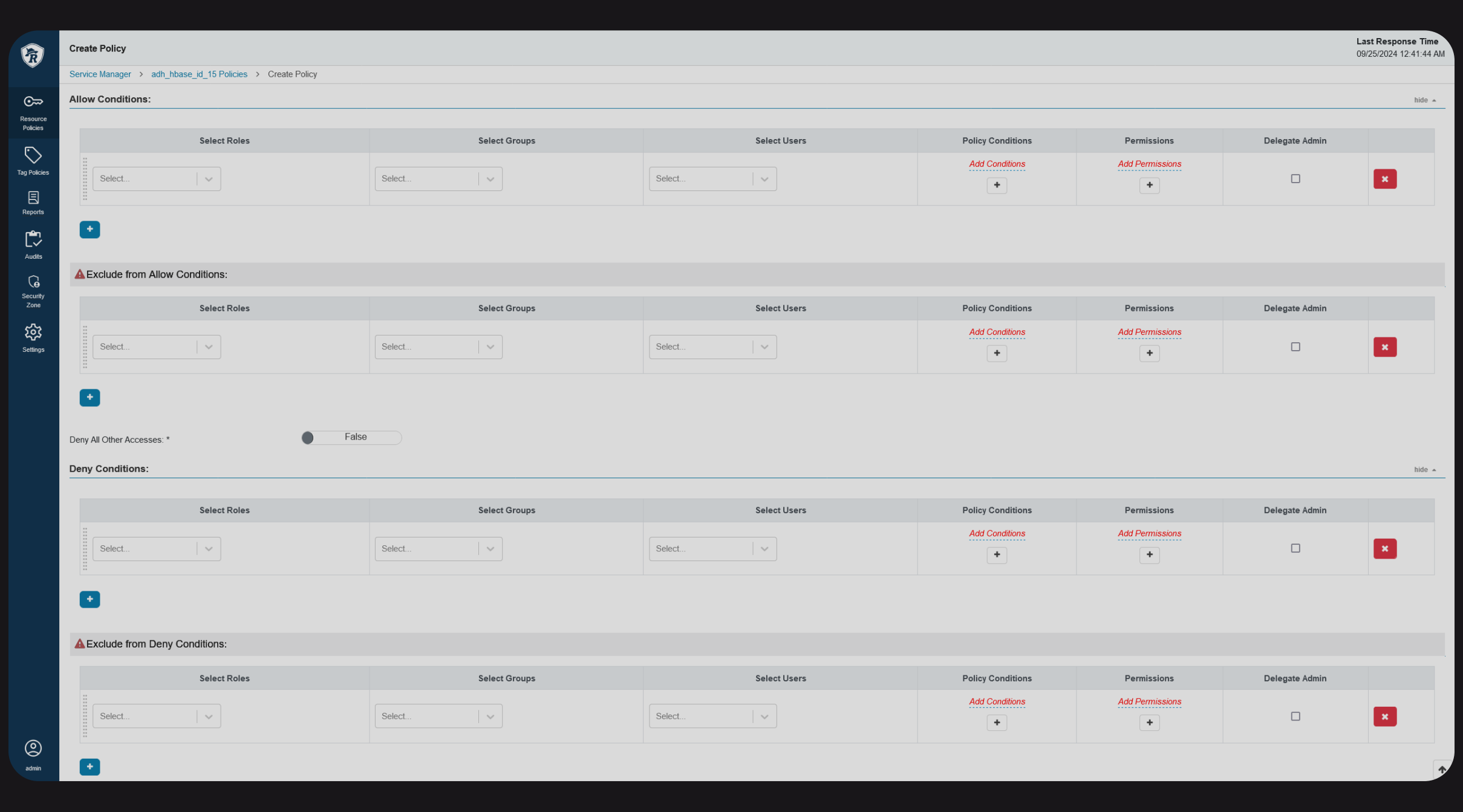
Create Policy
Service Manager > adh_ozone_id_45 Policies > Create Policy
Last Response Time
05/12/2025 11:30:25 AM
Policy Details :
Policy Type
Access
Policy Name*
Policy Label
Description
Audit Logging*
Yes
Enabled
Normal
Add Validity Period
Policy Conditions :
No Conditions
Resources :
Ozone Volume*
Ozone Bucket
Ozone Key
Select...
Include
Recursive
Add Resource

Параметры политики Ozone

Параметры политики

Параметр	Описание
Policy Name	Название политики. Должно быть уникальным в системе
Enabled	Флаг, позволяющий применить политику сразу после создания
Normal/Override	Позволяет применить override-политику. Если выбран режим ***override***, то права доступа новой политики переписывают права доступа существующих политик
Policy Label	Позволяет группировать несколько политик с метками, что дает возможность искать политики по меткам. Поиском можно воспользоваться на страницах ***Policy listing*** и ***Reports***. Также упрощает экспорт/импорт политик — если требуется экспортировать некоторые политики, то можно провести поиск по меткам и экспортировать результат
Description	Описание назначения политик
Audit Logging	Включает аудит для политики
Add Validity Period	Позволяет указать время начала и окончания действия политики
Ozone Volume	Наименование тома в Ozone
Ozone Bucket	Наименование бакета в Ozone
Ozone Key	Наименование ключа в Ozone

4. Установите параметры разрешений/запретов. Секция ***Allow Conditions*** позволяет разрешать доступ определенным ролям/группам/пользователям. Имеет смысл использовать эту секцию, когда требуется разрешить доступ небольшому числу пользователей, а остальным — запретить. Секция ***Deny Conditions*** позволяет запрещать доступ определенным ролям/группам/пользователями. Её удобно использовать, когда требуется запретить доступ небольшому числу пользователей, а остальным — разрешить. Также можно комбинировать правила из обеих секций. Для добавления дополнительных условий кликните +. Условия интерпретируются в порядке, указанном в политике. Сначала применяется условие, находящееся в начале списка, затем второе, затем третье и так далее.



Create Policy
Service Manager > adh_hbase_id_15 Policies > Create Policy
Last Response Time
09/25/2024 12:41:44 AM
Allow Conditions:
Exclude from Allow Conditions:
Deny All Other Accesses: *
False
Deny Conditions:
Exclude from Deny Conditions:
Select Roles
Select Groups
Select Users
Policy Conditions
Permissions
Delegate Admin
Add Conditions
Add Permissions
Select...

Параметры доступа к политике Ozone

Параметры Allow Conditions

Параметр	Описание
Select Role	Укажите роли, к которым применяется данная политика
Select Group	Укажите группы, к которым применяется эта политика. Публичная группа содержит всех пользователей, поэтому предоставление доступа к публичной группе дает доступ всем пользователям
Select User	Укажите пользователей, к которым применяется данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

Параметры Deny Conditions

Параметр	Описание
Select Role	Укажите роли, к которым не будет применяться данная политика
Select Group	Укажите группы, к которым не будет применяться эта политика. Публичная группа содержит всех пользователей, поэтому запрет доступа к публичной группе запрещает доступ всем пользователям
Select User	Укажите пользователей, к которым не будет применяться данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

5. Кликните ***Add*** внизу страницы.

Плагин Ranger Solr

Сергей Тихомиров

Содержание

Назад к содержанию

Включение плагина Solr

Добавление новой политики в Ranger

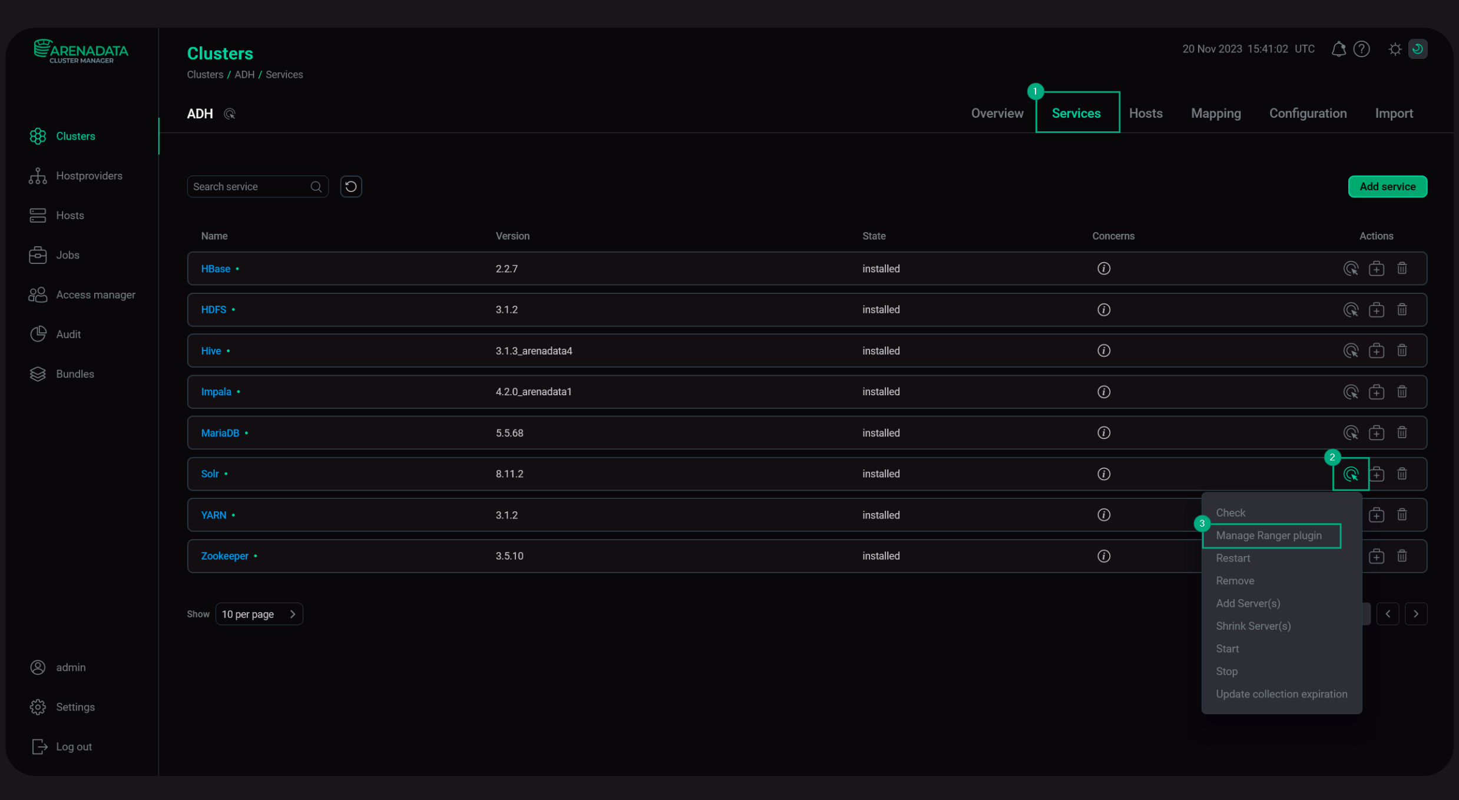
Включение плагина Solr

ВАЖНО

Solr не имеет базовой аутентификации, поэтому плагин Ranger для него доступен только в керберизованном кластере ADH.

Чтобы включить плагин Ranger для Solr, выполните следующие действия.

1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку в строке Solr и выберите действие ***Manage Ranger plugin*** в выпадающем меню.

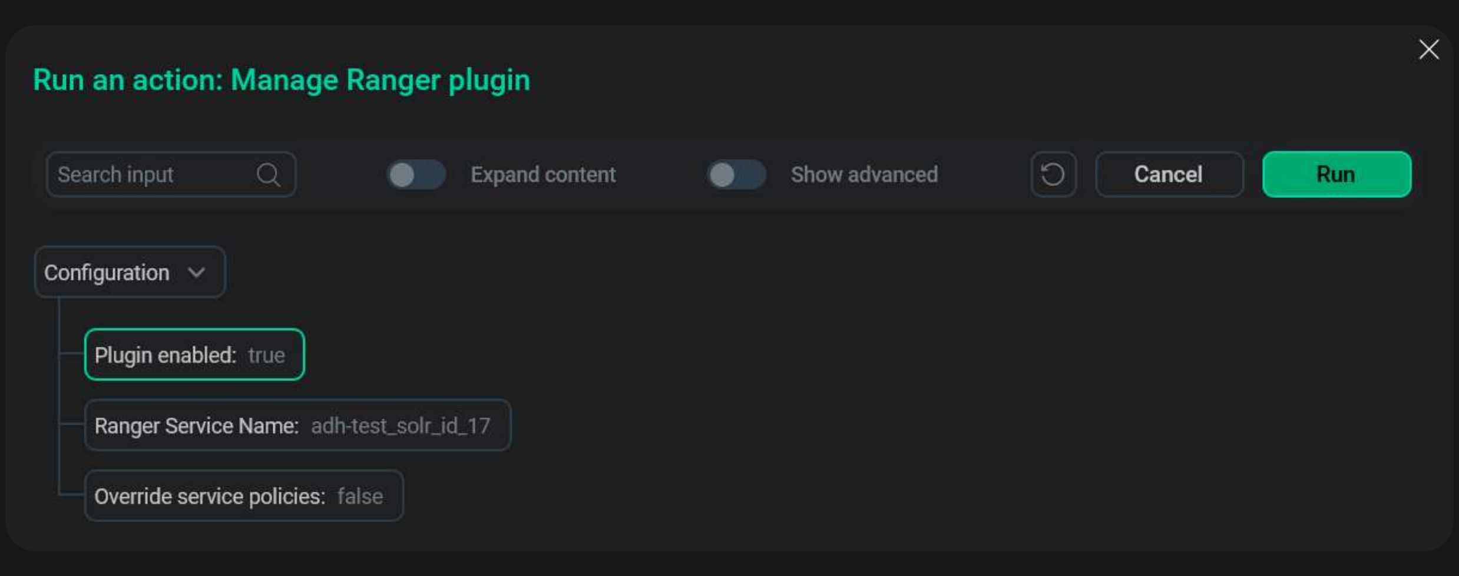


ARENADATA CLUSTER MANAGER
Clusters
Clusters / ADH / Services
20 Nov 2023 15:41:02 UTC
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
Search service
Add service
1
2
3
Check
Manage Ranger plugin
Restart
Remove
Add Server(s)
Shrink Server(s)
Start
Stop
Update collection expiration
Show
10 per page
admin
Settings
Log out

Name	Version	State
HBase	2.2.7	installed
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
Impala	4.2.0_arenadata1	installed
MariaDB	5.5.68	installed
Solr	8.11.2	installed
YARN	3.1.2	installed
Zookeeper	3.5.10	installed

Сервисы кластера

3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.



Run an action: Manage Ranger plugin
Search input
Expand content
Show advanced
Cancel
Run
Configuration

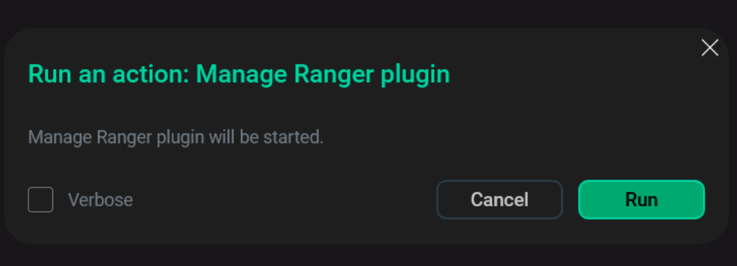
Plugin enabled: true
Ranger Service Name: adh-test_solr_id_17
Override service policies: false

Состояние плагина

ПРИМЕЧАНИЕ

При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.



Run an action: Manage Ranger plugin
☐ Verbose
Cancel
Run

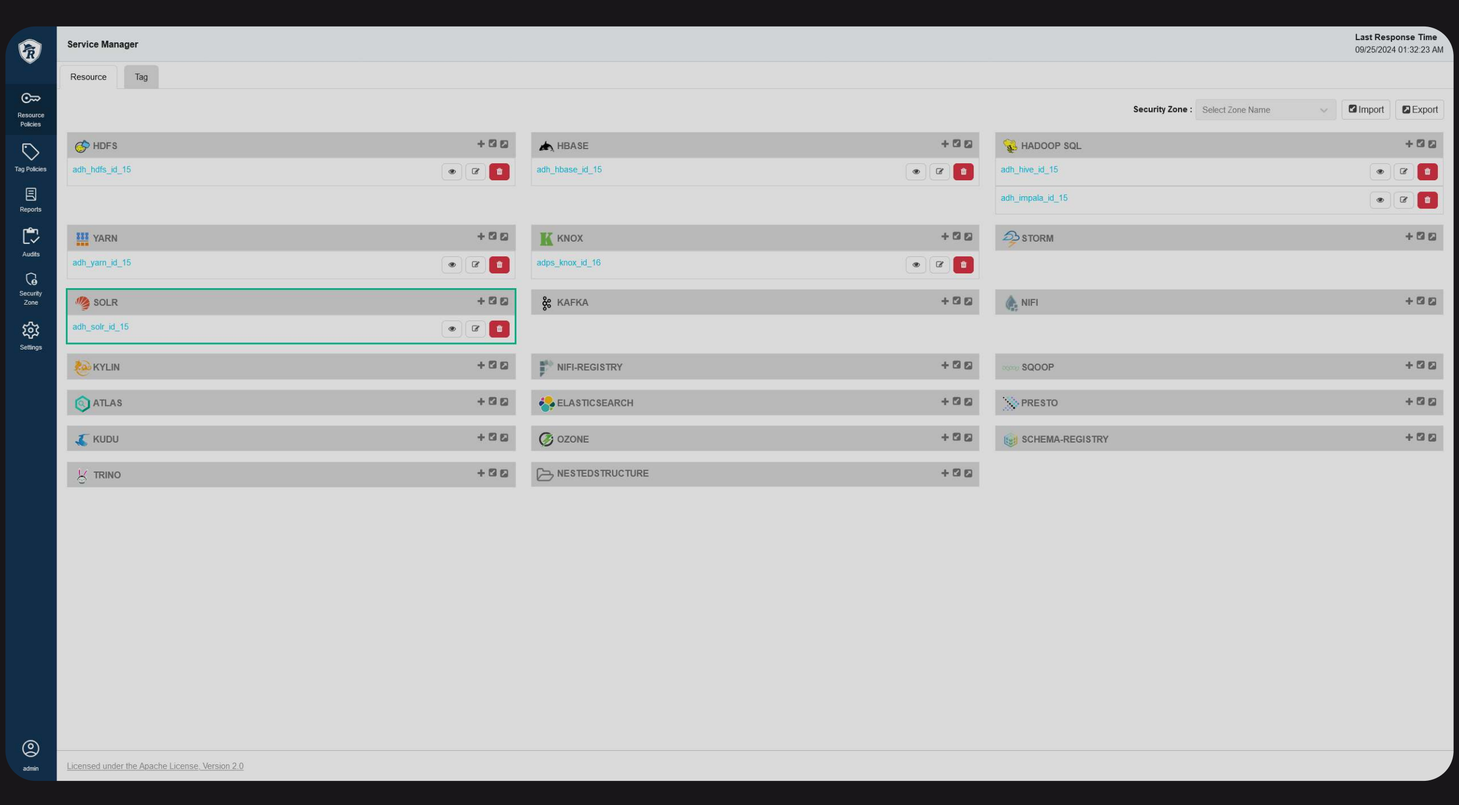
Manage Ranger plugin will be started.

Подтверждение действия

Добавление новой политики в Ranger

Чтобы добавить новую политику к существующему сервису Solr, необходимо выполнить следующие действия:

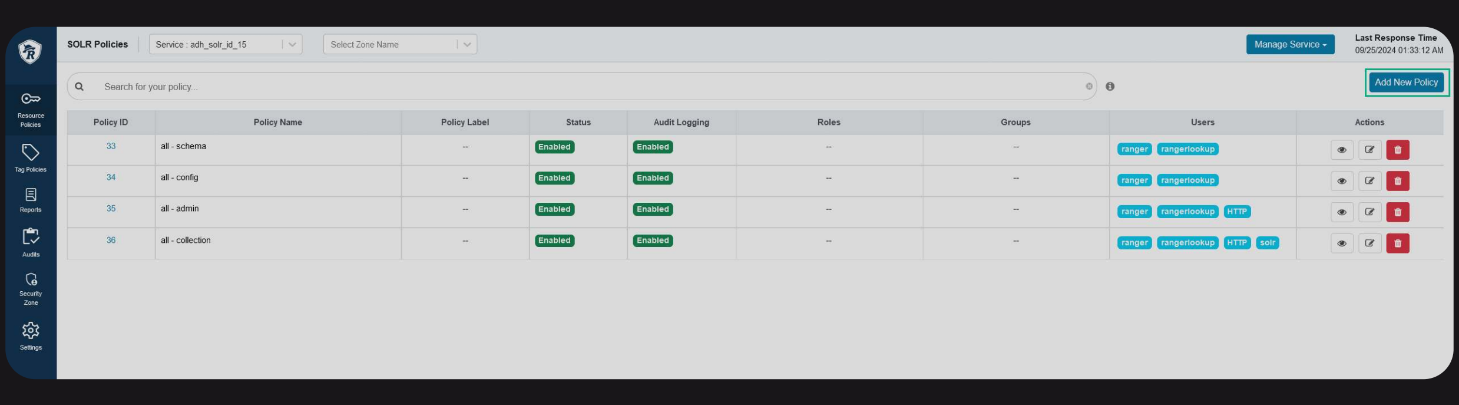
1. На странице ***Service Manager*** нажмите на существующий сервис Solr в панели Hadoop SQL.



Service Manager
Resource
Tag
Security Zone :
Select Zone Name
Import
Export
HDFS
adh_hdfs_id_15
HBASE
adh_hbase_id_15
HADOOP SQL
adh_hive_id_15
adh_impala_id_15
YARN
adh_yarn_id_15
KNOX
adps_knox_id_16
STORM
SOLR
adh_solr_id_15
KAFKA
NIFI
KYLIN
NIFI-REGISTRY
SQOOP
ATLAS
ELASTICSEARCH
PRESTO
KUDU
OZONE
SCHEMA-REGISTRY
TRINO
NESTEDSTRUCTURE
Licensed under the Apache License, Version 2.0

Страница Service Manager

2. На странице политик Solr нажмите ***Add New Policy***.

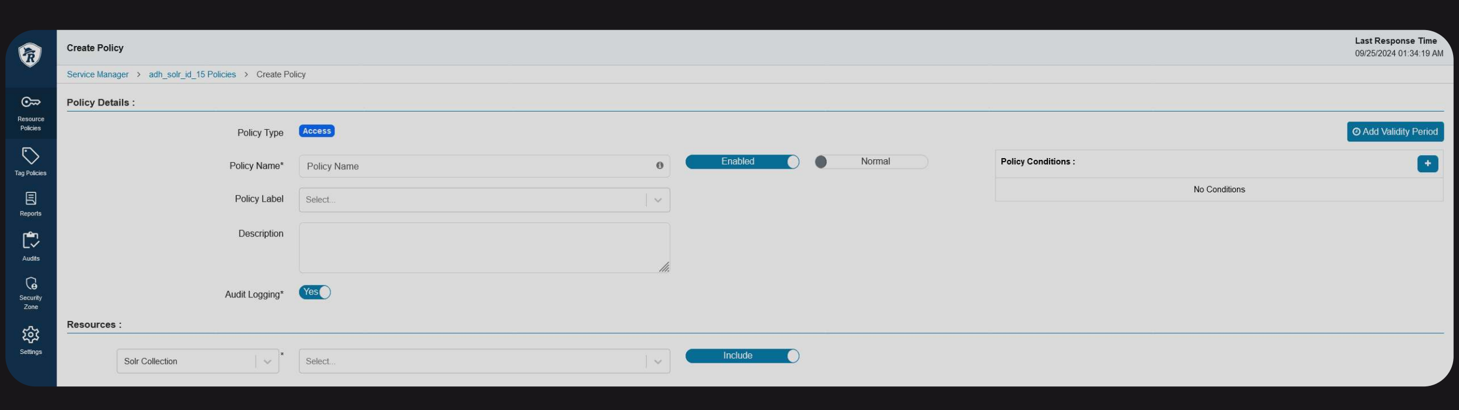


SOLR Policies
Service : adh_solr_id_15
Select Zone Name
Manage Service
Last Response Time
09/25/2024 01:33:12 AM
Search for your policy...
Add New Policy

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users
33	all - schema	--	Enabled	Enabled	--	--	ranger rangerlookup
34	all - config	--	Enabled	Enabled	--	--	ranger rangerlookup
35	all - admin	--	Enabled	Enabled	--	--	ranger rangerlookup HTTP
36	all - collection	--	Enabled	Enabled	--	--	ranger rangerlookup HTTP solr

Добавление новой политики

3. На открывшейся странице ***Create Policy*** заполните необходимые поля.



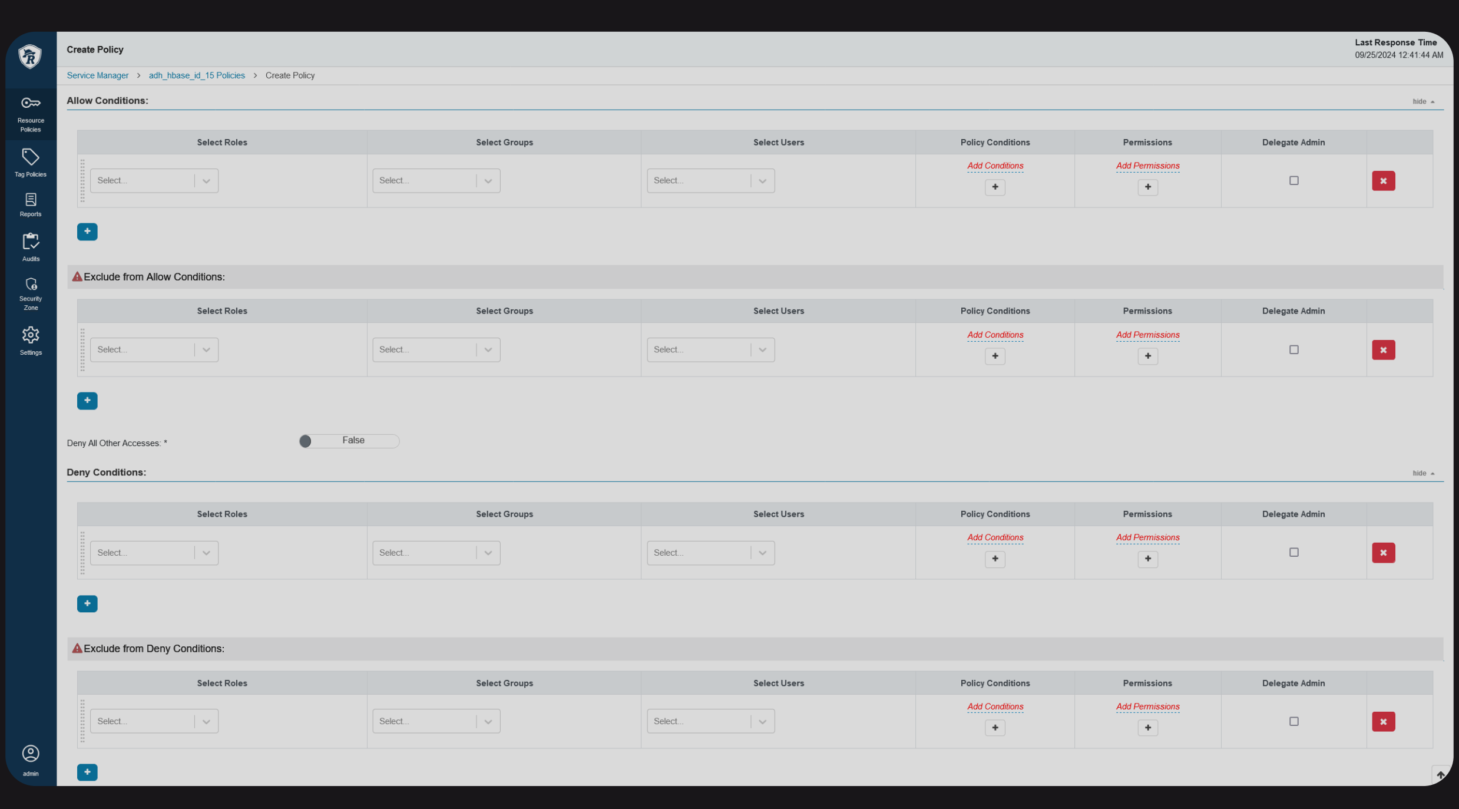
Create Policy
Service Manager > adh_solr_id_15 Policies > Create Policy
Last Response Time
09/25/2024 01:34:19 AM
Policy Details :
Policy Type
Access
Policy Name*
Policy Label
Description
Audit Logging*
Yes
Enabled
Normal
Add Validity Period
Policy Conditions :
No Conditions
Resources :
Solr Collection
Select...
Include

Параметры политики Solr

Параметры политики

Параметр	Описание
Policy Name	Название политики. Должно быть уникальным в системе
Enabled	Флаг, позволяющий применить политику сразу после создания
Normal/Override	Позволяет применить override-политику. Если выбран режим ***override***, то права доступа новой политики переписывают права доступа существующих политик
Policy Label	Позволяет группировать несколько политик с метками, что дает возможность искать политики по меткам. Поиском можно воспользоваться на страницах ***Policy listing*** и ***Reports***. Также упрощает экспорт/импорт политик — если требуется экспортировать некоторые политики, то можно провести поиск по меткам и экспортировать результат
Solr Collection	URL коллекции Solr (например, *http://<solr_host>:6083/solr/ranger_audits*)
Description	Описание назначения политик
Audit Logging	Включает аудит для политики
Add Validity Period	Позволяет указать время начала и окончания действия политики

4. Установите параметры разрешений/запретов. Секция ***Allow Conditions*** позволяет разрешать доступ определенным ролям/группам/пользователям. Имеет смысл использовать эту секцию, когда требуется разрешить доступ небольшому числу пользователей, а остальным — запретить. Секция ***Deny Conditions*** позволяет запрещать доступ определенным ролям/группам/пользователями. Её удобно использовать, когда требуется запретить доступ небольшому числу пользователей, а остальным — разрешить. Также можно комбинировать правила из обеих секций. Для добавления дополнительных условий кликните **+**. Условия интерпретируются в порядке, указанном в политике. Сначала применяется условие, находящееся в начале списка, затем второе, затем третье и так далее.



Create Policy
Service Manager > adh_hbase_id_15 Policies > Create Policy
Last Response Time
09/25/2024 12:41:44 AM
Allow Conditions:
Exclude from Allow Conditions:
Deny All Other Accesses: *
False
Deny Conditions:
Exclude from Deny Conditions:
Select Roles
Select Groups
Select Users
Policy Conditions
Permissions
Delegate Admin
Add Conditions
Add Permissions

Параметры доступа к политике Solr

Параметры Allow Conditions

Параметр	Описание
Select Role	Укажите роли, к которым применяется данная политика
Select Group	Укажите группы, к которым применяется эта политика. Публичная группа содержит всех пользователей, поэтому предоставление доступа к публичной группе дает доступ всем пользователям
Select User	Укажите пользователей, к которым применяется данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

Параметры Deny Conditions

Параметр	Описание
Select Role	Укажите роли, к которым не будет применяться данная политика
Select Group	Укажите группы, к которым не будет применяться эта политика. Публичная группа содержит всех пользователей, поэтому запрет доступа к публичной группе запрещает доступ всем пользователям
Select User	Укажите пользователей, к которым не будет применяться данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

5. Кликните ***Add*** внизу страницы.

Плагин Ranger Spark3

Сергей Тихомиров

Содержание

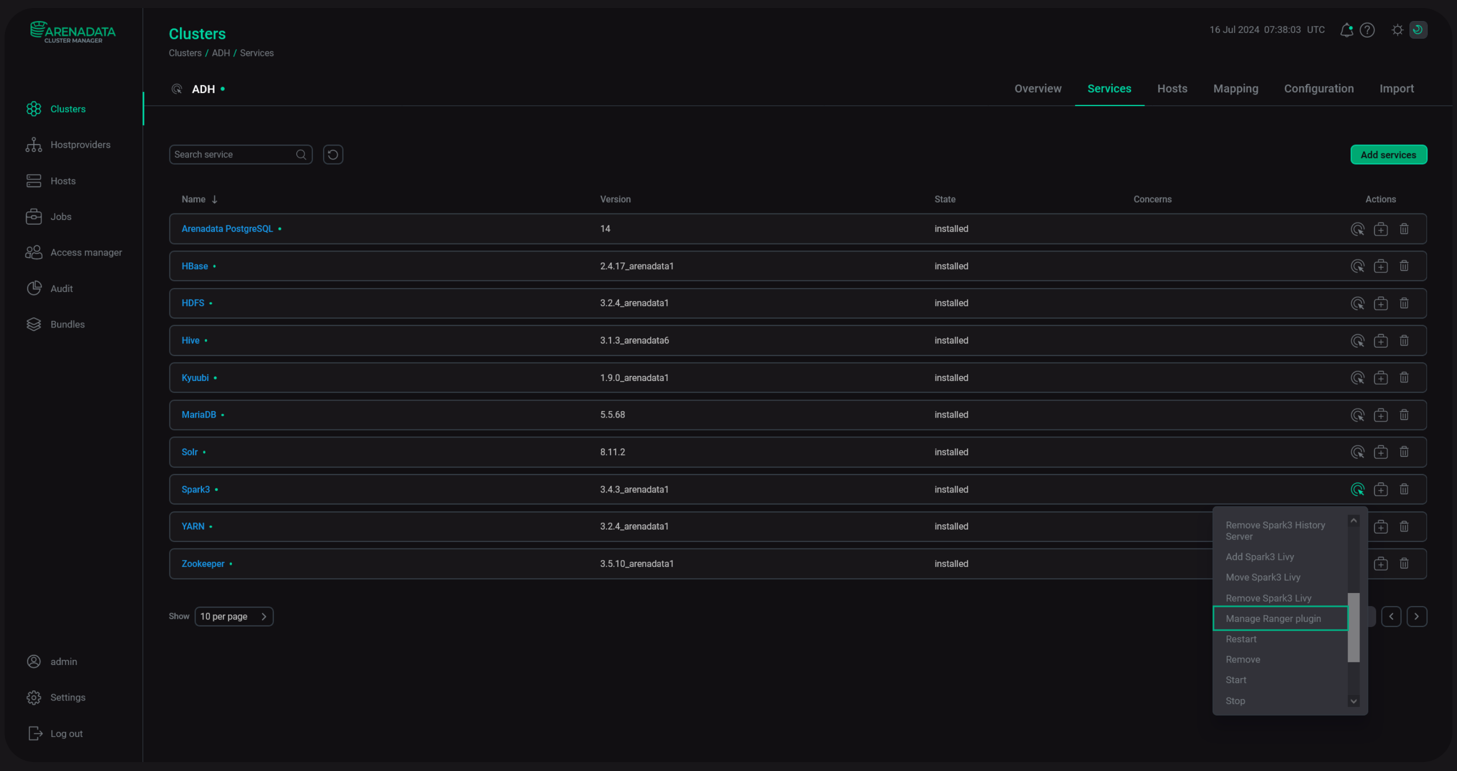
Назад к содержанию

Включение плагина Spark3

Включение плагина Spark3

Чтобы включить плагин Ranger для Spark3, выполните следующие действия.

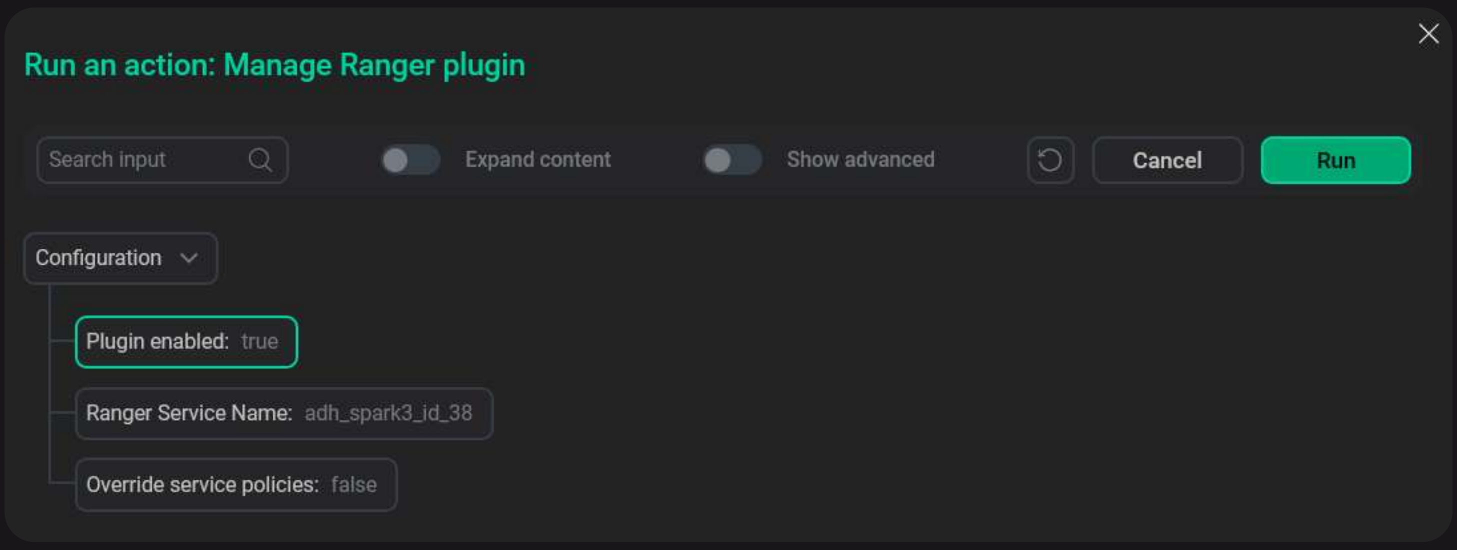
1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку в строке Spark3 и выберите действие ***Manage Ranger plugin*** в выпадающем меню.


ARENADATA
CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
Clusters / ADH / Services
16 Jul 2024 07:38:03 UTC
ADH
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add services
Show
10 per page
Remove Spark3 History Server
Add Spark3 Livy
Move Spark3 Livy
Remove Spark3 Livy
Manage Ranger plugin
Restart
Remove
Start
Stop

Name ↓	Version	State	Concerns	Actions
Arenadata PostgreSQL	14	installed		
HBase	2.4.17_arenadata1	installed		
HDFS	3.2.4_arenadata1	installed		
Hive	3.1.3_arenadata6	installed		
Kyuubi	1.9.0_arenadata1	installed		
MariaDB	5.5.68	installed		
Solr	8.11.2	installed		
Spark3	3.4.3_arenadata1	installed		
YARN	3.2.4_arenadata1	installed		
Zookeeper	3.5.10_arenadata1	installed		

Сервисы кластера

3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.


Run an action: Manage Ranger plugin
Search input
Expand content
Show advanced
Cancel
Run
Configuration

Plugin enabled: true

Ranger Service Name: adh_spark3_id_38

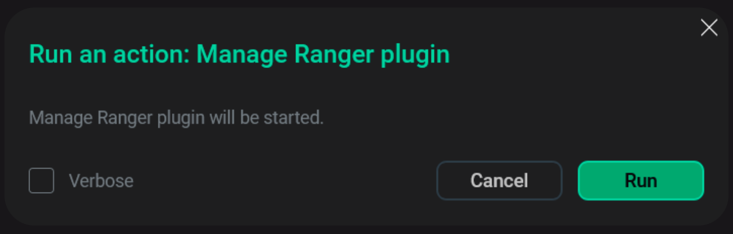
Override service policies: false

Состояние плагина

ПРИМЕЧАНИЕ

При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.


Run an action: Manage Ranger plugin
☐ Verbose
Cancel
Run

Manage Ranger plugin will be started.

Подтверждение действия

Плагин Ranger Trino

Сергей Тихомиров

Содержание

Назад к содержанию

Включение плагина Trino

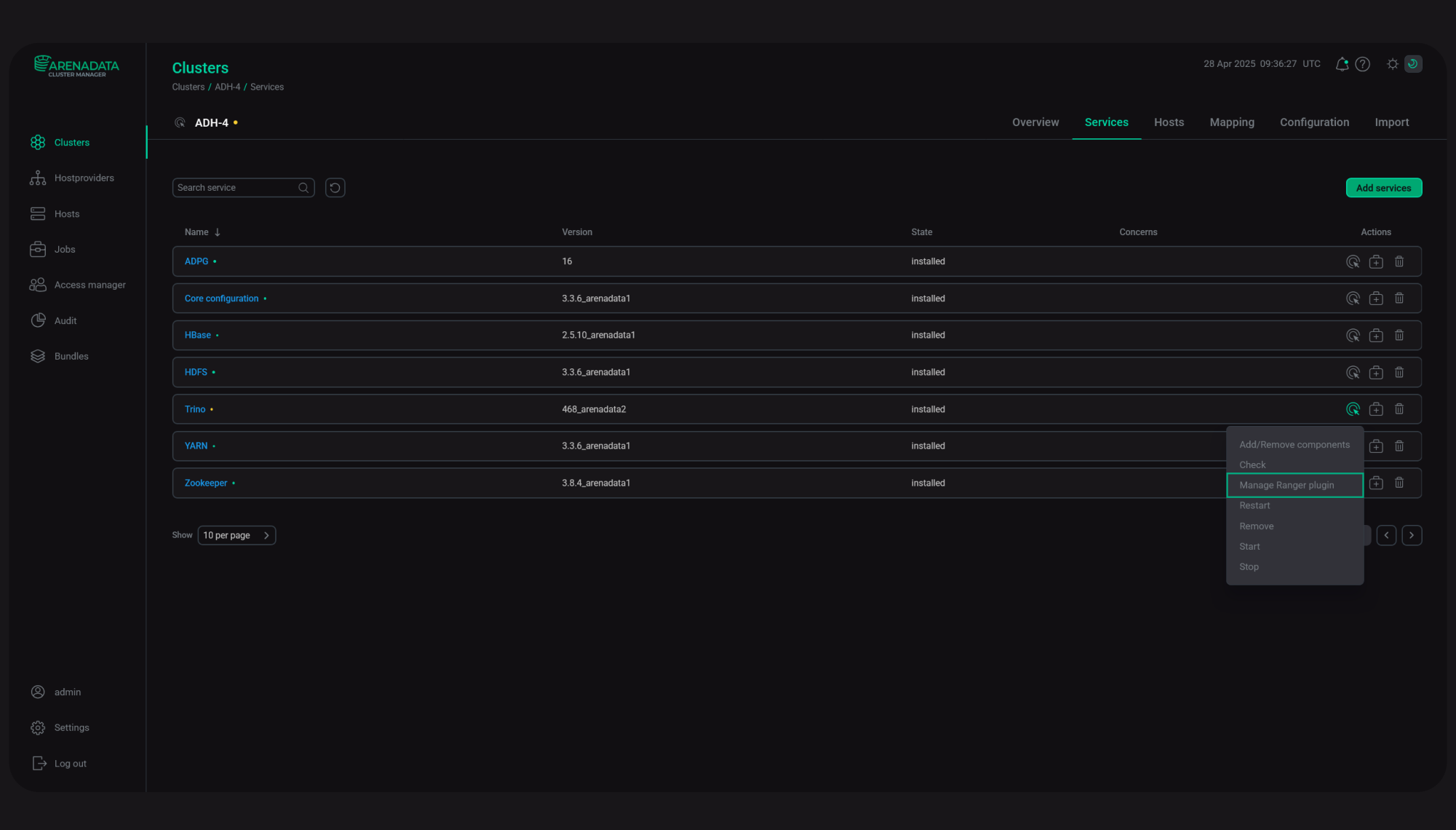
Соответствие ресурсов и типов доступа между Trino и Hive

Добавление новой политики в Ranger

Включение плагина Trino

Чтобы включить плагин Ranger для Trino, выполните следующие действия.

1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку в строке Trino и выберите действие ***Manage Ranger plugin*** в выпадающем меню.



Name	Version	State	Concerns	Actions
ADPG	16	installed		
Core configuration	3.3.6_arenadata1	installed		
HBase	2.5.10_arenadata1	installed		
HDFS	3.3.6_arenadata1	installed		
Trino	468_arenadata2	installed		
YARN	3.3.6_arenadata1	installed		
Zookeeper	3.8.4_arenadata1	installed		

Clusters
Clusters / ADH-4 / Services
ADH-4
Overview
Services
Hosts
Mapping
Configuration
Import
Search service
Add services
Hostproviders
Jobs
Access manager
Audit
Bundles
Add/Remove components
Check
Manage Ranger plugin
Restart
Remove
Start
Stop
Show 10 per page
admin
Settings
Log out

Сервисы кластера

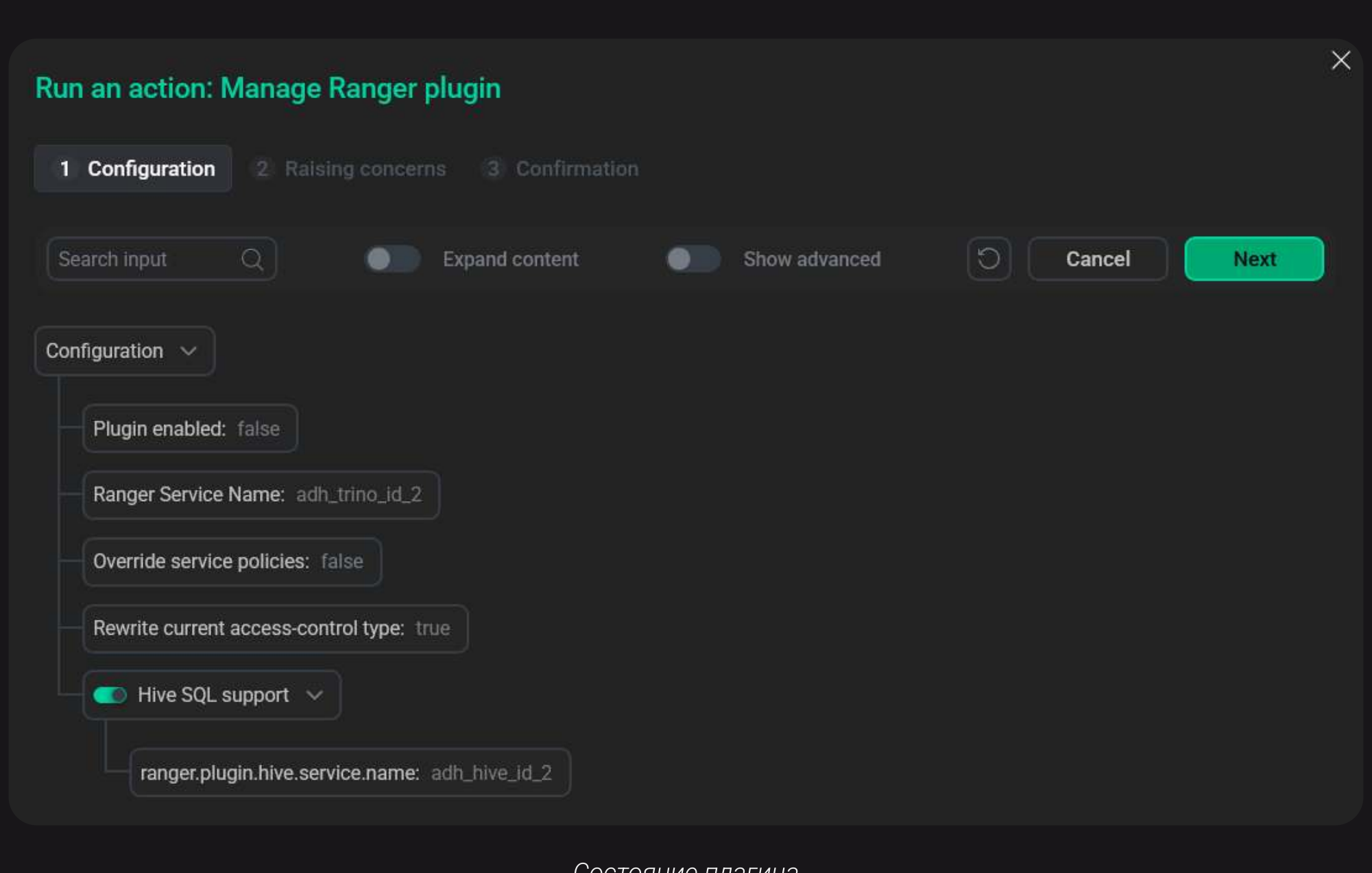
3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.

ВАЖНО

Чтобы параметр ***Hive SQL Support*** был активен, необходимо включить плагин Ranger Hive. Данный параметр позволяет использовать общую политику Hadoop SQL совместно с такими сервисами, как Hive/Impala/SparkSQL.

Политики применяются в следующем порядке:

1. Политики, специфичные для Trino, применяются первыми.
2. Если политика определена в двух местах, применяется политика Trino.
3. Если политика определена только в одном месте, она применяется.



Run an action: Manage Ranger plugin

- Plugin enabled: false
- Ranger Service Name: adh_trino_id_2
- Override service policies: false
- Rewrite current access-control type: true
- Hive SQL support
  - ranger.plugin.hive.service.name: adh_hive_id_2

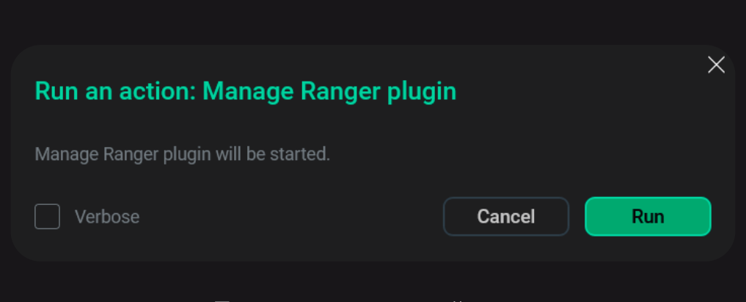
1 Configuration
2 Raising concerns
3 Confirmation
Search input
Expand content
Show advanced
Cancel
Next
Configuration

Состояние плагина

ПРИМЕЧАНИЕ

При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.



Run an action: Manage Ranger plugin

Manage Ranger plugin will be started.

Verbose
Cancel
Run

Подтверждение действия

Необходимые политики

- Пользователи должны иметь права на выполнение запросов в Trino. Без политики в Apache Ranger, предоставляющей эти права, пользователи не смогут выполнять какие-либо запросы. Чтобы разрешить это, создайте политику в Apache Ranger для ресурса `queryId` со значением `*` и разрешением `execute` для пользователя `{USER}`.
- Пользователи должны иметь разрешение на имперсонацию самих себя в Trino. Без политики в Apache Ranger, предоставляющей это разрешение, пользователи не смогут выполнять какие-либо запросы. Чтобы разрешить это, создайте политику в Apache Ranger для ресурса `trinouser` со значением `{USER}` и с разрешением `impersonate` для пользователя `{USER}`.

Соответствие ресурсов и типов доступа между Trino и Hive

Соответствие ресурсов выглядит следующим образом:

- `{schema}` → `{database}`
- `{schema}.{table}` → `{database}.{table}`
- `{catalog}.{schema}.{table}.{column}` → `{database}.{table}.{column}`

Типы доступа к базе данных

Тип доступа в Trino	Тип доступа в Hive
all	all
_any	use
create	create
drop	drop
alter	alter
select	use
show	use
use	use

Типы доступа к таблице

Тип доступа в Trino	Тип доступа в Hive
all	all
_any	use
create	create
drop	drop
alter	alter
insert	update
delete	update
select	select
show	use
use	use

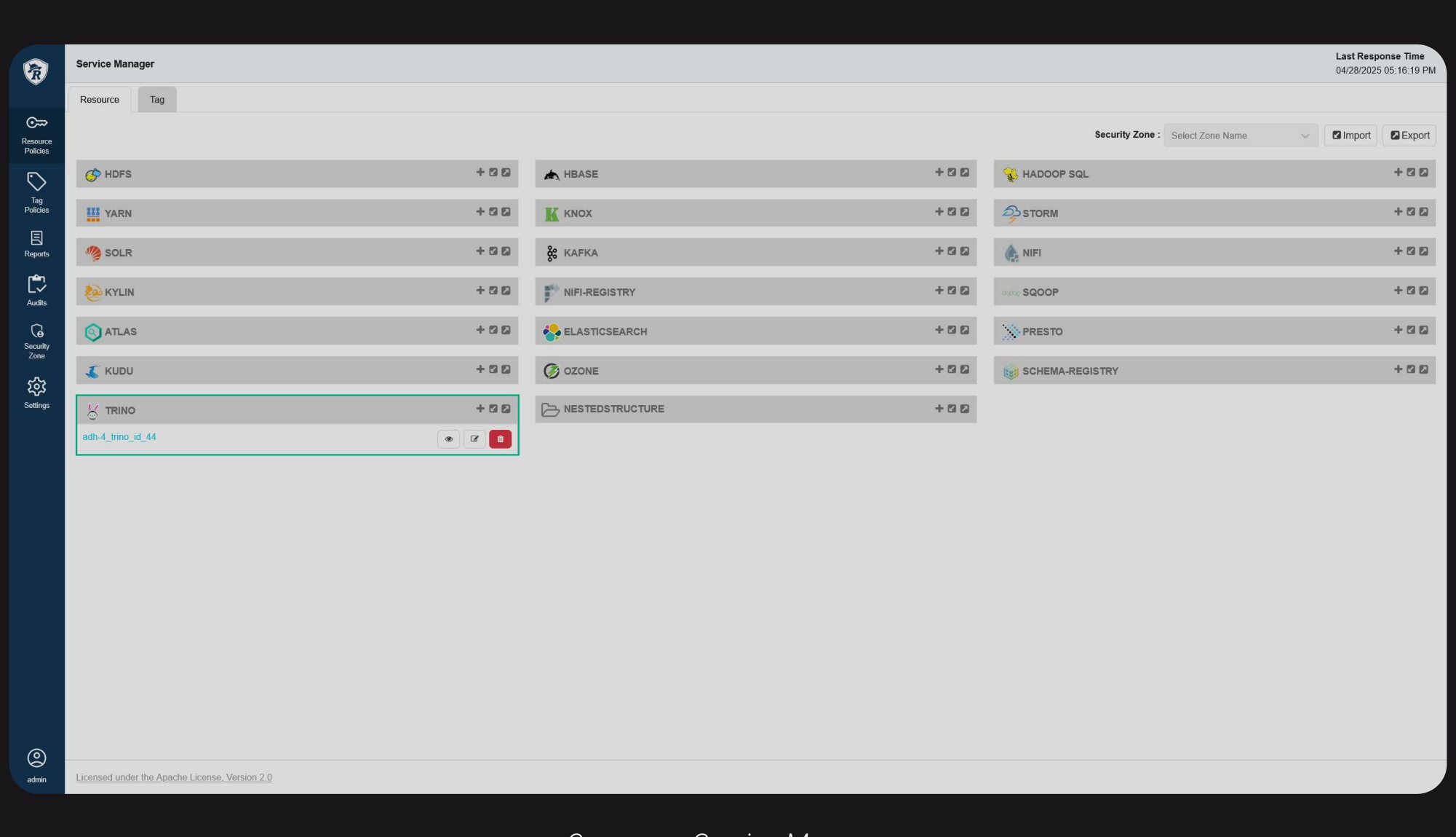
Типы доступа к столбцу

Тип доступа в Trino	Тип доступа в Hive
all	all
_any	use
select	select

Добавление новой политики в Ranger

Чтобы добавить новую политику к существующему сервису Trino, необходимо выполнить следующие действия.

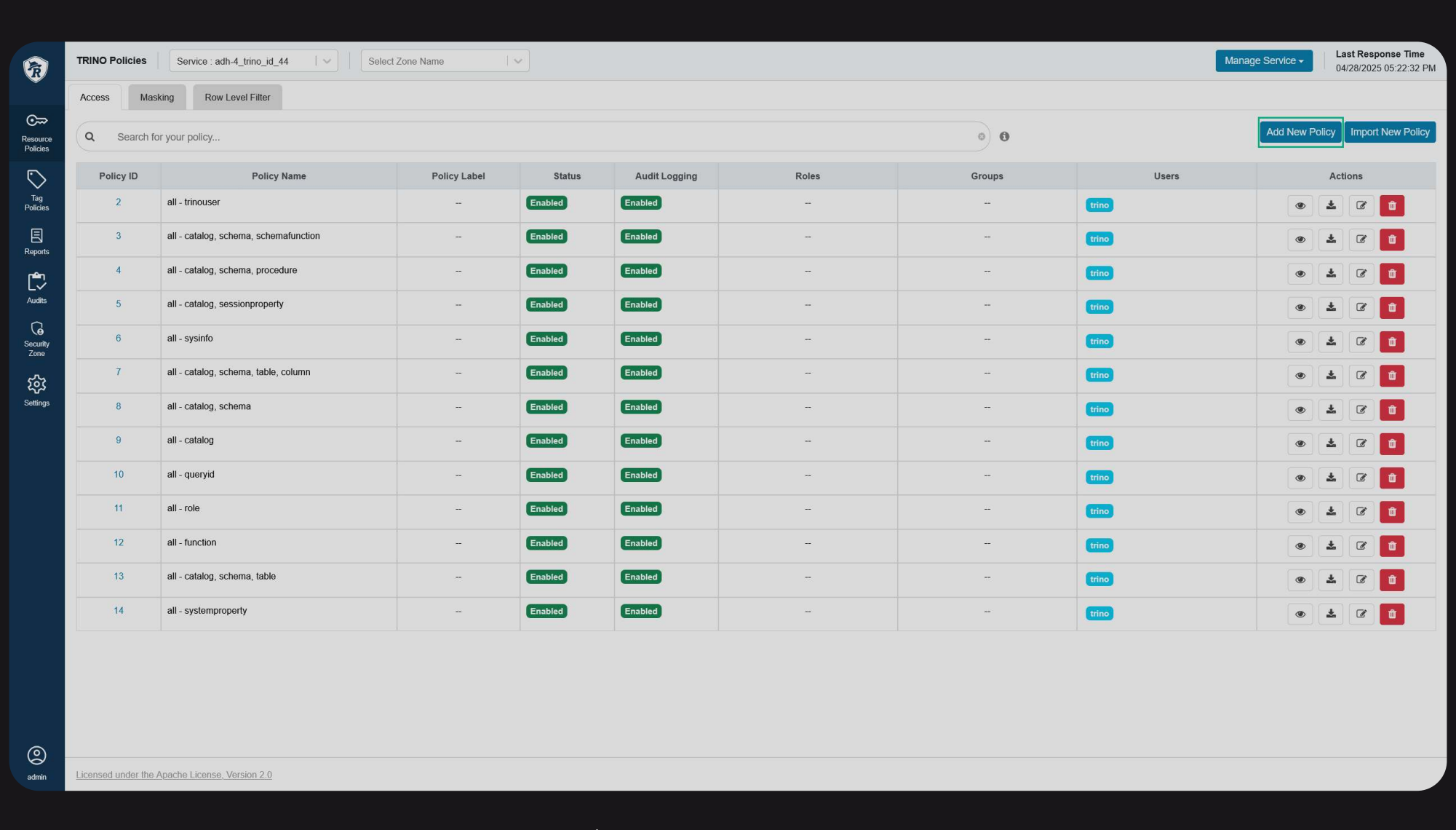
1. На странице ***Service Manager*** нажмите на существующий сервис Trino в панели Trino.



Service Manager
Resource
Tag
Security Zone : Select Zone Name
Import
Export
HDFS
YARN
SOLR
KYLIN
ATLAS
KUDU
TRINO
adh-4_trino_id_44
HBASE
KNOX
KAFKA
NIFI-REGISTRY
ELASTICSEARCH
OZONE
NESTEDSTRUCTURE
HADOOP SQL
STORM
NIFI
SQOOP
PRESTO
SCHEMA-REGISTRY
Last Response Time 04/28/2025 05:16:19 PM
Licensed under the Apache License, Version 2.0

Страница Service Manager

2. На странице политик Trino нажмите ***Add New Policy***.



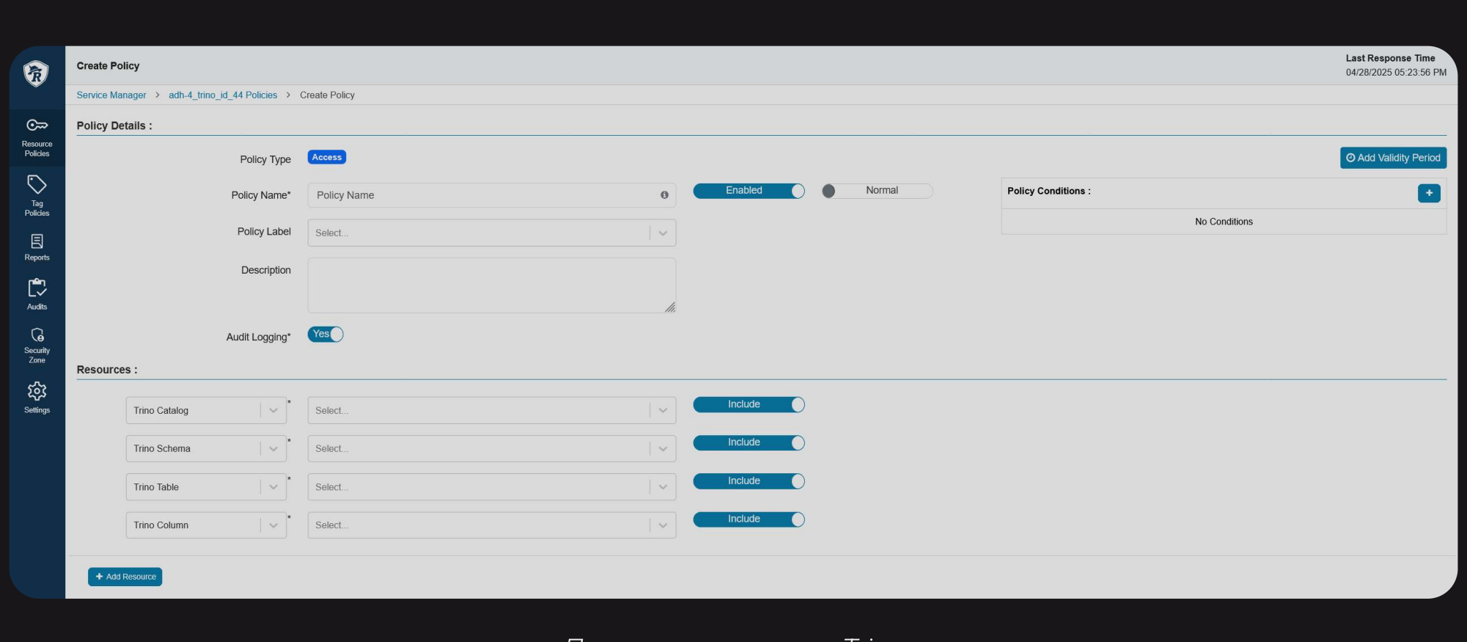
TRINO Policies — Service : adh-4_trino_id_44

Policy ID	Policy Name	Policy Label	Status	Audit Logging	Roles	Groups	Users	Actions
2	all - trinouser	--	Enabled	Enabled	--	--	trino	
3	all - catalog, schema, schemafunction	--	Enabled	Enabled	--	--	trino	
4	all - catalog, schema, procedure	--	Enabled	Enabled	--	--	trino	
5	all - catalog, sessionproperty	--	Enabled	Enabled	--	--	trino	
6	all - sysinfo	--	Enabled	Enabled	--	--	trino	
7	all - catalog, schema, table, column	--	Enabled	Enabled	--	--	trino	
8	all - catalog, schema	--	Enabled	Enabled	--	--	trino	
9	all - catalog	--	Enabled	Enabled	--	--	trino	
10	all - queryid	--	Enabled	Enabled	--	--	trino	
11	all - role	--	Enabled	Enabled	--	--	trino	
12	all - function	--	Enabled	Enabled	--	--	trino	
13	all - catalog, schema, table	--	Enabled	Enabled	--	--	trino	
14	all - systemproperty	--	Enabled	Enabled	--	--	trino	

Select Zone Name
Manage Service
Access
Masking
Row Level Filter
Search for your policy...
Add New Policy
Import New Policy
Last Response Time 04/28/2025 05:22:32 PM

Добавление новой политики

3. На открывшейся странице ***Create Policy*** заполните необходимые поля.



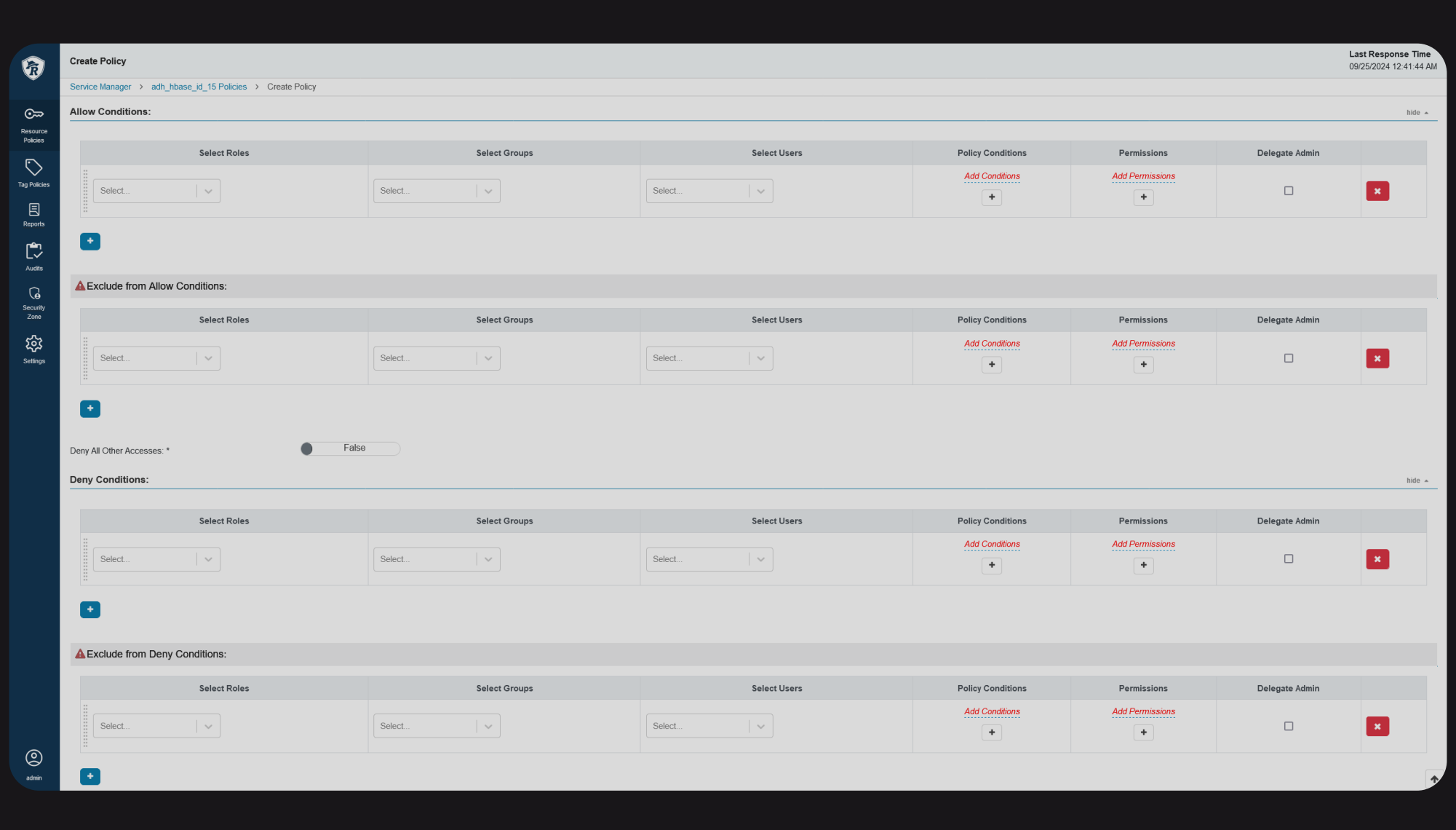
Create Policy
Service Manager > adh-4_trino_id_44 Policies > Create Policy
Policy Details :
Policy Type
Access
Policy Name*
Policy Label
Description
Audit Logging*
Yes
Enabled
Normal
Add Validity Period
Policy Conditions :
No Conditions
Resources :
Trino Catalog
Trino Schema
Trino Table
Trino Column
Select...
Include
Add Resource
Last Response Time 04/28/2025 05:23:56 PM

Параметры политики Trino

Параметры политики

Параметр	Описание
Policy Name	Название политики. Должно быть уникальным в системе
Enabled	Флаг, позволяющий применить политику сразу после создания
Normal/Override	Позволяет применить override-политику. Если выбран режим ***override***, то права доступа новой политики переписывают права доступа существующих политик
Policy Label	Позволяет группировать несколько политик с метками, что дает возможность искать политики по меткам. Поиском можно воспользоваться на страницах ***Policy listing*** и ***Reports***. Также упрощает экспорт/импорт политик — если требуется экспортировать некоторые политики, то можно провести поиск по меткам и экспортировать результат
Description	Описание назначения политик
Audit Logging	Включает аудит для политики
Add Validity Period	Позволяет указать время начала и окончания действия политики
Trino Catalog	Название каталога в Trino
Trino Schema	Название схемы в Trino
Trino Table	Название таблицы в Trino
Trino Column	Название колонки в Trino
Trino User	Имя пользователя Trino
System Property	Название свойства системы
Trino Function	Название функции в Trino
Query ID	Идентификатор запроса
System Information	Информация о среде кластера Trino
Role	Название роли в Trino

4. Установите параметры разрешений/запретов. Секция ***Allow Conditions*** позволяет разрешать доступ определенным ролям/группам/пользователям. Имеет смысл использовать эту секцию, когда требуется разрешить доступ небольшому числу пользователей, а остальным — запретить. Секция ***Deny Conditions*** позволяет запрещать доступ определенным ролям/группам/пользователями. Её удобно использовать, когда требуется запретить доступ небольшому числу пользователей, а остальным — разрешить. Также можно комбинировать правила из обеих секций. Для добавления дополнительных условий кликните +. Условия интерпретируются в порядке, указанном в политике. Сначала применяется условие, находящееся в начале списка, затем второе, затем третье и так далее.



Create Policy
Service Manager > adh_hbase_id_15 Policies > Create Policy
Allow Conditions:
Exclude from Allow Conditions:
Deny All Other Accesses: *
False
Deny Conditions:
Exclude from Deny Conditions:
Select Roles
Select Groups
Select Users
Policy Conditions
Permissions
Delegate Admin
Add Conditions
Add Permissions
Select...
Last Response Time 09/25/2024 12:41:44 AM

Параметры доступа к политике Trino

Параметры Allow Conditions

Параметр	Описание
Select Role	Укажите роли, к которым применяется данная политика
Select Group	Укажите группы, к которым применяется эта политика. Публичная группа содержит всех пользователей, поэтому предоставление доступа к публичной группе дает доступ всем пользователям
Select User	Укажите пользователей, к которым применяется данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

Параметры Deny Conditions

Параметр	Описание
Select Role	Укажите роли, к которым не будет применяться данная политика
Select Group	Укажите группы, к которым не будет применяться эта политика. Публичная группа содержит всех пользователей, поэтому запрет доступа к публичной группе запрещает доступ всем пользователям
Select User	Укажите пользователей, к которым не будет применяться данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

5. Кликните ***Add*** внизу страницы.

Плагин Ranger YARN

Сергей Тихомиров

Содержание

Назад к содержанию

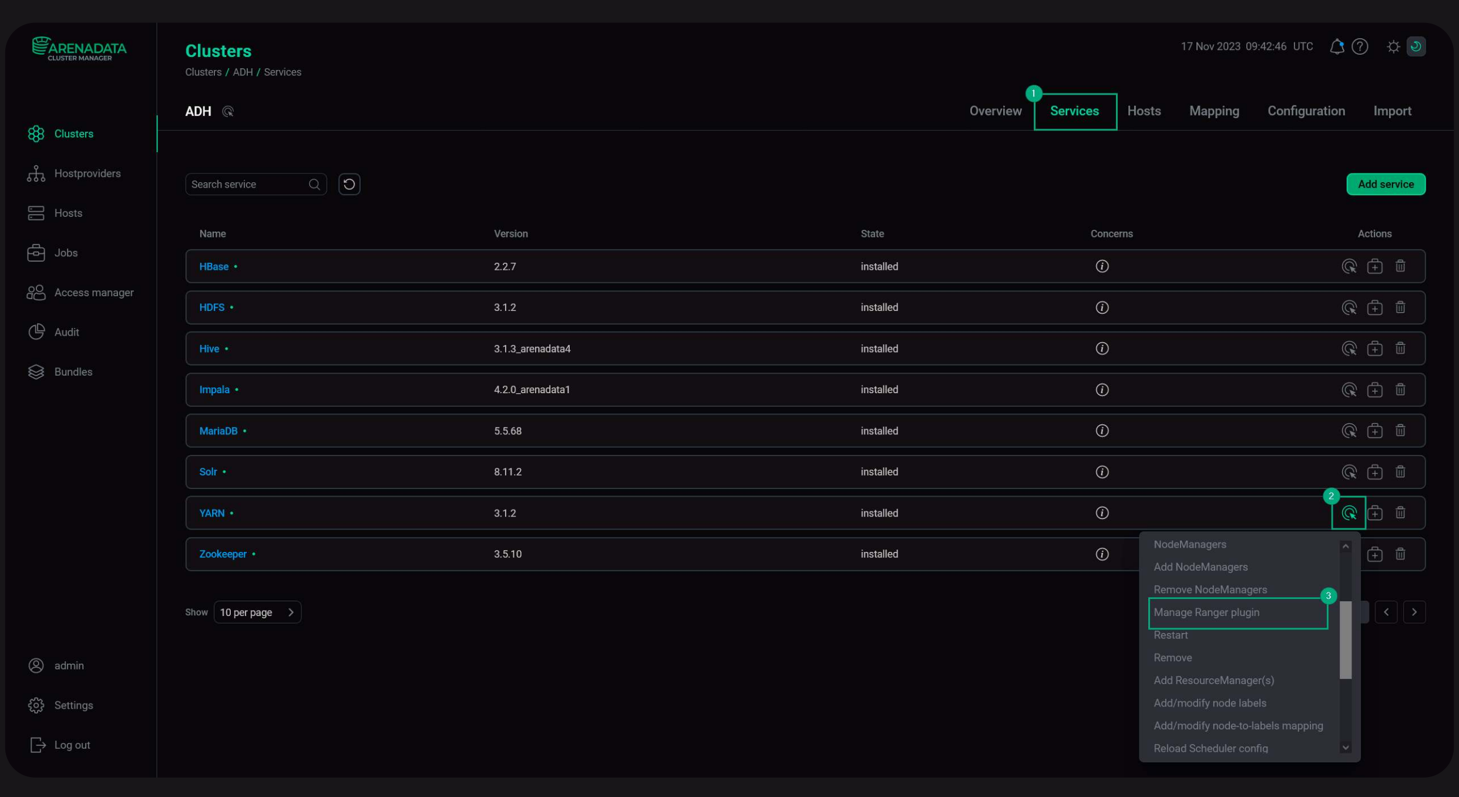
Включение плагина YARN

Добавление новой политики в Ranger

Включение плагина YARN

Чтобы включить плагин Ranger для YARN, выполните следующие действия.

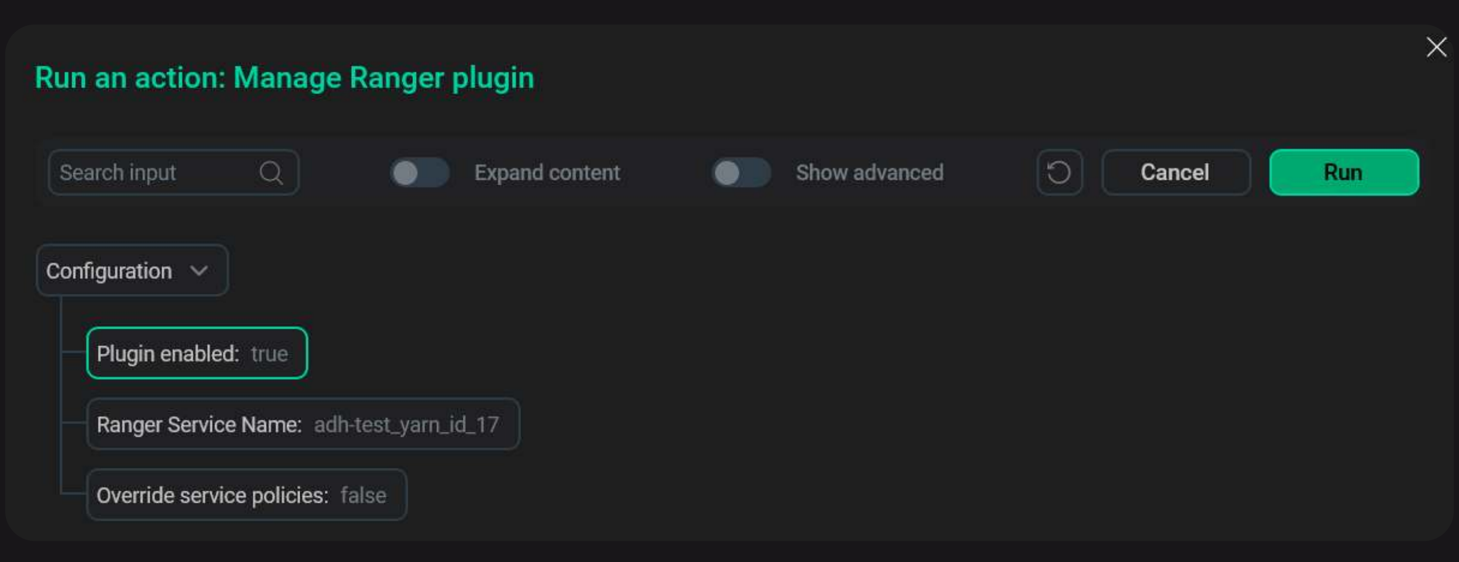
1. Перейдите на страницу ***Clusters*** → ***<кластер_ADH>*** → ***Services***.
2. Кликните на иконку в строке YARN и выберите действие ***Manage Ranger plugin*** в выпадающем меню.



Name	Version	State
HBase	2.2.7	installed
HDFS	3.1.2	installed
Hive	3.1.3_arenadata4	installed
Impala	4.2.0_arenadata1	installed
MariaDB	5.5.68	installed
Solr	8.11.2	installed
YARN	3.1.2	installed
Zookeeper	3.5.10	installed

Сервисы кластера

3. Выберите требуемое состояние флага ***Plugin enabled*** (`true` — включенный плагин, `false` — выключенный). Также здесь можно установить название сервиса Ranger, который будет добавлен. Если сервис с таким именем уже существует, его можно пересоздать, активировав параметр ***Override service policies*** (`true` — параметр активен, `false` — неактивен) — в таком случае старый сервис будет удален, а для нового сервиса будут заново сгенерированы политики.



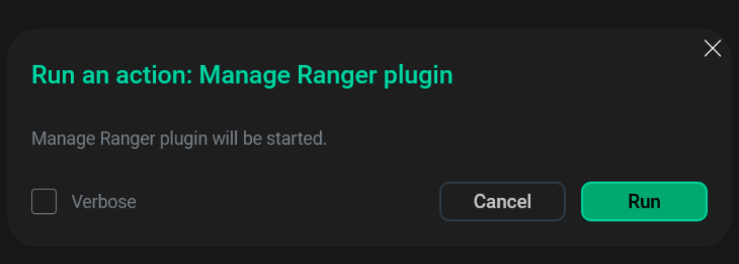
Run an action: Manage Ranger plugin

Configuration
- Plugin enabled: true
- Ranger Service Name: adh-test_yarn_id_17
- Override service policies: false

Состояние плагина

> **ПРИМЕЧАНИЕ**
>
> При первом включении плагина сервис с политиками будет создан (если еще не существует) вне зависимости от значения параметра ***Override service policies***.

4. Кликните ***Run*** и подтвердите действие в открывшемся окне.



Run an action: Manage Ranger plugin

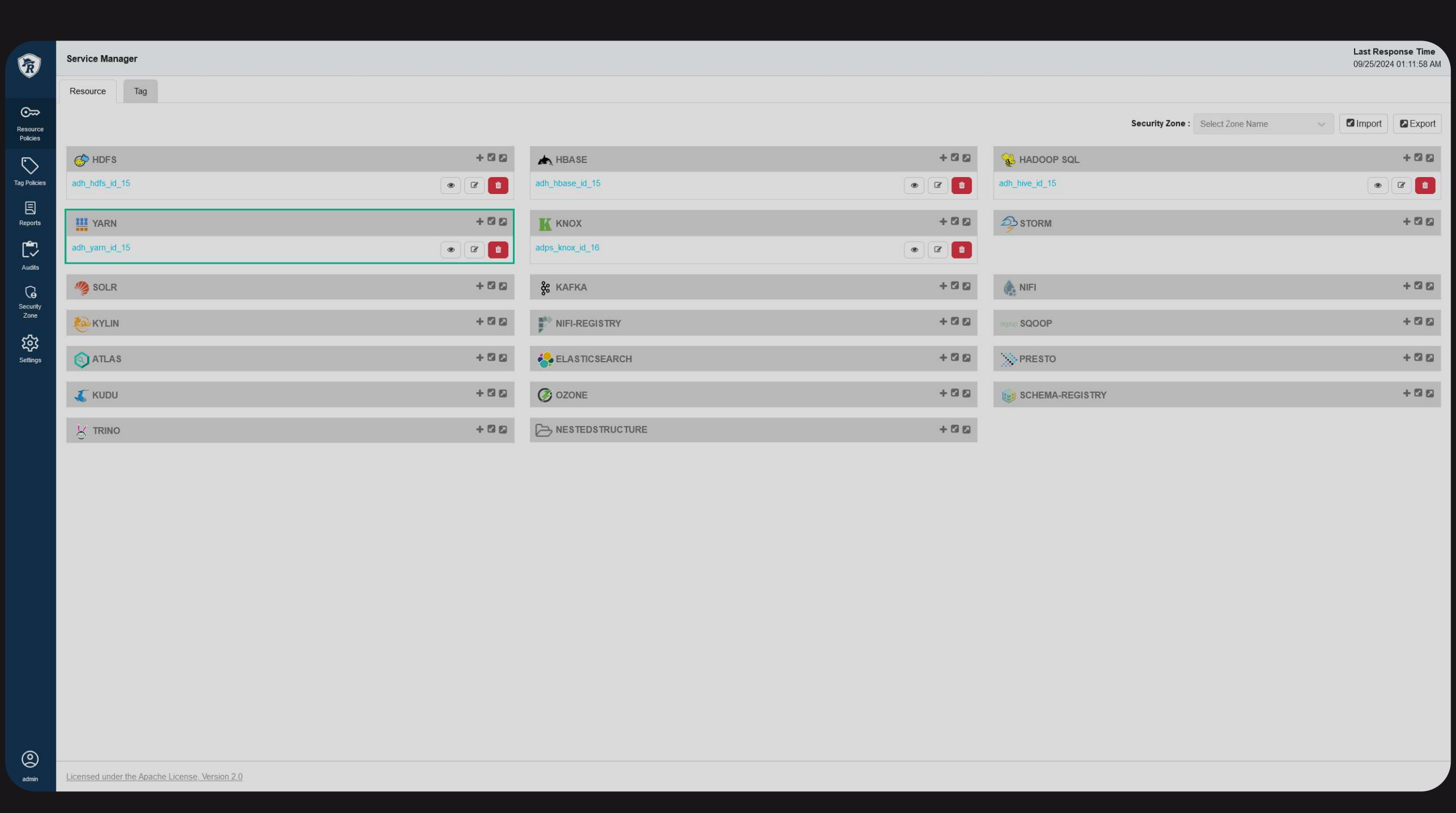
Manage Ranger plugin will be started.

Подтверждение действия

Добавление новой политики в Ranger

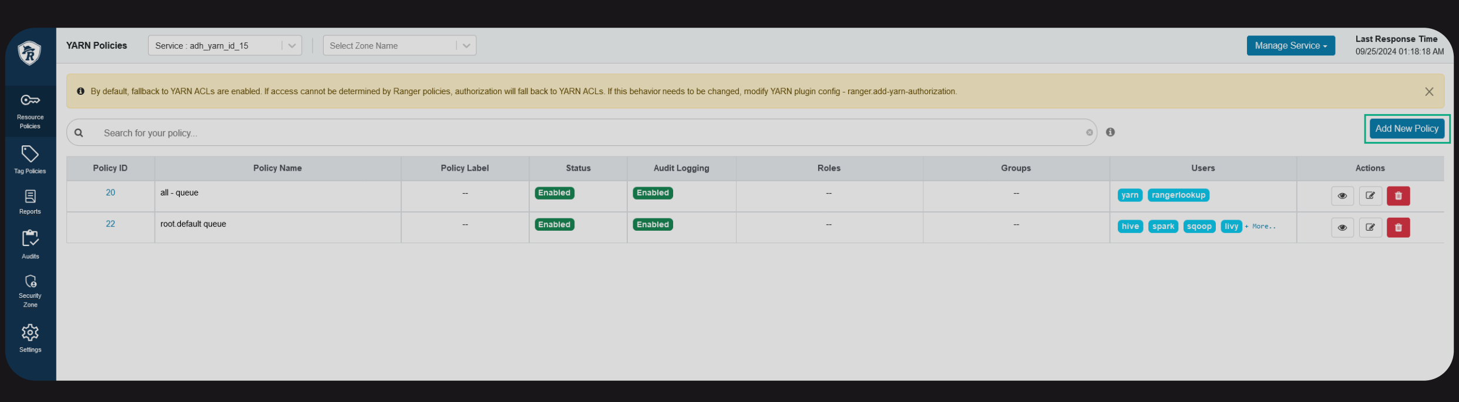
Чтобы добавить новую политику к существующему сервису YARN, необходимо выполнить следующие действия:

1. На странице ***Service Manager*** нажмите на существующий сервис YARN в панели Hadoop SQL.



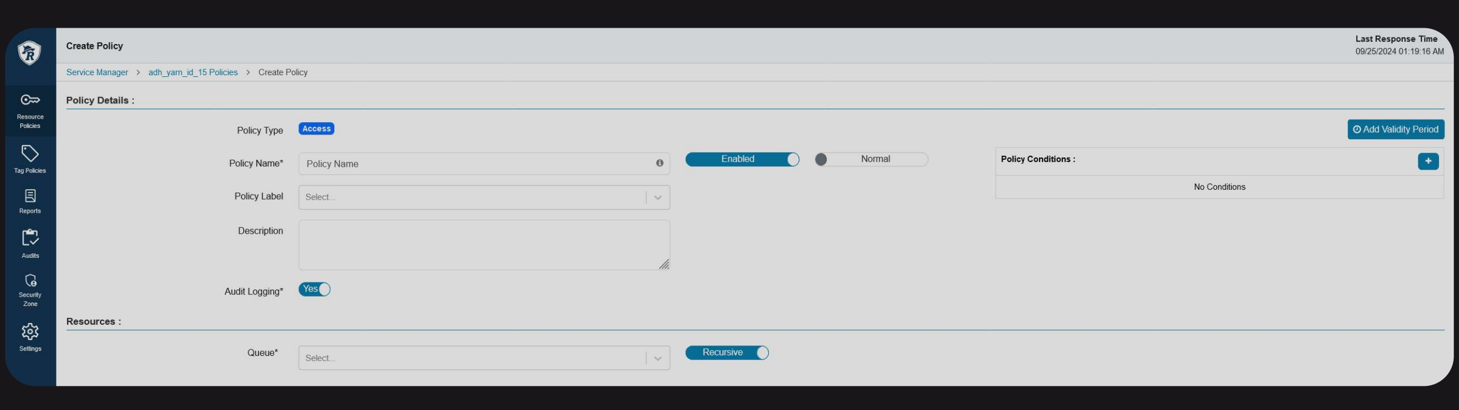
Страница Service Manager

2. На странице политик YARN нажмите ***Add New Policy***.



Добавление новой политики

3. На открывшейся странице ***Create Policy*** заполните необходимые поля.

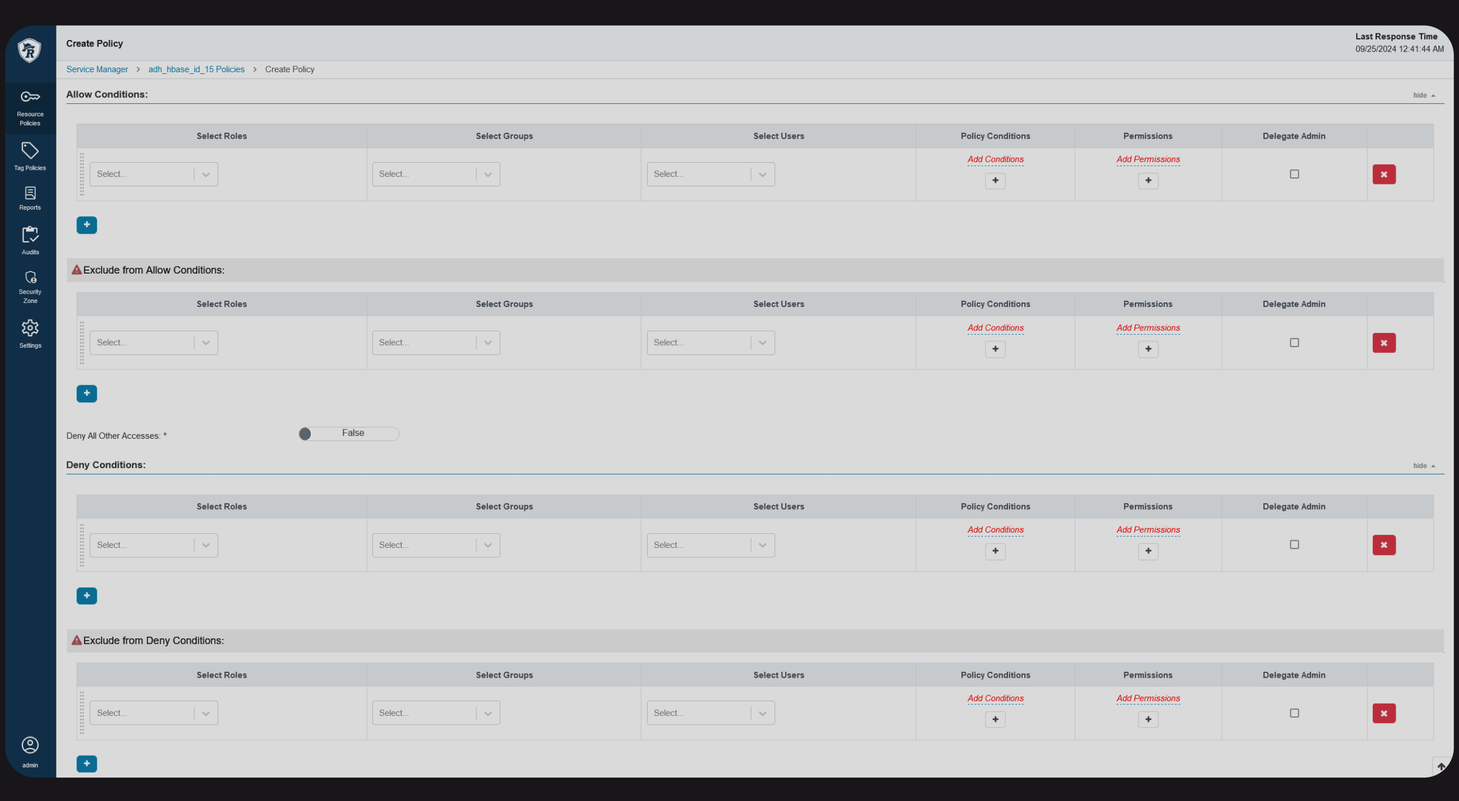


Параметры политики YARN

Параметры политики

Параметр	Описание
Policy Name	Название политики. Должно быть уникальным в системе
Enabled	Флаг, позволяющий применить политику сразу после создания
Normal/Override	Позволяет применить override-политику. Если выбран режим ***override***, то права доступа новой политики переписывают права доступа существующих политик
Policy Label	Позволяет группировать несколько политик с метками, что дает возможность искать политики по меткам. Поиском можно воспользоваться на страницах ***Policy listing*** и ***Reports***. Также упрощает экспорт/импорт политик — если требуется экспортировать некоторые политики, то можно провести поиск по меткам и экспортировать результат
Queue	Фундаментальная единица планирования в YARN
Recursive/Non-recursive	Указывает, подпадают ли под действие политики все файлы и папки в существующей папке. Может использоваться вместо символов подстановки
Description	Описание назначения политик
Audit Logging	Включает аудит для политики
Add Validity Period	Позволяет указать время начала и окончания действия политики

4. Установите параметры разрешений/запретов. Секция ***Allow Conditions*** позволяет разрешать доступ определенным ролям/группам/пользователям. Имеет смысл использовать эту секцию, когда требуется разрешить доступ небольшому числу пользователей, а остальным — запретить. Секция ***Deny Conditions*** позволяет запрещать доступ определенным ролям/группам/пользователями. Её удобно использовать, когда требуется запретить доступ небольшому числу пользователей, а остальным — разрешить. Также можно комбинировать правила из обеих секций. Для добавления дополнительных условий кликните +. Условия интерпретируются в порядке, указанном в политике. Сначала применяется условие, находящееся в начале списка, затем второе, затем третье и так далее.



Параметры доступа к политике YARN

Параметры Allow Conditions

Параметр	Описание
Select Role	Укажите роли, к которым применяется данная политика
Select Group	Укажите группы, к которым применяется эта политика. Публичная группа содержит всех пользователей, поэтому предоставление доступа к публичной группе дает доступ всем пользователям
Select User	Укажите пользователей, к которым применяется данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

Параметры Deny Conditions

Параметр	Описание
Select Role	Укажите роли, к которым не будет применяться данная политика
Select Group	Укажите группы, к которым не будет применяться эта политика. Публичная группа содержит всех пользователей, поэтому запрет доступа к публичной группе запрещает доступ всем пользователям
Select User	Укажите пользователей, к которым не будет применяться данная политика
Permissions	Добавьте или измените разрешения
Delegate Admin	Используйте эту опцию, чтобы назначить привилегии администратора пользователям или группам, указанным в политике. Администраторы могут редактировать или удалять политику, а также создавать дочерние политики

5. Кликните ***Add*** внизу страницы.

Обзор Kerberos

Константин Алпашкин

Содержание

Назад к содержанию

Общие требования

Настройка сопоставления групп Hadoop для LDAP/AD

Kerberos может быть настроен в кластере с использованием следующих типов KDC:

- MIT KDC, который состоит из базы данных принципалов и хранилища ключей Kerberos.
- Active Directory (AD), состоящий из базы данных принципалов и хранилища ключей Windows Server.
- FreeIPA — бесплатная система управления идентификацией пользователей для Linux/UNIX с открытым исходным кодом.
- Samba — ПО, которое можно использовать как контроллер домена Active Directory.

Процесс идентификации и аутентификации по любой из схем аналогичен. Единственное различие заключается в типе используемого KDC.

Общие требования

В данном разделе описаны требования, необходимые для внедрения протокола Kerberos в кластер ADH.

Чтобы внедрить Kerberos в существующий кластер ADH, необходимо выполнить следующие условия:

- На всех хостах должны быть отключены IPTables и SELinux.
- Кластер ADH установлен и настроен. Включить Kerberos можно только тогда, когда кластер готов к работе.
- DNS-сервис настроен и доступен, включая конфигурацию обратной DNS-зоны.
- Существует учетная запись администратора Active Directory или MIT Kerberos, которая имеет привилегии для создания, удаления и управления учетными записями пользователей.
- Хранилище ключей и пользователей Active Directory или MIT Kerberos настроено и готово.

Настройка сопоставления групп Hadoop для LDAP/AD

В данном разделе описаны требования, необходимые для настройки Active Directory для Kerberos.

Чтобы обеспечить принудительную авторизацию на уровне групп LDAP/AD в Hadoop, необходимо настроить сопоставление групп Hadoop для LDAP/AD.

> **РЕКОМЕНДАЦИЯ**
>
> Настройки LDAP могут отличаться в зависимости от используемой реализации LDAP.

Существует два способа настройки сопоставления групп Hadoop.

- Настройка сопоставления групп Hadoop для LDAP/AD с помощью SSSD (рекомендуемый способ).

  Для сопоставления групп мы рекомендуем использовать SSSD или одну из следующих служб для соединения Linux и LDAP:

  - Centrify;
  - NSLCD;
  - Winbind;
  - SAMBA.

  Большинство из перечисленных служб позволяют выполнять поиск пользователей, составлять список групп и выполнять другие действия на хосте. Однако ни одно из этих действий не требуется для отображения групп LDAP в Hadoop. Поэтому при оценке этих служб необходимо понимать разницу между модулем NSS (выполняет авторизацию пользователей/групп) и модулем PAM (выполняет аутентификацию пользователей). NSS необходим для поиска или проверки пользователя в LDAP и перечисления групп. PAM также может представлять угрозу безопасности.

- Настройка сопоставления групп через *core-site.xml*.

  Настройка использования отображения групп на основе LDAP в файле *core-site.xml* выполняется в следующем порядке:

  - Добавьте свойства, показанные в примере ниже, в файл *core-site.xml*. Необходимо указать значение для связанного пользователя, его пароль и другие свойства, специфичные для экземпляра LDAP. Убедитесь, что фильтры классов объектов, пользователей и групп соответствуют значениям, указанным в экземпляре LDAP.

Настройка параметров аутентификации через core-site.xml

```
<property>
<name>hadoop.security.group.mapping</name>
<value>org.apache.hadoop.security.LdapGroupsMapping</value>
</property>

<property>
<name>hadoop.security.group.mapping.ldap.bind.user</name>
<value>cn=Manager,dc=hadoop,dc=apache,dc=org</value>
</property>

<!-
<property>
<name>hadoop.security.group.mapping.ldap.bind.password.file</name>
<value>/etc/hadoop/conf/ldap-conn-pass.txt</value>
</property>
->

<property>
<name>hadoop.security.group.mapping.ldap.bind.password</name>
<value>hadoop</value>
</property>

<property>
<name>hadoop.security.group.mapping.ldap.url</name>
<value>ldap://localhost:389/dc=hadoop,dc=apache,dc=org</value>
</property>

<property>
<name>hadoop.security.group.mapping.ldap.url</name>
<value>ldap://localhost:389/dc=hadoop,dc=apache,dc=org</value>
</property>

<property>
<name>hadoop.security.group.mapping.ldap.base</name>
<value></value>
</property>

<property>
<name>hadoop.security.group.mapping.ldap.search.filter.user</name>
<value>(&amp;(|(objectclass=person)(objectclass=applicationProcess))(cn={0}))</value>
</property>

<property>
<name>hadoop.security.group.mapping.ldap.search.filter.group</name>
<value>(objectclass=groupOfNames)</value>
</property>

<property>
<name>hadoop.security.group.mapping.ldap.search.attr.member</name>
<value>member</value>
</property>

<property>
<name>hadoop.security.group.mapping.ldap.search.attr.group.name</name>
<value>cn</value>
</property>
```

  - В зависимости от конфигурации, вы можете обновить сопоставления пользователей и групп с помощью следующих команд HDFS и YARN:

```
$ hdfs dfsadmin -refreshUserToGroupsMappings
$ yarn rmadmin -refreshUserToGroupsMappings
```

  - Проверьте сопоставление групп LDAP, выполнив команду `hdfs groups`. Команда отображает группы из LDAP для текущего пользователя. При настроенном отображении групп LDAP разрешения HDFS могут использовать группы, определенные LDAP, для контроля доступа.

Настройка сервера MIT Kerberos

Константин Алпашкин

Назад к содержанию

ADH-кластер, где необходимо настроить Kerberos, должен быть полностью подготовлен и настроен. Конфигурация может быть любой.

Прежде чем приступить к настройке Kerberos в кластере, схема (scheme) должна содержать базу данных принципалов MIT KDC и базу данных пользователей ADPS. Логины для всех пользователей и служб должны совпадать в обеих базах данных.

После керберизации кластера у каждого пользователя и службы будет соответствующий принципал.

В дополнение к MIT Kerberos схема (scheme) использует ADPS для управления разрешениями пользователей. MIT Kerberos управляет аутентификацией принципалов, а ADPS — разрешениями принципалов.

База данных принципалов MIT Kerberos должна быть синхронизирована с базой данных пользователей ADPS. Эта синхронизация выполняется вручную перед настройкой Kerberos в кластере. Каждое последующее изменение в базе данных принципалов MIT Kerberos должно быть продублировано в базе данных пользователей ADPS.

В этом случае при работе с кластером ADH все принципалы получают TGT от MIT Kerberos KDC.

Настройки сервера Kerberos

Константин Алпашкин

Содержание

Назад к содержанию

Основная настройка сервера Kerberos

Настройка вторичного сервера Kerberos для обеспечения высокой доступности

Основная настройка сервера Kerberos

РЕКОМЕНДАЦИЯ

Приведенная ниже инструкция может быть использована также для замены основного и дополнительного KDC.

1. Установите Kerberos на сервер, выполнив команду:

```
$ sudo yum install -y krb5-server pam_krb5 krb5-workstation xinetd
```

2. Отредактируйте и внесите в файл */etc/krb5.conf* следующие изменения:

krb5.conf

```
# Configuration snippets may be placed in this directory as well
includedir /etc/krb5.conf.d/
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log
[libdefaults]
dns_lookup_realm = false
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true
rdns = false
pkinit_anchors = FILE:/etc/pki/tls/certs/ca-bundle.crt
default_realm = ADREALM.IO
default_ccache_name = KEYRING:persistent:%{uid}
[realms]
ADREALM.IO = {
kdc = kerberos-primary-host.ru-central1.internal #primary KDC host
kdc = kerberos-secondary-host.ru-central1.internal #secondary KDC host
admin_server = kerberos-primary-host.ru-central1.internal
}
[domain_realm]
.adrealm.io = ADREALM.IO
adrealm.io  = ADREALM.IO
```

3. Отредактируйте и внесите в файл */var/kerberos/krb5kdc/kdc.conf* следующие изменения:

kdc.conf

```
[kdcdefaults]
kdc_ports = 88
kdc_tcp_ports = 88
[realms]
ADREALM.IO = {
#master_key_type = aes256-cts
acl_file = /var/kerberos/krb5kdc/kadm5.acl
dict_file = /usr/share/dict/words
admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal arcfour-hmac:normal camellia256-cts:normal
camellia128-cts:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal
}
```

4. Отредактируйте и внесите в файл */var/keberos/krb5kdc/kadm5.acl* следующие изменения:

kadm5.acl

```
*/admin@ADREALM.IO *
```

5. Используйте утилиту `kdb5_util` для настройки базы данных Kerberos.

```
$ sudo kdb5_util create -s -r ADREALM.IO
```

6. Запомните пароль Master KDC и запустите сервис, выполнив команды:

```
$ sudo systemctl enable kadmin
$ sudo systemctl enable krb5kdc
$ sudo systemctl start kadmin
$ sudo systemctl start krb5kdc
```

7. Создайте новые хост-принципалы для каждого экземпляра KDC. Используя консоль администратора Kerberos, выполните команды:

```
$ sudo kadmin.local
$ addprinc root/admin
$ addprinc -randkey host/kerberos-primary-host.ru-central1.internal
$ addprinc -randkey host/kerberos-secondary-host.ru-central1.internal
$ ktadd host/kerberos-primary-host.ru-central1.internal
```

8. На главном сервере Kerberos сгенерируйте файл *keytab*, который будет использоваться для вторичного сервера Kerberos.

```
$ ktadd -k /tmp/kerberos-secondary-host.keytab host/kerberos-secondary-host.ru-central1.internal
```

Настройка вторичного сервера Kerberos для обеспечения высокой доступности

1. Установите необходимые пакеты на второй сервер:

```
$ sudo yum install -y krb5-server krb5-workstation xinetd
```

2. В файле */etc/ssh/sshd_config* раскомментируйте следующую строку для включения аутентификации по паролю.

```
PasswordAuthentication yes
```

3. Перезапустите сервис:

```
$ sudo systemctl restart sshd
```

4. Скопируйте файл *keytab* вторичного сервера, который был создан ранее, на вторичный узел KDC.

```
$ sudo scp /tmp/kerberos-secondary-host.keytab <user>@<ip address>:/tmp/
```

5. Находясь на вторичном сервере, переместите файл *keytab* в каталог */etc/* и переименуйте его.

```
$ sudo cp /tmp/kerberos-secondary-host.keytab /etc/krb5.keytab
```

6. База данных принципалов передается от главного KDC к репликам KDC с помощью демона *kpropd*. Необходимо указать принципалов, которым разрешено получать обновления дампов вторичного сервера Kerberos. Эти обновления содержат новую базу данных.
   Создайте файл *kpropd.acl* в каталоге мониторинга состояния KDC. Файл *kpropd.acl* содержит имена хостов принципалов для каждого KDC.

```
$ sudo vi /var/kerberos/krb5kdc/kpropd.acl
```

7. Добавьте следующие строки в файл *kpropd.acl*:

```
host/kerberos-primary-host.ru-central1.internal@ADREALM.IO
host/kerberos-secondary-host.ru-central1.internal@ADREALM.IO
```

ПРИМЕЧАНИЕ

Если предполагается, что первичный и вторичный KDC могут поменяться местами, необходимо указать адреса хостов всех серверов KDC в файле *kpropd.acl* на каждом KDC. В противном случае в файле *kpropd.acl* на репликах KDC необходимо указать только основной хост.

8. Скопируйте следующие файлы с первичного узла KDC на вторичный узел KDC.
   - *krb5.conf*
   - *kdc.conf*
   - *kadm5.acl*
   - master key stash file

9. Переместите файлы, которые были скопированы, в соответствующие каталоги. Структура файлов и каталогов должна быть точно такой же, как на основном KDC.

РЕКОМЕНДАЦИЯ

Файл *kadm5.acl* нужен для того, чтобы реплика могла взаимодействовать с основным KDC.

10. На основном сервере выполните последовательно следующие команды:

```
$ sudo scp /etc/krb5.conf <user>@<ip address>:/tmp/
$ sudo scp /var/kerberos/krb5kdc/kdc.conf <user>@<ip address>:/tmp/
$ sudo scp /var/kerberos/krb5kdc/kadm5.acl <user>@<ip address>:/tmp/
$ sudo scp /var/kerberos/krb5kdc/.k5.ADREALM.IO <user>@<ip address>:/tmp/
$ sudo scp /var/kerberos/krb5kdc/kpropd.acl <user>@<ip address>:/tmp/
```

11. На вторичном сервере выполните последовательно следующие команды:

```
$ sudo cp /tmp/krb5.conf /etc/
$ sudo cp /tmp/kdc.conf /var/kerberos/krb5kdc/
$ sudo cp /tmp/kadm5.acl /var/kerberos/krb5kdc/
$ sudo cp /tmp/.k5.ADREALM.IO /var/kerberos/krb5kdc/
$ sudo cp /tmp/kpropd.acl /var/kerberos/krb5kdc/
```

12. Создайте конфигурационный файл */etc/xinetd.d/krb5_prop*.

/etc/xinetd.d/krb5_prop

```
service krb5_prop

{
socket_type = stream
wait = no
user = root
server = /usr/sbin/kpropd
only_from = 0.0.0.0/0
log_on_success = PID HOST EXIT DURATION
}
```

13. На каждом KDC добавьте следующую строку в файл */etc/services*:

```
krb5_prop       754/tcp               # Kerberos replica propagation
```

14. Выполните следующую команду:

```
$ sudo systemctl restart xinetd
```

15. Перед копированием базы данных создайте файл дампа базы данных на первичном KDC.

```
$ sudo kdb5_util dump /var/kerberos/krb5kdc/slavedump
```

16. Скопируйте базу данных в каждую реплику KDC:

```
$ sudo kprop -f /var/kerberos/krb5kdc/slavedump kerberos-secondary-host.ru-central1.internal
```

```
Database propagation to kerberos-secondary-host.ru-central.internal: SUCCEEDED
```

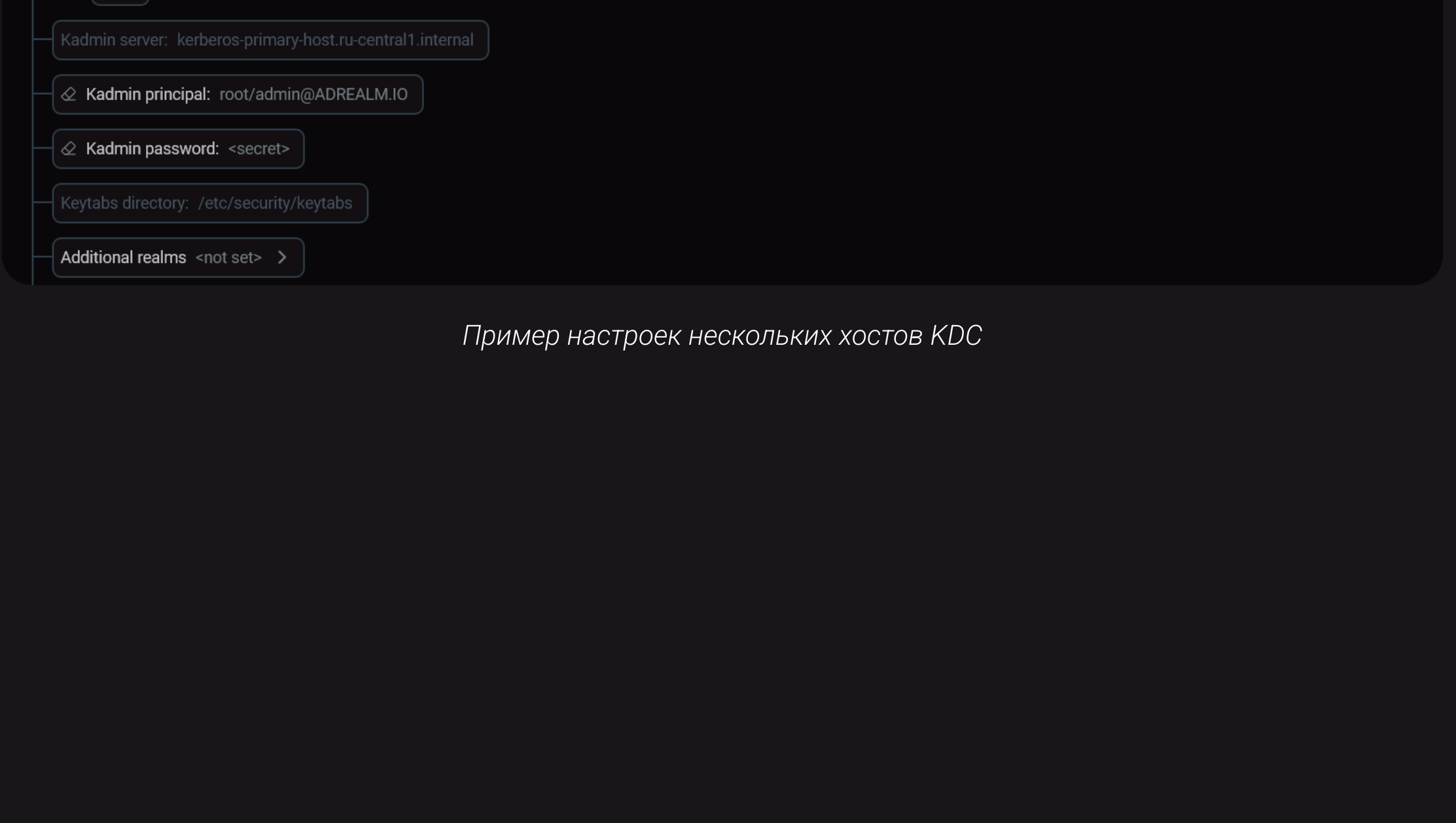
17. Запустите KDC на вторичном узле:

```
$ sudo systemctl enable krb5kdc
$ sudo systemctl start krb5kdc
```

18. Войдите в консоль `admin.local` и выполните команду `listprincs` для просмотра списка принципалов, чтобы убедиться, что список совпадает со списком на основном хосте.

Чтобы реализовать регулярную синхронизацию базы данных принципалов, необходимо создать cron-задачу. Пример такого скрипта доступен ниже:

```
#!/bin/sh

kdclist = "kerberos-primary-host.ru-central1.internal kerberos-secondary-host.ru-central1.internal"
kdb5_util dump /var/kerberos/krb5kdc/slavedump
for kdc in $kdclist
do
    kprop -f /var/kerberos/krb5kdc/slavedump $kdc
done
```



Kerberos

Enable Kerberos: true

Authentication on WEB UIs: false

Kerberos KDC type: MIT

KDC hosts

KDC hosts [0]: kerberos-primary-host.ru-central1.internal

KDC hosts [1]: kerberos-secondary-host.ru-central1.internal

Realm: ADREALM.IO

Domains

Domains [0]: adrealm.io

Domains [1]: .adrealm.io

Kadmin server: kerberos-primary-host.ru-central1.internal

Kadmin principal: root/admin@ADREALM.IO

Kadmin password: <secret>

Keytabs directory: /etc/security/keytabs

Additional realms <not set>

Пример настроек нескольких хостов KDC

Настройка использования MIT Kerberos для Kerberos-аутентификации с помощью ADCM

Константин Алпашкин, Сергей Тихомиров

Содержание

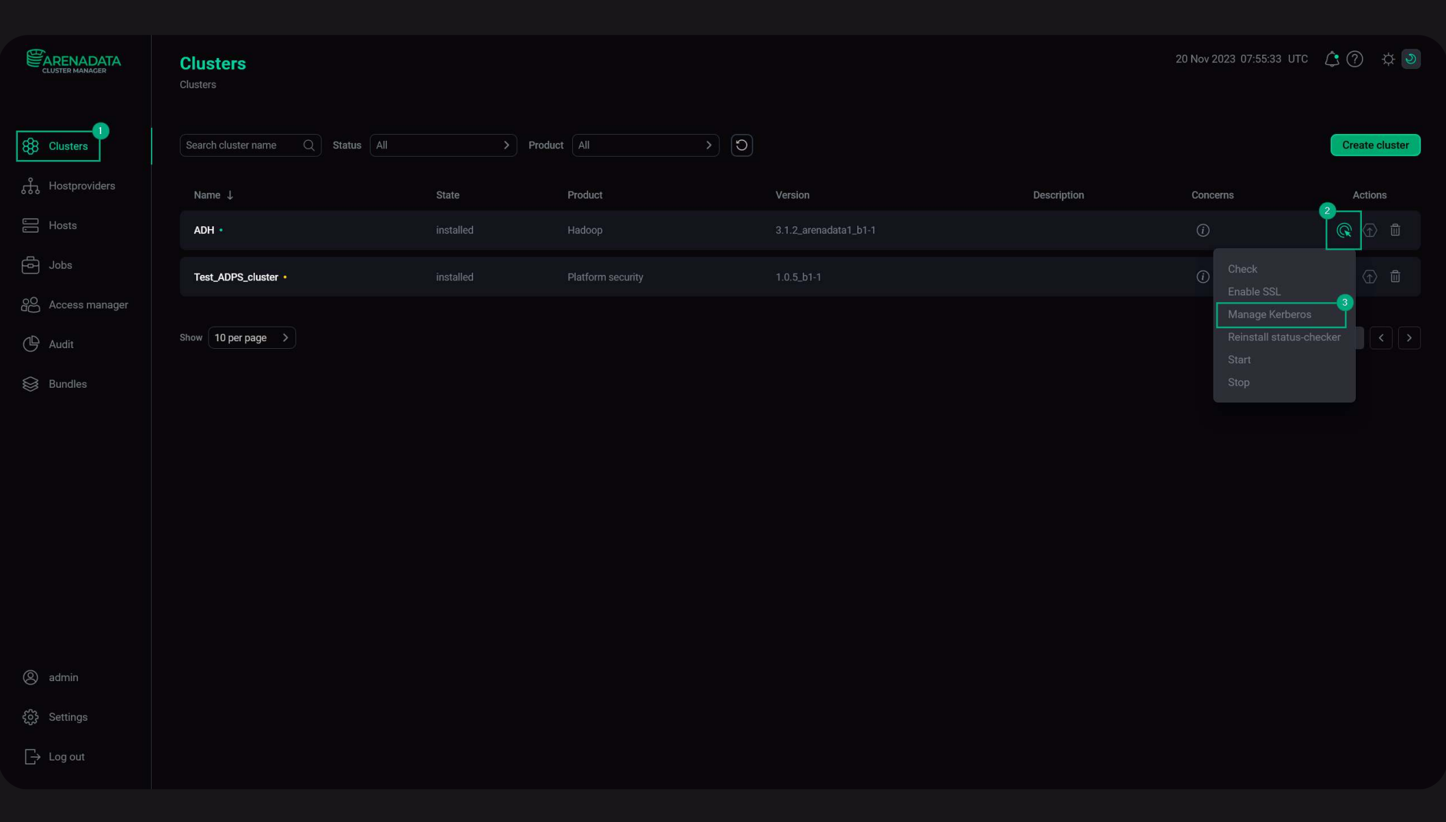
Назад к содержанию

Обзор
Параметры MIT Kerberos
Проверка успешной керберизации кластера с помощью MIT Kerberos
  Проверка HDFS перед керберизацией кластера
  Проверка HDFS после керберизации кластера

Обзор

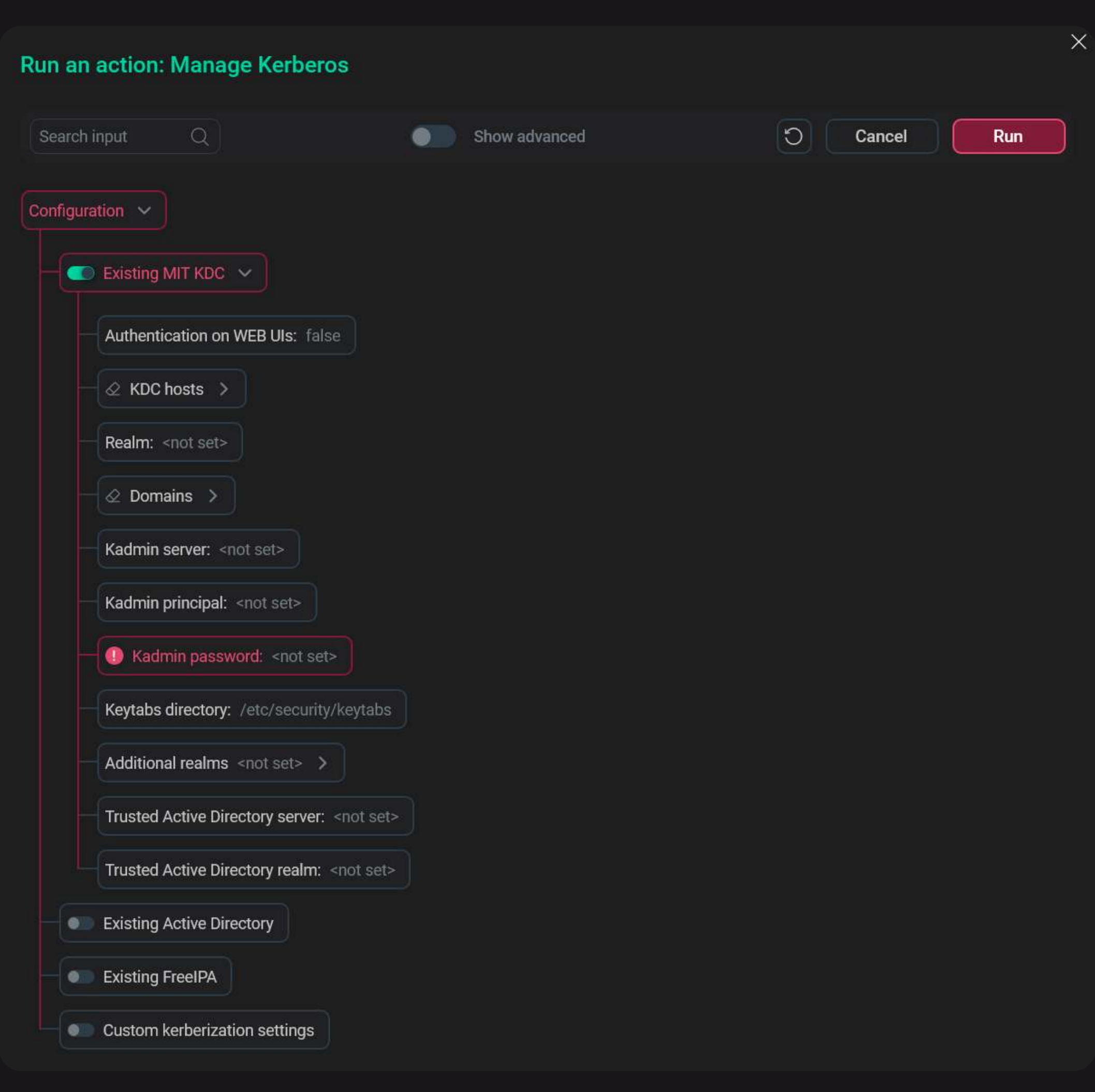
Для керберизации кластера с помощью MIT KDC выполните описанные ниже шаги:

1. В веб-интерфейсе ADCM перейдите на страницу ***Clusters***. Выберите установленный и подготовленный кластер ADH и запустите действие ***Manage Kerberos***.


ARENADATA CLUSTER MANAGER
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Clusters
20 Nov 2023 07:55:33 UTC
Search cluster name
Status
All
Product
All
Create cluster
Name
State
Product
Version
Description
Concerns
Actions
ADH
installed
Hadoop
3.1.2_arenadata1_b1-1
Test_ADPS_cluster
installed
Platform security
1.0.5_b1-1
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
Show
10 per page
1
2
3

Manage Kerberos

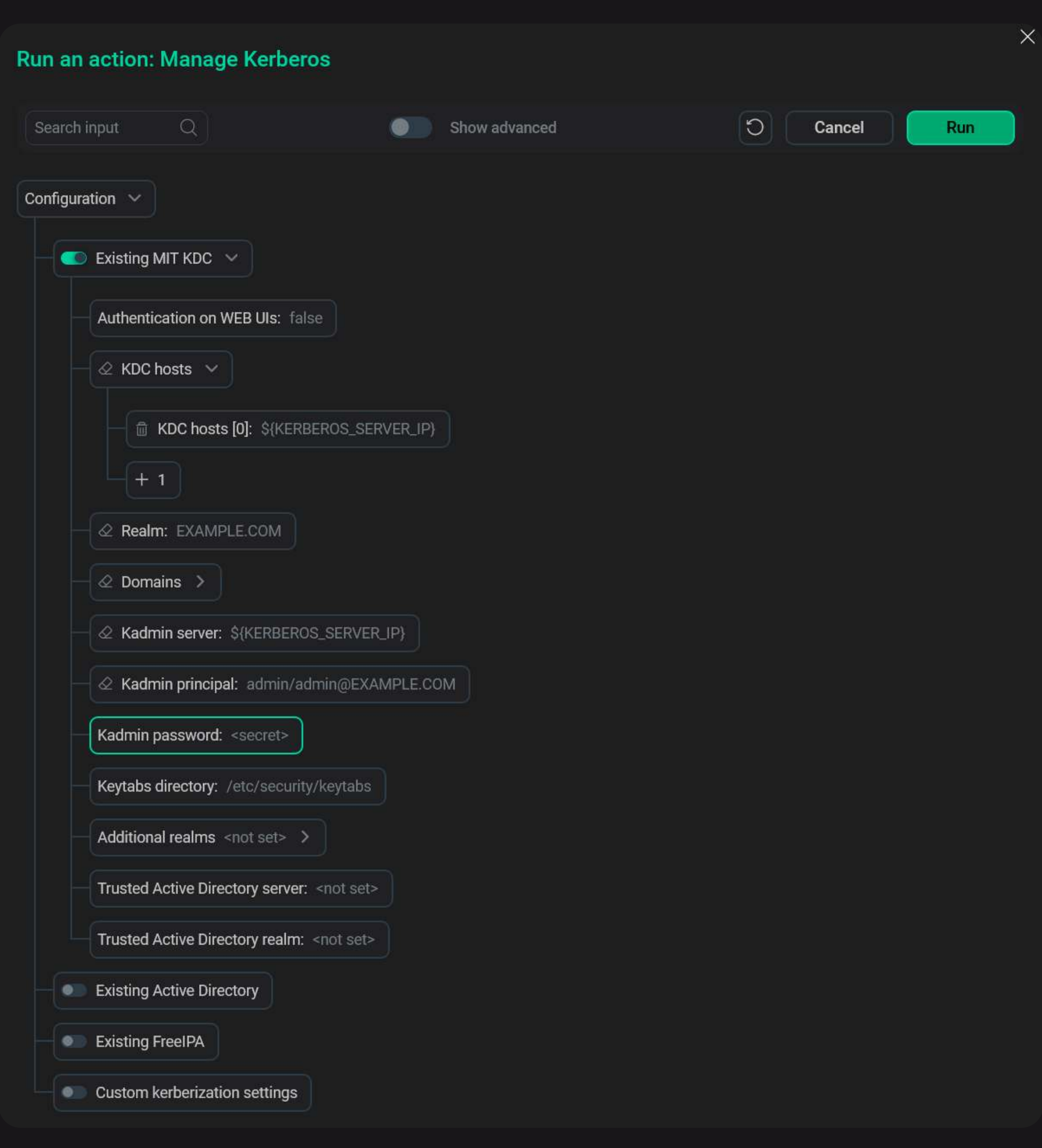
2. В появившемся окне выберите опцию ***Existing MIT KDC***.


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration

Existing MIT KDC
Authentication on WEB UIs: false
KDC hosts
Realm: <not set>
Domains
Kadmin server: <not set>
Kadmin principal: <not set>
Kadmin password: <not set>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
Trusted Active Directory server: <not set>
Trusted Active Directory realm: <not set>
Existing Active Directory
Existing FreeIPA
Custom kerberization settings

Выбор соответствующей опции

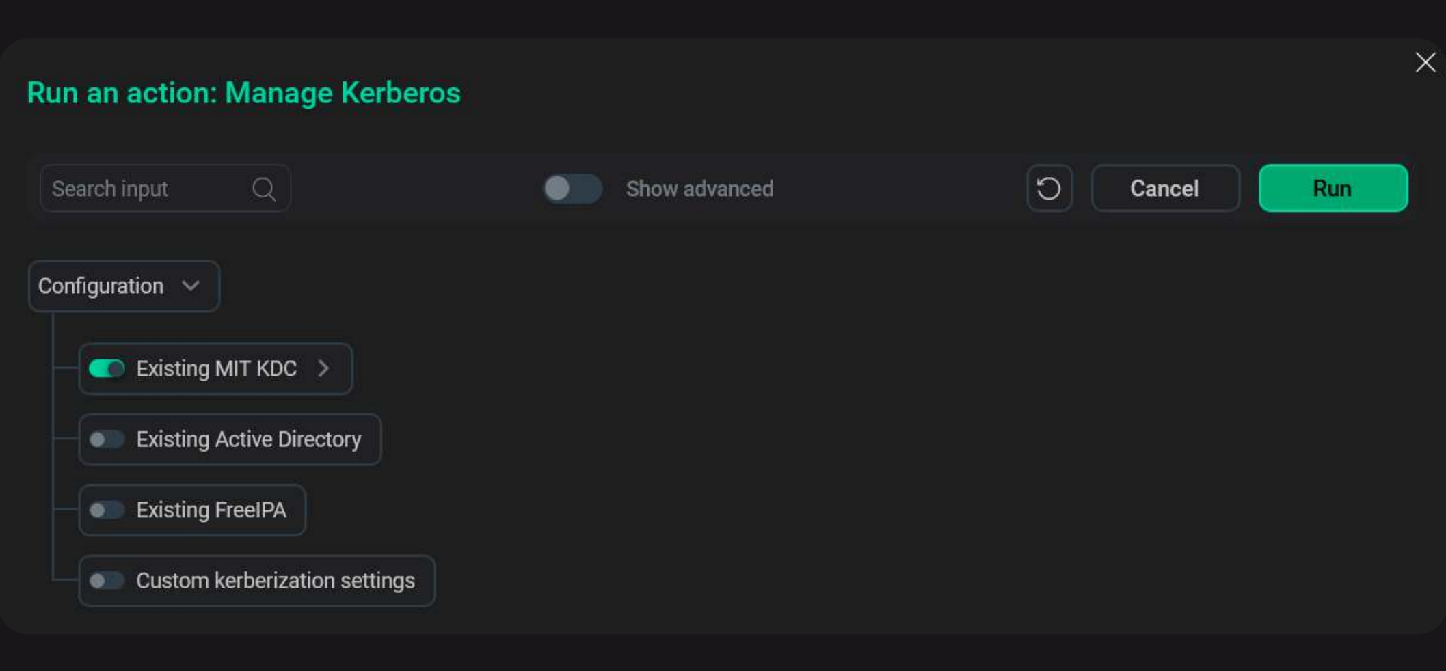
3. Заполните параметры MIT Kerberos (настроенного ранее).


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration

Existing MIT KDC
Authentication on WEB UIs: false
KDC hosts
KDC hosts [0]: ${KERBEROS_SERVER_IP}
+ 1
Realm: EXAMPLE.COM
Domains
Kadmin server: ${KERBEROS_SERVER_IP}
Kadmin principal: admin/admin@EXAMPLE.COM
Kadmin password: <secret>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
Trusted Active Directory server: <not set>
Trusted Active Directory realm: <not set>
Existing Active Directory
Existing FreeIPA
Custom kerberization settings

Параметры MIT Kerberos

4. Нажмите ***Run*** и дождитесь окончания действия. Затем приступайте к настройке Kerberos в кластере.


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Custom kerberization settings

Запуск действия

Параметры MIT Kerberos

Параметр	Описание
Authentication on WEB UIs	Включение Kerberos-аутентификации для Web UI
KDC hosts	Один или более MIT KDC-хостов
Realm	Kerberos realm — сеть, содержащая KDC-хосты и множество клиентов
Domains	Домены, в которых находятся хосты
Kadmin server	Адрес хоста, на котором запущен `kadmin`
Kadmin principal	Имя принципала для подключения через `kadmin`, например `admin@RU-CENTRAL1.INTERNAL`
Kadmin password	Пароль принципала для подключения через `kadmin`
Keytabs directory	Директория keytab-файла, содержащего записи одного или нескольких принципалов и их ключи
Additional realms	Дополнительные сети Kerberos realm
Trusted Active Directory server	Сервер с Active Directory для одностороненней доверенности от MIT Kerberos KDC внутри realm
Trusted Active Directory realm	Realm с Active Directory для одностороненней доверенности от MIT Kerberos KDC внутри realm

Проверка успешной керберизации кластера с помощью MIT Kerberos

После керберизации кластера любое действие пользователя должно выполняться после успешной аутентификации. Ниже вы можете найти примеры запросов HDFS до и после керберизации кластера с помощью MIT Kerberos.

Проверка HDFS перед керберизацией кластера

HDFS CLI

Пример HDFS-команды:

```
$ sudo -u hdfs hdfs dfs -touch /tmp/arenadata_test.txt
$ sudo -u hdfs hdfs dfs -ls /tmp
```

Вывод имеет следующий вид:

```
Found 1 items
-rw-r--r--   3 hdfs hadoop          0 2022-02-05 11:17 /tmp/arenadata_test.txt
```

HDFS HttpFS

Запрос curl на создание сущности в HDFS:

```
$ curl -I -H "Content-Type:application/octet-stream" -X PUT 'http://httpfs_hostname:14000/webhdfs/v1/tmp/arenadata_httpfs_test.
txt?op=CREATE&data=true&user.name=hdfs'
```

Ответ имеет следующий вид:

```
HTTP/1.1 201 Created
Date: Sat, 05 Feb 2022 11:23:31 GMT
Cache-Control: no-cache
Expires: Sat, 05 Feb 2022 11:23:31 GMT
Date: Sat, 05 Feb 2022 11:23:31 GMT
Pragma: no-cache
Set-Cookie: hadoop.auth="u=hdfs&p=hdfs&t=simple-dt&e=1628198611064&s=gS6tylp5MZw+aiHs1EzuNfd1qqbJpFAGeGLxTtXxZfg="; Path=/;
HttpOnly
Content-Type: application/json;charset=utf-8
Content-Length: 0
```

Запрос curl на получение данных о файле в HDFS:

```
$ curl -H "Content-Type:application/json" -X GET 'http://httpfs_hostname:14000/webhdfs/v1/tmp?op=LISTSTATUS&user.name=hdfs'
```

Ответ имеет следующий вид:

```
{"FileStatuses":{"FileStatus":[{"pathSuffix":"arenadata_httpfs_test.
txt","type":"FILE","length":0,"owner":"hdfs","group":"hadoop","permission":"755","accessTime":1628162611089,"modificationTime"
:1628162611113,"blockSize":134217728,"replication":3}]}}
```

Проверка HDFS после керберизации кластера

HDFS CLI

Пример HDFS-команды:

```
$ sudo -u hdfs hdfs dfs -ls /tmp
```

Результат (ошибка доступа):

```
2022-02-05 11:34:14,753 WARN ipc.Client: Exception encountered while connecting to the server : org.apache.hadoop.security.
AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
2022-02-05 11:34:14,765 WARN ipc.Client: Exception encountered while connecting to the server : org.apache.hadoop.security.
AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
2022-02-05 11:34:14,767 INFO retry.RetryInvocationHandler: java.io.IOException: DestHost:destPort namenode_host:8020 ,
LocalHost:localPort namenode_host/namenode_ip:0. Failed on local exception: java.io.IOException: org.apache.hadoop.security.
AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS], while invoking ClientNamenodeProtocolTranslatorPB.
getFileInfo over namenode_host/namenode_ip:8020 after 1 failover attempts. Trying to failover after sleeping for 1087ms.
```

Создание тикета:

```
$ sudo -u hdfs kinit -k -t /etc/security/keytabs/hdfs.service.keytab hdfs/node_hostname@REALM
```

Результат (успешный):

```
Found 1 items
-rw-r--r--   3 hdfs hadoop          0 2022-02-05 11:17 /tmp/arenadata_test.txt
```

HDFS HttpFS

HDFS-запрос:

```
$ sudo -u hdfs curl -H "Content-Type:application/json" -X GET 'http://httpfs_hostname:14000/webhdfs/v1/tmp?op=LISTSTATUS&user.
name=hdfs'
```

Результат (ошибка доступа):

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8"/>
<title>Error 401 Authentication required</title>
</head>
<body><h2>HTTP ERROR 401</h2>
<p>Problem accessing /webhdfs/v1/tmp. Reason:
<pre>    Authentication required</pre></p>
</body>
</html>
```

Создание тикета:

```
$ sudo -u hdfs kinit -k -t /etc/security/keytabs/HTTP.service.keytab HTTP/node_hostname@REALM
```

HDFS-запрос:

```
$ sudo -u hdfs curl -sfSIL -I -H "Content-Type:application/octet-stream" -X PUT --negotiate -u :
'http://httpfs_hostname:14000/webhdfs/v1/tmp/arenadata_httpfs_test.txt?op=CREATE&data=true&user.name=hdfs'
```

Результат (успешный):

```
HTTP/1.1 201 Created
Date: Sat, 05 Feb 2022 11:23:31 GMT
Cache-Control: no-cache
Expires: Sat, 05 Feb 2022 11:23:31 GMT
Date: Sat, 05 Feb 2022 11:23:31 GMT
Pragma: no-cache
Set-Cookie: hadoop.auth="u=hdfs&p=hdfs&t=simple-dt&e=1628198611064&s=gS6tylp5MZw+aiHs1EzuNfd1qqbJpFAGeGLxTtXxZfg="; Path=/;
HttpOnly
Content-Type: application/json;charset=utf-8
Content-Length: 0
```

Настройка Active Directory для Kerberos

Константин Алпашкин

Назад к содержанию

Если ADH-кластер Kerberos используется с Active Directory, Kerberos запрашивает у KDC информацию об учетных записях администратора и всех пользователей. После получения этой информации Kerberos автоматически создает принципалов для каждого пользователя и службы.

Вы можете создать любую конфигурацию для вашего кластера, при этом у каждого пользователя и службы будет соответствующий принципал.

В дополнение к Active Directory, схема использует ADPS для управления разрешениями пользователей.

База данных принципалов Active Directory должна быть синхронизирована с базой данных принципалов ADPS. Эта синхронизация выполняется автоматически перед настройкой Kerberos в кластере.

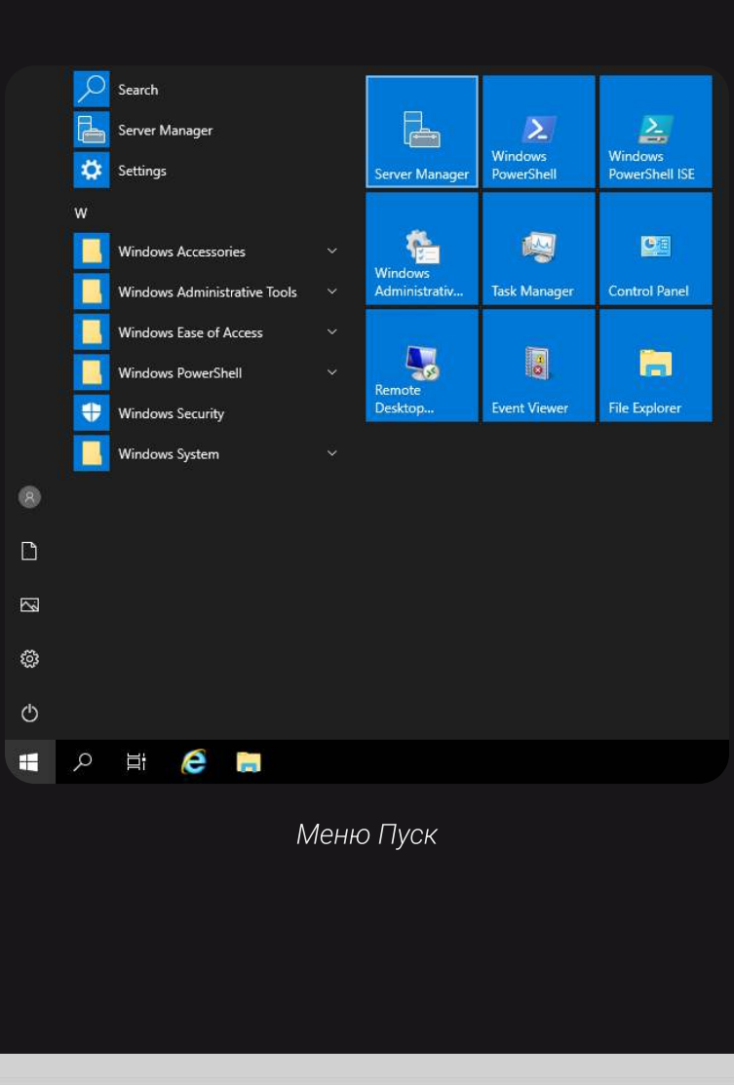
Каждое последующее изменение в базе данных принципалов Active Directory синхронизируется с базой данных ADPS автоматически. В этом случае при работе с кластером ADH все принципалы получают TGT от KDC AD.

Установка службы сертификации AD

Константин Алпашкин

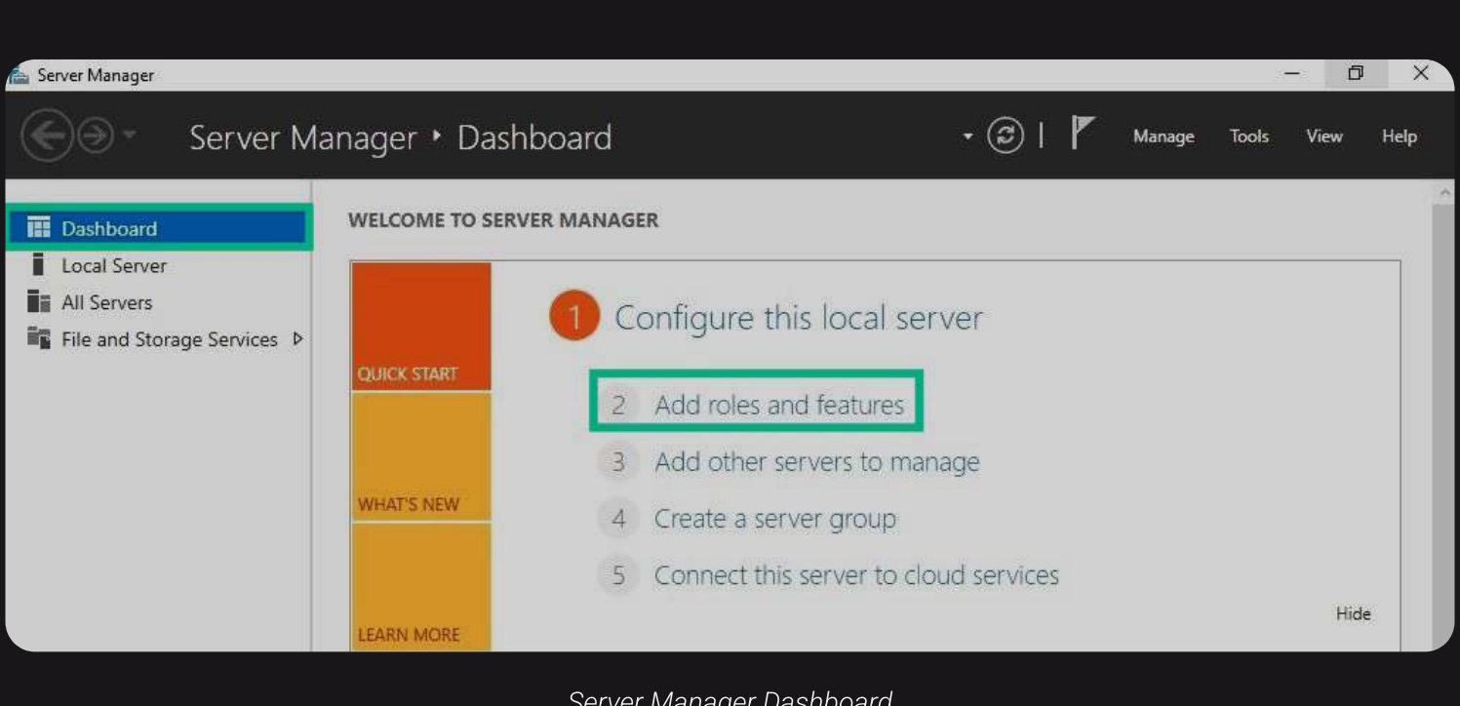
Назад к содержанию

1. Откройте меню ***Пуск*** и запустите утилиту ***Server Manager***.


Search
Server Manager
Settings
W
Windows Accessories
Windows Administrative Tools
Windows Ease of Access
Windows PowerShell
Windows Security
Windows System
Server Manager
Windows PowerShell
Windows PowerShell ISE
Windows Administrativ...
Task Manager
Control Panel
Remote Desktop...
Event Viewer
File Explorer

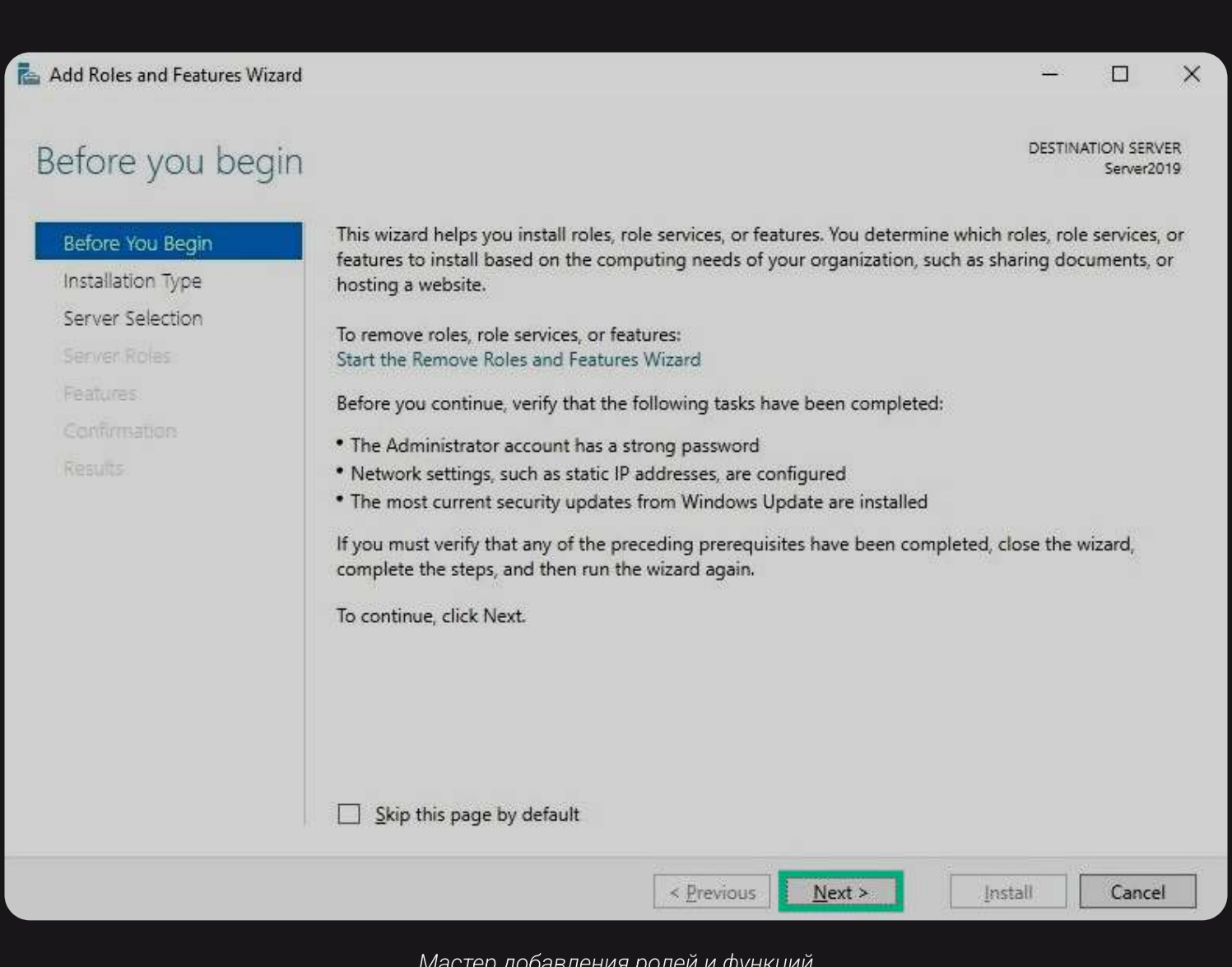
Меню Пуск

2. Кликните ***Dashboard***, а затем ***Add roles and features***.


Server Manager
Server Manager ▸ Dashboard
Manage
Tools
View
Help
Dashboard
Local Server
All Servers
File and Storage Services
WELCOME TO SERVER MANAGER
QUICK START
WHAT'S NEW
LEARN MORE
1 Configure this local server
2 Add roles and features
3 Add other servers to manage
4 Create a server group
5 Connect this server to cloud services
Hide

Server Manager Dashboard

3. Кликните ***Next***.


Add Roles and Features Wizard

Before you begin

DESTINATION SERVER
Server2019

This wizard helps you install roles, role services, or features. You determine which roles, role services, or features to install based on the computing needs of your organization, such as sharing documents, or hosting a website.

To remove roles, role services, or features:
Start the Remove Roles and Features Wizard

Before you continue, verify that the following tasks have been completed:

- The Administrator account has a strong password
- Network settings, such as static IP addresses, are configured
- The most current security updates from Windows Update are installed

If you must verify that any of the preceding prerequisites have been completed, close the wizard, complete the steps, and then run the wizard again.

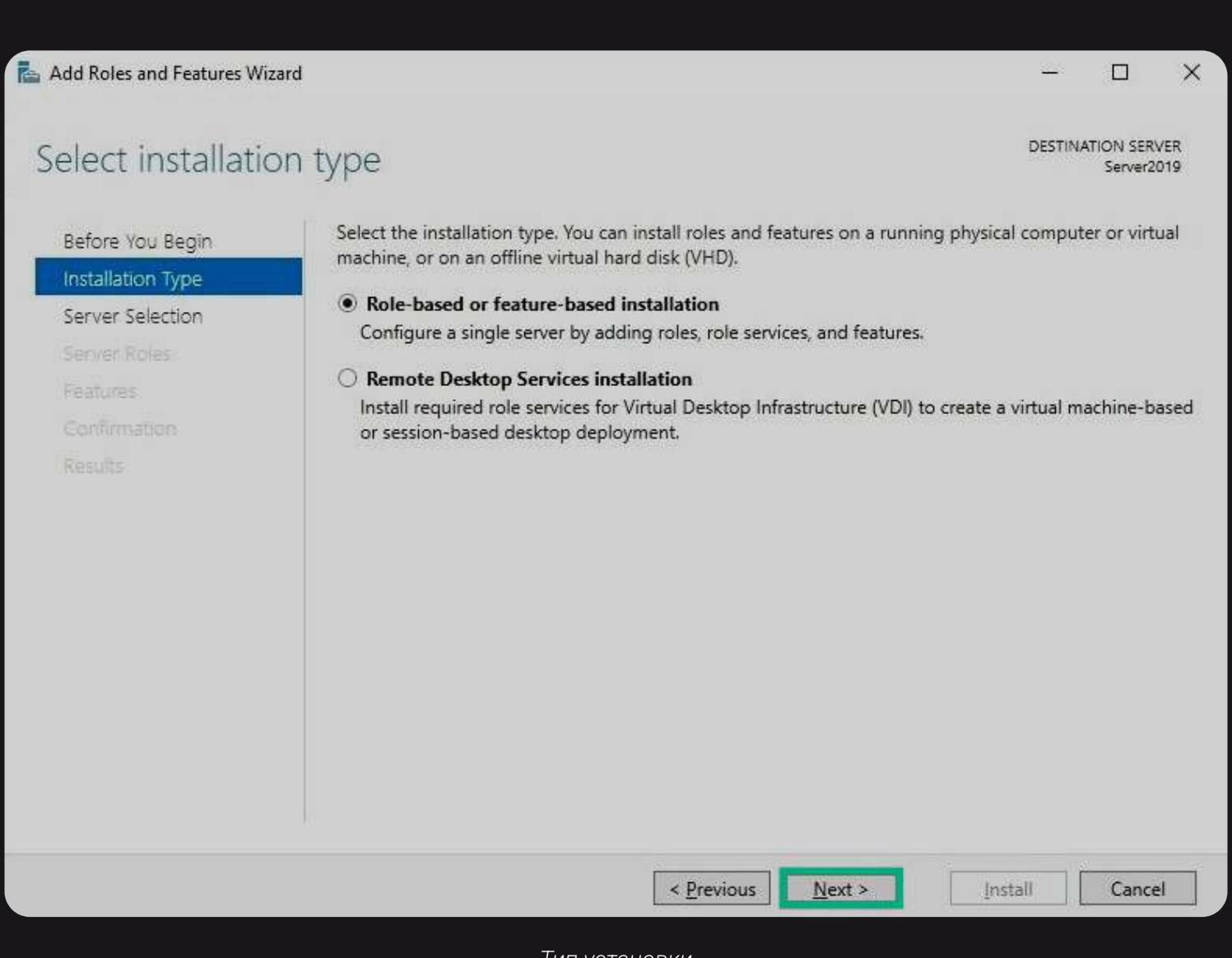
To continue, click Next.

☐ Skip this page by default

Before You Begin
Installation Type
Server Selection
Server Roles
Features
Confirmation
Results
< Previous
Next >
Install
Cancel

Мастер добавления ролей и функций

4. Выберите ***Role-based or feature-based installation*** и кликните ***Next***.


Add Roles and Features Wizard

Select installation type

DESTINATION SERVER
Server2019

Select the installation type. You can install roles and features on a running physical computer or virtual machine, or on an offline virtual hard disk (VHD).

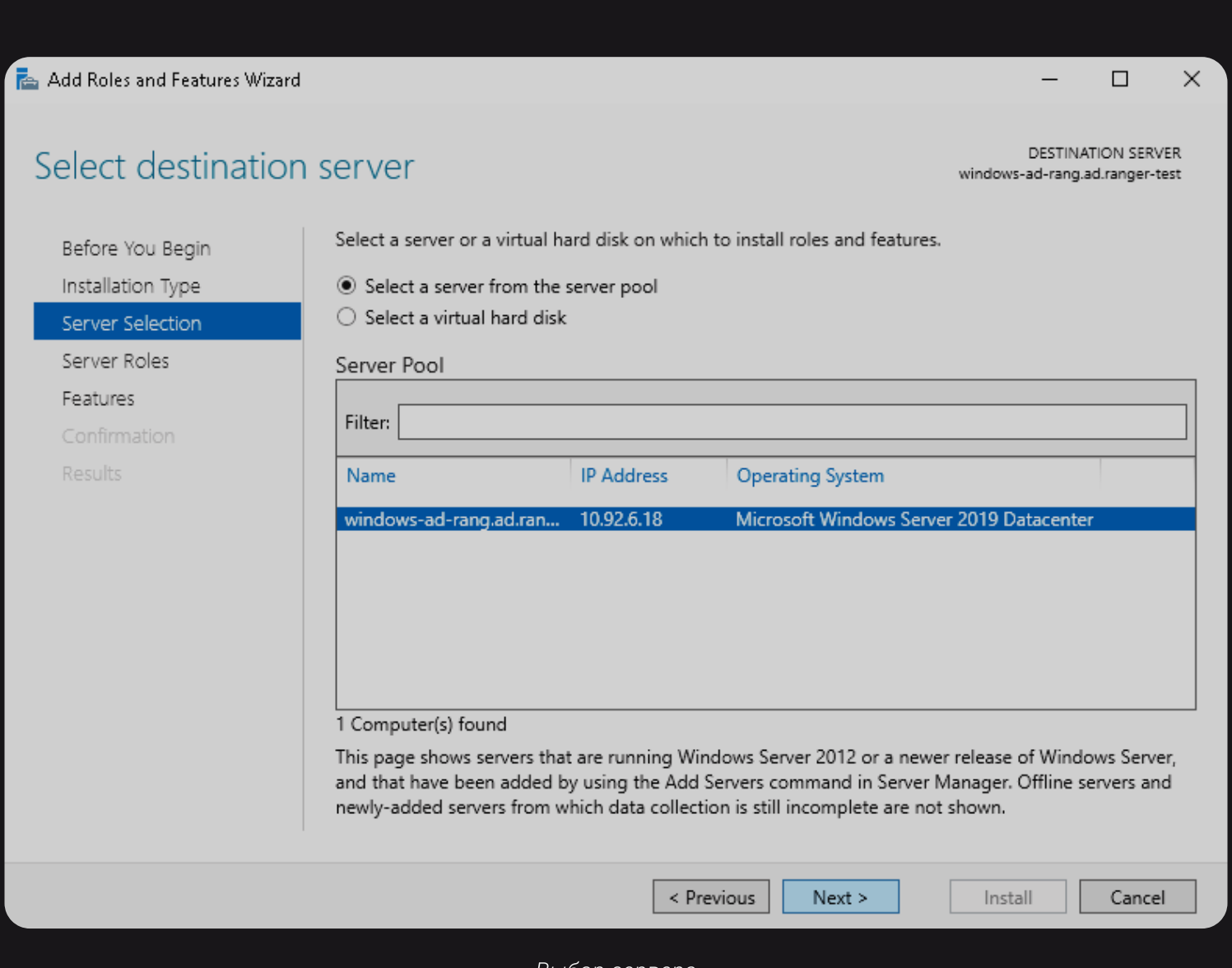
● **Role-based or feature-based installation**
Configure a single server by adding roles, role services, and features.

○ **Remote Desktop Services installation**
Install required role services for Virtual Desktop Infrastructure (VDI) to create a virtual machine-based or session-based desktop deployment.

Before You Begin
Installation Type
Server Selection
Server Roles
Features
Confirmation
Results
< Previous
Next >
Install
Cancel

Тип установки

5. Выберите ***Select a server from the server pool*** и кликните ***Next***.


Add Roles and Features Wizard

Select destination server

DESTINATION SERVER
windows-ad-rang.ad.ranger-test

Select a server or a virtual hard disk on which to install roles and features.

● Select a server from the server pool
○ Select a virtual hard disk

Server Pool

Filter:

Name	IP Address	Operating System
windows-ad-rang.ad.ran...	10.92.6.18	Microsoft Windows Server 2019 Datacenter

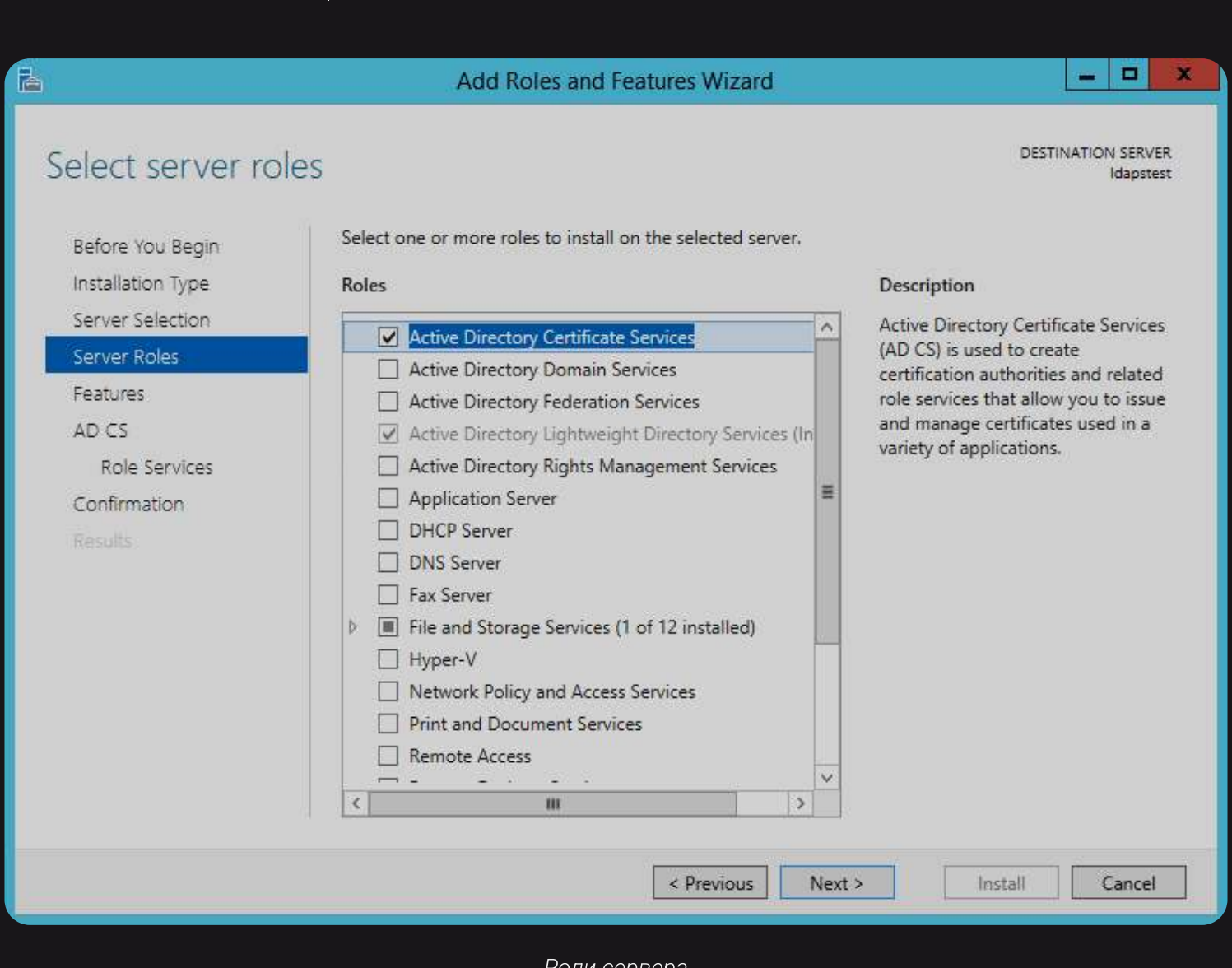
1 Computer(s) found

This page shows servers that are running Windows Server 2012 or a newer release of Windows Server, and that have been added by using the Add Servers command in Server Manager. Offline servers and newly-added servers from which data collection is still incomplete are not shown.

Before You Begin
Installation Type
Server Selection
Server Roles
Features
Confirmation
Results
< Previous
Next >
Install
Cancel

Выбор сервера

6. Выберите ***Active Directory Certificate Services*** из списка ролей и кликните ***Next***.


Add Roles and Features Wizard

Select server roles

DESTINATION SERVER
ldapstest

Select one or more roles to install on the selected server.

Roles

☑ Active Directory Certificate Services
☐ Active Directory Domain Services
☐ Active Directory Federation Services
☑ Active Directory Lightweight Directory Services (In
☐ Active Directory Rights Management Services
☐ Application Server
☐ DHCP Server
☐ DNS Server
☐ Fax Server
▣ File and Storage Services (1 of 12 installed)
☐ Hyper-V
☐ Network Policy and Access Services
☐ Print and Document Services
☐ Remote Access

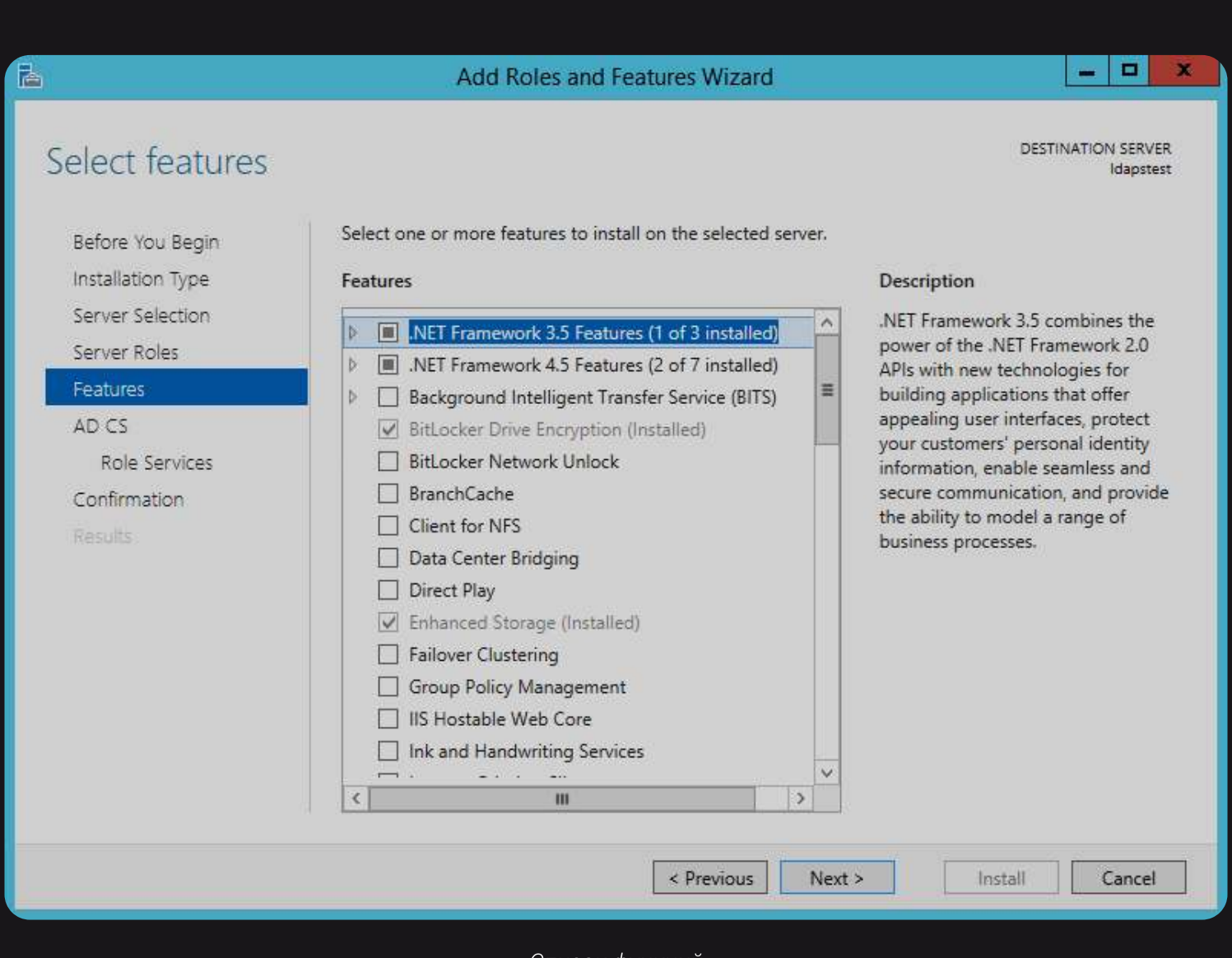
Description

Active Directory Certificate Services (AD CS) is used to create certification authorities and related role services that allow you to issue and manage certificates used in a variety of applications.

Before You Begin
Installation Type
Server Selection
Server Roles
Features
AD CS
Role Services
Confirmation
Results
< Previous
Next >
Install
Cancel

Роли сервера

7. Не добавляйте в установку дополнительные программные пакеты; кликните ***Next***.


Add Roles and Features Wizard

Select features

DESTINATION SERVER
ldapstest

Select one or more features to install on the selected server.

Features

▣ .NET Framework 3.5 Features (1 of 3 installed)
▣ .NET Framework 4.5 Features (2 of 7 installed)
☐ Background Intelligent Transfer Service (BITS)
☑ BitLocker Drive Encryption (Installed)
☐ BitLocker Network Unlock
☐ BranchCache
☐ Client for NFS
☐ Data Center Bridging
☐ Direct Play
☑ Enhanced Storage (Installed)
☐ Failover Clustering
☐ Group Policy Management
☐ IIS Hostable Web Core
☐ Ink and Handwriting Services

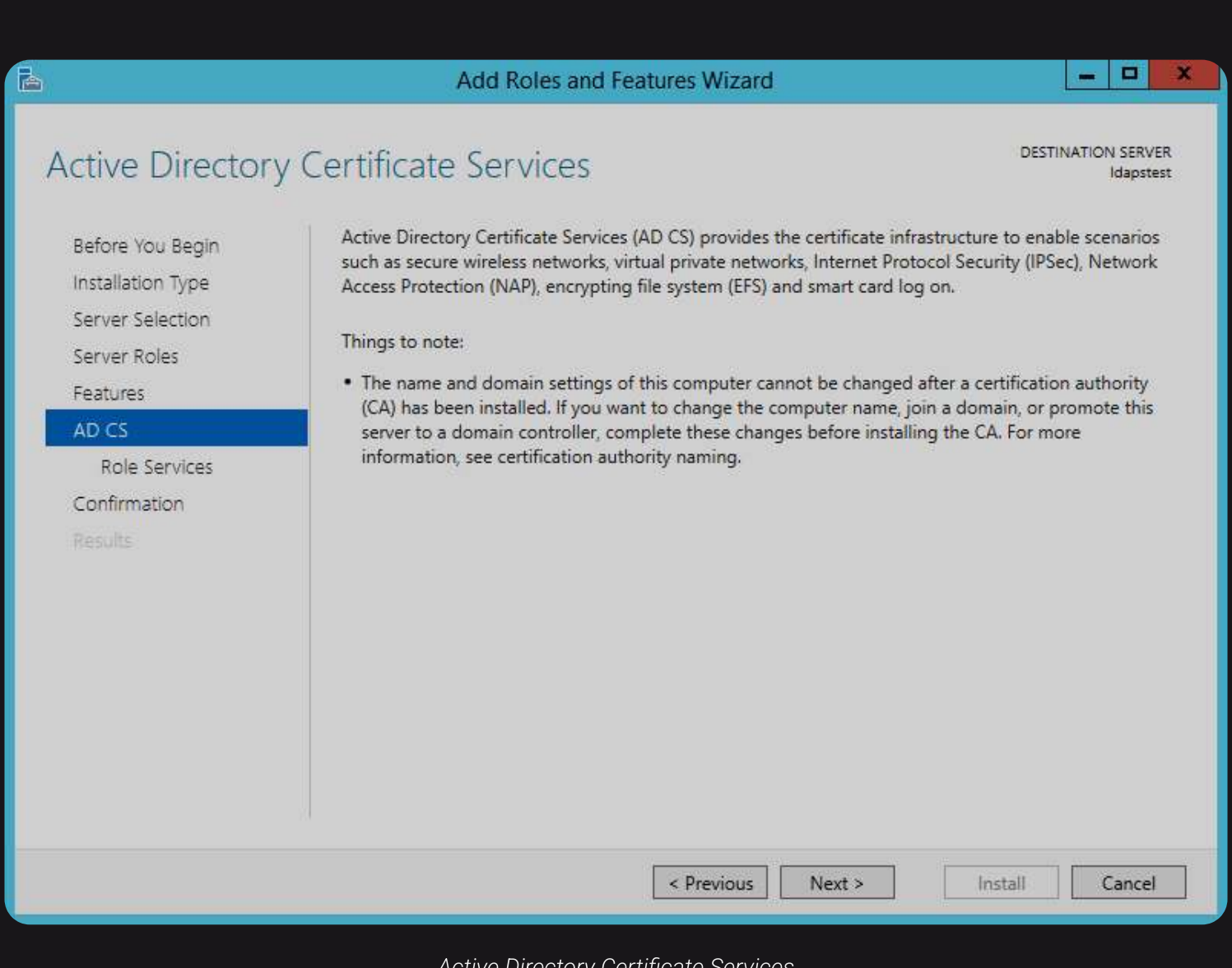
Description

.NET Framework 3.5 combines the power of the .NET Framework 2.0 APIs with new technologies for building applications that offer appealing user interfaces, protect your customers' personal identity information, enable seamless and secure communication, and provide the ability to model a range of business processes.

Before You Begin
Installation Type
Server Selection
Server Roles
Features
AD CS
Role Services
Confirmation
Results
< Previous
Next >
Install
Cancel

Список функций

8. Кликните ***Next***.


Add Roles and Features Wizard

Active Directory Certificate Services

DESTINATION SERVER
ldapstest

Active Directory Certificate Services (AD CS) provides the certificate infrastructure to enable scenarios such as secure wireless networks, virtual private networks, Internet Protocol Security (IPSec), Network Access Protection (NAP), encrypting file system (EFS) and smart card log on.

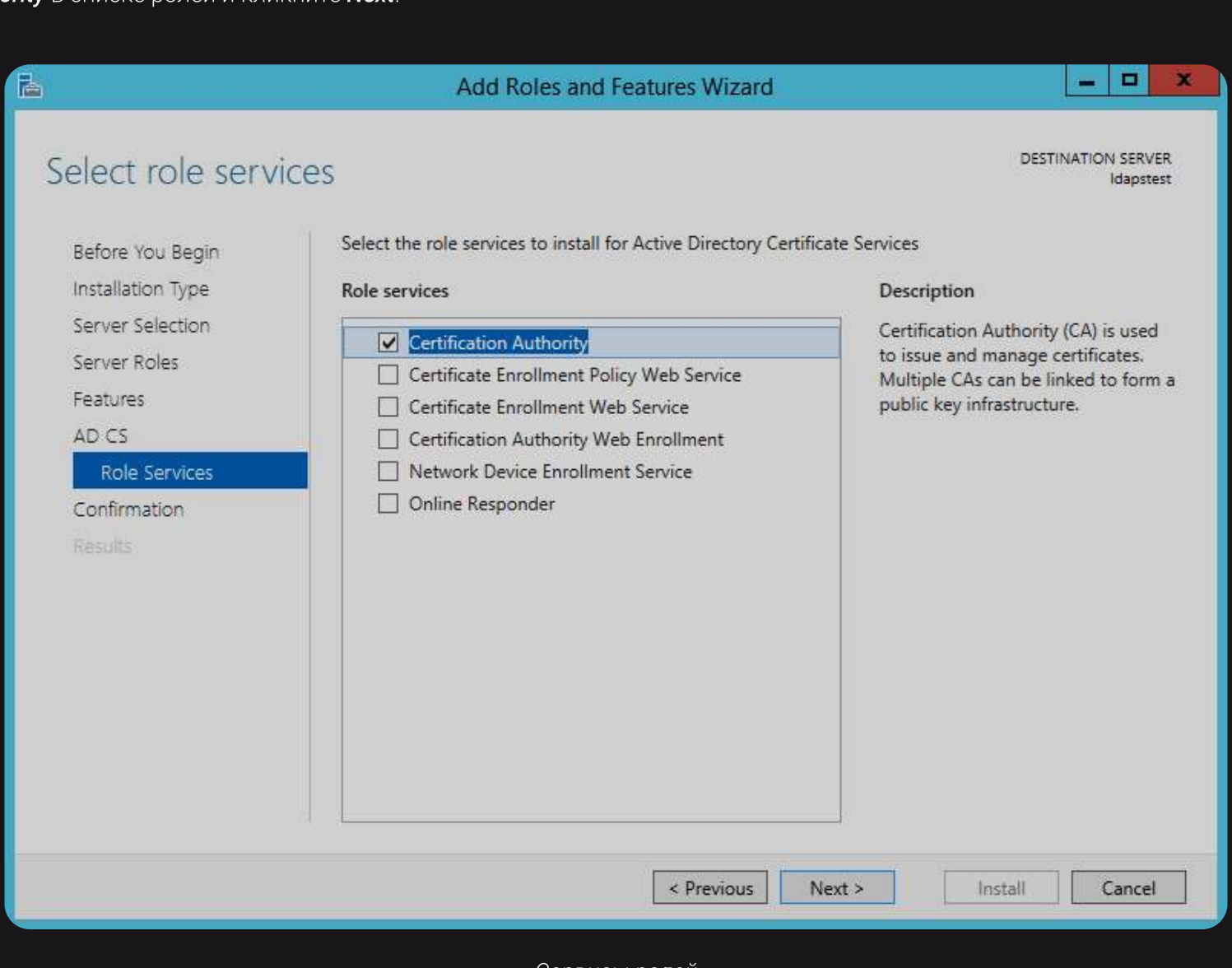
Things to note:

- The name and domain settings of this computer cannot be changed after a certification authority (CA) has been installed. If you want to change the computer name, join a domain, or promote this server to a domain controller, complete these changes before installing the CA. For more information, see certification authority naming.

Before You Begin
Installation Type
Server Selection
Server Roles
Features
AD CS
Role Services
Confirmation
Results
< Previous
Next >
Install
Cancel

Active Directory Certificate Services

9. Выберите ***Certificate Authority*** в списке ролей и кликните ***Next***.


Add Roles and Features Wizard

Select role services

DESTINATION SERVER
ldapstest

Select the role services to install for Active Directory Certificate Services

Role services

☑ Certification Authority
☐ Certificate Enrollment Policy Web Service
☐ Certificate Enrollment Web Service
☐ Certification Authority Web Enrollment
☐ Network Device Enrollment Service
☐ Online Responder

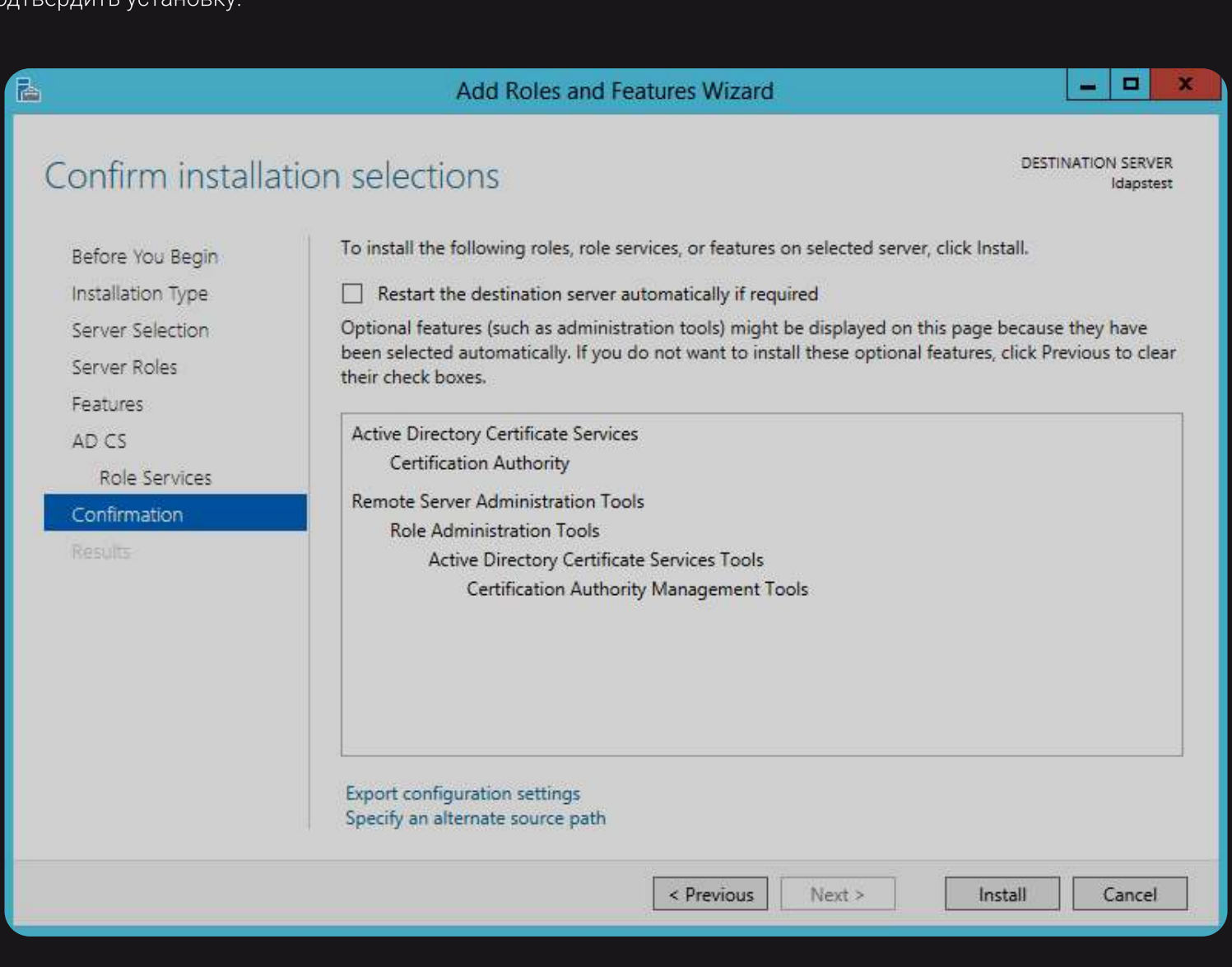
Description

Certification Authority (CA) is used to issue and manage certificates. Multiple CAs can be linked to form a public key infrastructure.

Before You Begin
Installation Type
Server Selection
Server Roles
Features
AD CS
Role Services
Confirmation
Results
< Previous
Next >
Install
Cancel

Сервисы ролей

10. Кликните ***Install***, чтобы подтвердить установку.


Add Roles and Features Wizard

Confirm installation selections

DESTINATION SERVER
ldapstest

To install the following roles, role services, or features on selected server, click Install.

☐ Restart the destination server automatically if required

Optional features (such as administration tools) might be displayed on this page because they have been selected automatically. If you do not want to install these optional features, click Previous to clear their check boxes.

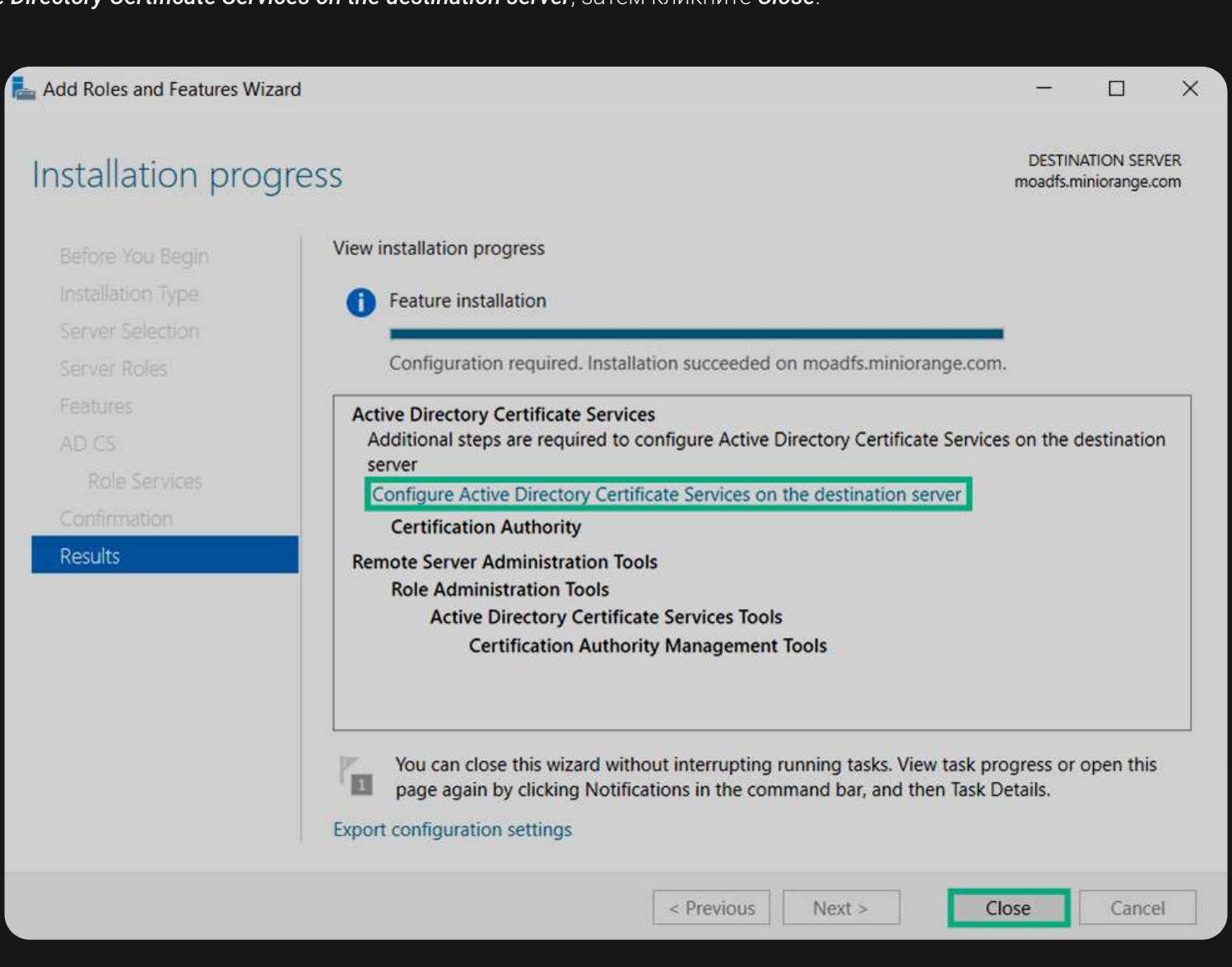
Active Directory Certificate Services
    Certification Authority
Remote Server Administration Tools
    Role Administration Tools
        Active Directory Certificate Services Tools
            Certification Authority Management Tools

Export configuration settings
Specify an alternate source path

Before You Begin
Installation Type
Server Selection
Server Roles
Features
AD CS
Role Services
Confirmation
Results
< Previous
Next >
Install
Cancel

Подтверждение

11. Кликните ***Configure Active Directory Certificate Services on the destination server***, затем кликните ***Close***.


Add Roles and Features Wizard

Installation progress

DESTINATION SERVER
moadfs.miniorange.com

View installation progress

Feature installation

Configuration required. Installation succeeded on moadfs.miniorange.com.

Active Directory Certificate Services
Additional steps are required to configure Active Directory Certificate Services on the destination server
Configure Active Directory Certificate Services on the destination server
Certification Authority
Remote Server Administration Tools
Role Administration Tools
Active Directory Certificate Services Tools
Certification Authority Management Tools

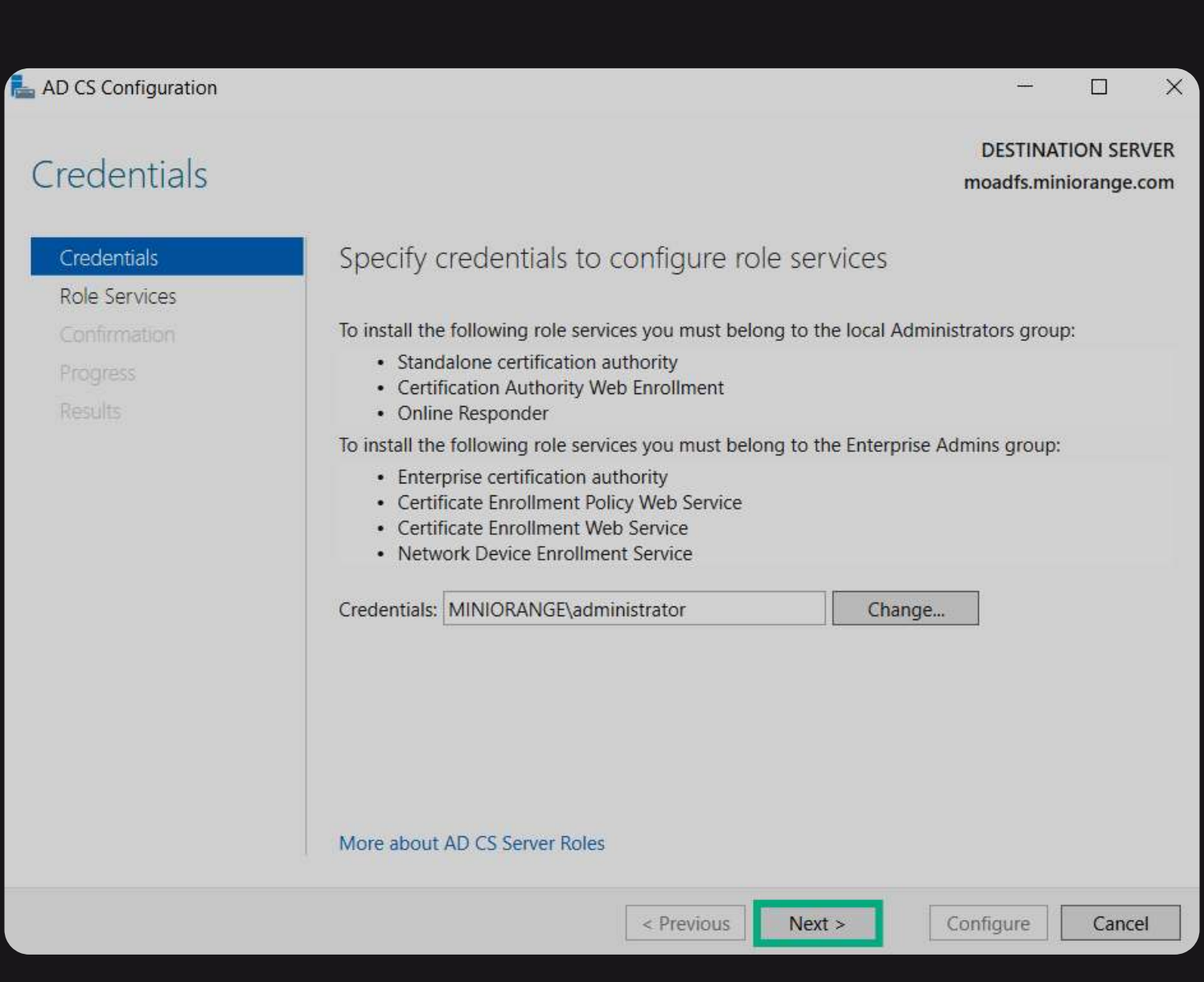
You can close this wizard without interrupting running tasks. View task progress or open this page again by clicking Notifications in the command bar, and then Task Details.

Export configuration settings

Before You Begin
Installation Type
Server Selection
Server Roles
Features
AD CS
Role Services
Confirmation
Results
< Previous
Next >
Close
Cancel

Результат установки

12. Кликните ***Next***.


AD CS Configuration

Credentials

DESTINATION SERVER
moadfs.miniorange.com

Specify credentials to configure role services

To install the following role services you must belong to the local Administrators group:

- Standalone certification authority
- Certification Authority Web Enrollment
- Online Responder

To install the following role services you must belong to the Enterprise Admins group:

- Enterprise certification authority
- Certificate Enrollment Policy Web Service
- Certificate Enrollment Web Service
- Network Device Enrollment Service

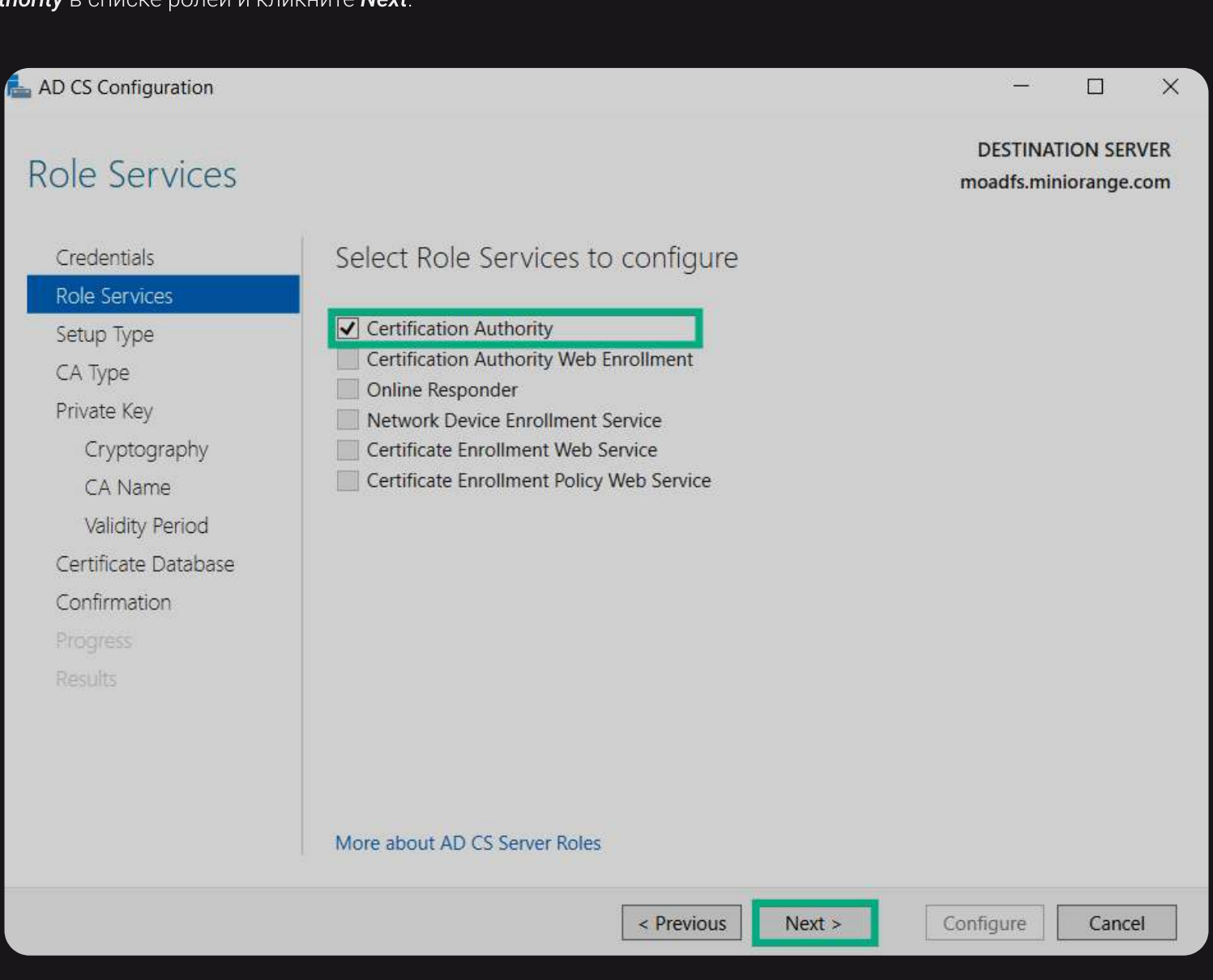
Credentials: MINIORANGE\administrator

More about AD CS Server Roles

Credentials
Role Services
Confirmation
Progress
Results
Change...
< Previous
Next >
Configure
Cancel

Учетные данные

13. Выберите ***Certification Authority*** в списке ролей и кликните ***Next***.


AD CS Configuration

Role Services

DESTINATION SERVER
moadfs.miniorange.com

Select Role Services to configure

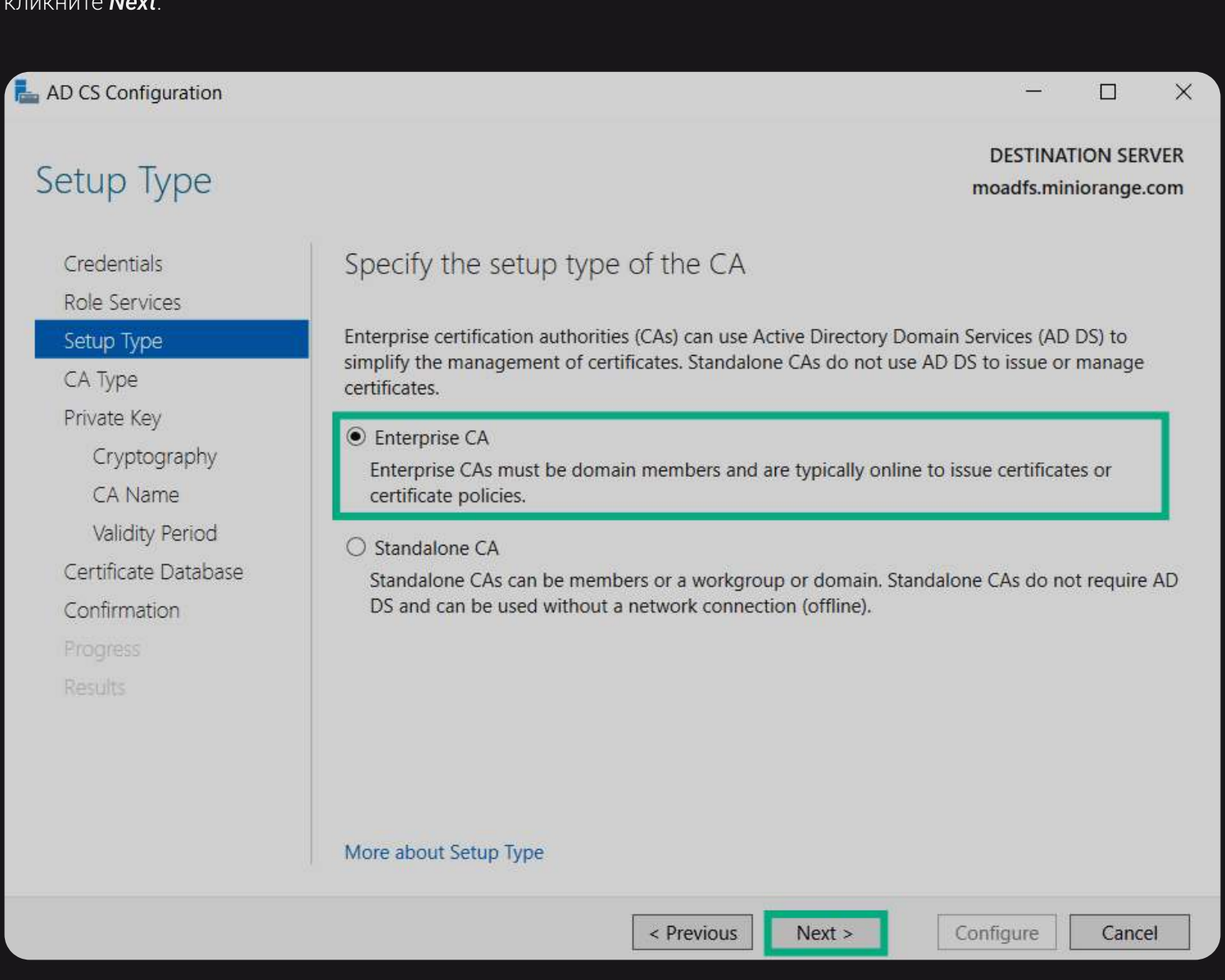
☑ Certification Authority
☐ Certification Authority Web Enrollment
☐ Online Responder
☐ Network Device Enrollment Service
☐ Certificate Enrollment Web Service
☐ Certificate Enrollment Policy Web Service

More about AD CS Server Roles

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

Сервисы ролей для настройки

14. Выберите ***Enterprise CA*** и кликните ***Next***.


AD CS Configuration

Setup Type

DESTINATION SERVER
moadfs.miniorange.com

Specify the setup type of the CA

Enterprise certification authorities (CAs) can use Active Directory Domain Services (AD DS) to simplify the management of certificates. Standalone CAs do not use AD DS to issue or manage certificates.

● Enterprise CA
Enterprise CAs must be domain members and are typically online to issue certificates or certificate policies.

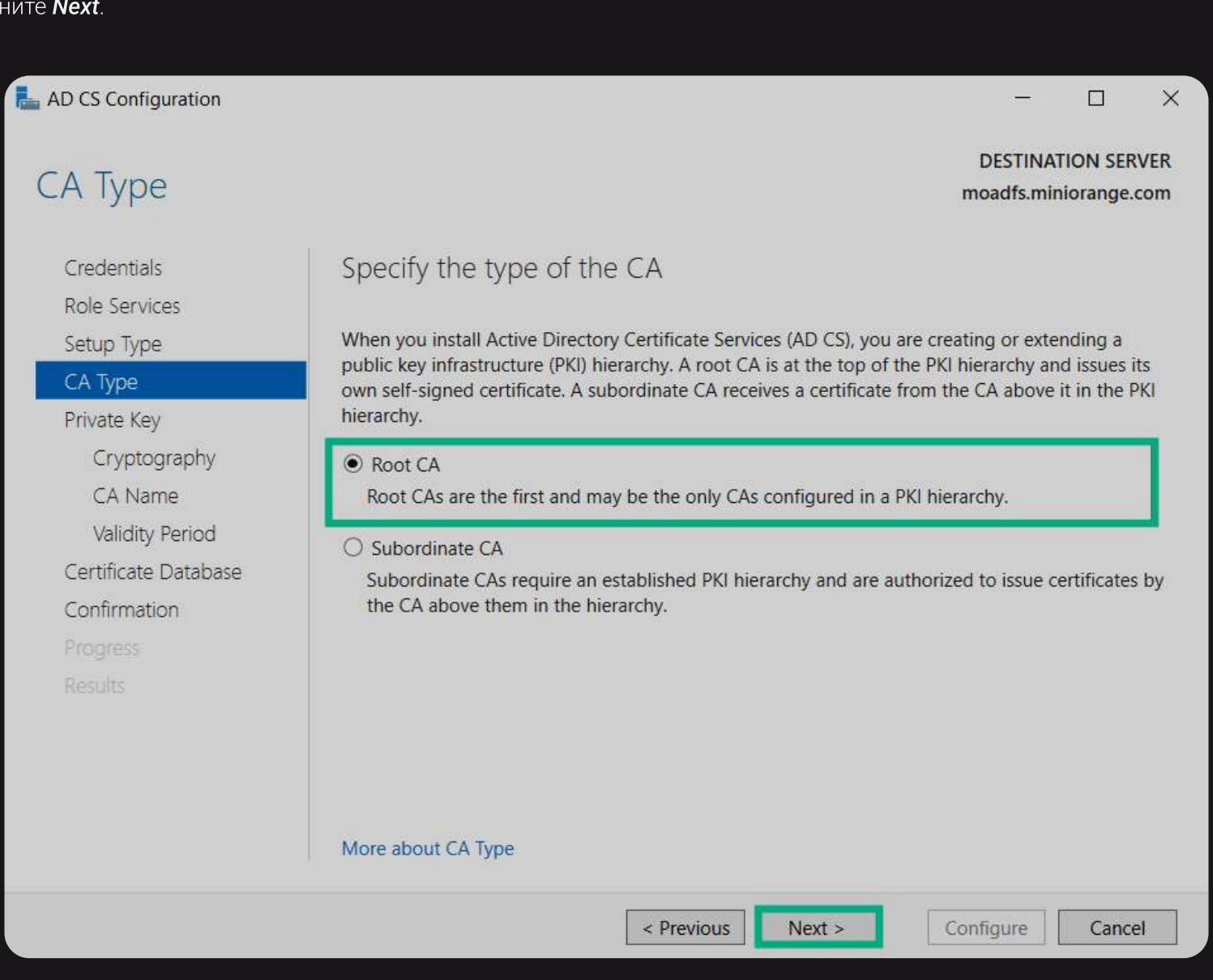
○ Standalone CA
Standalone CAs can be members or a workgroup or domain. Standalone CAs do not require AD DS and can be used without a network connection (offline).

More about Setup Type

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

Тип настройки

15. Выберите ***Root CA*** и кликните ***Next***.


AD CS Configuration

CA Type

DESTINATION SERVER
moadfs.miniorange.com

Specify the type of the CA

When you install Active Directory Certificate Services (AD CS), you are creating or extending a public key infrastructure (PKI) hierarchy. A root CA is at the top of the PKI hierarchy and issues its own self-signed certificate. A subordinate CA receives a certificate from the CA above it in the PKI hierarchy.

● Root CA
Root CAs are the first and may be the only CAs configured in a PKI hierarchy.

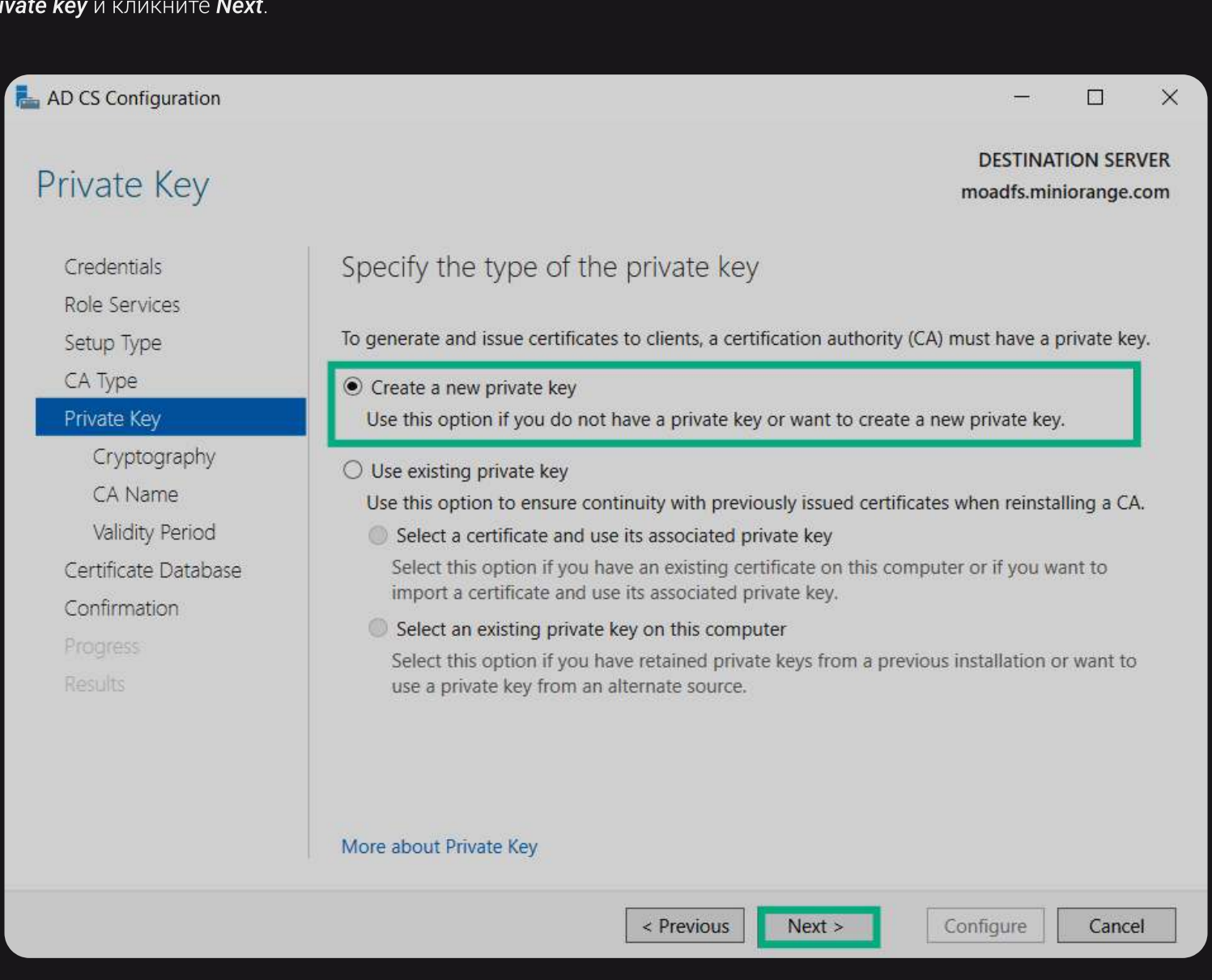
○ Subordinate CA
Subordinate CAs require an established PKI hierarchy and are authorized to issue certificates by the CA above them in the hierarchy.

More about CA Type

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

Тип центра сертификации

16. Выберите ***Create a new private key*** и кликните ***Next***.


AD CS Configuration

Private Key

DESTINATION SERVER
moadfs.miniorange.com

Specify the type of the private key

To generate and issue certificates to clients, a certification authority (CA) must have a private key.

● Create a new private key
Use this option if you do not have a private key or want to create a new private key.

○ Use existing private key
Use this option to ensure continuity with previously issued certificates when reinstalling a CA.

○ Select a certificate and use its associated private key
Select this option if you have an existing certificate on this computer or if you want to import a certificate and use its associated private key.

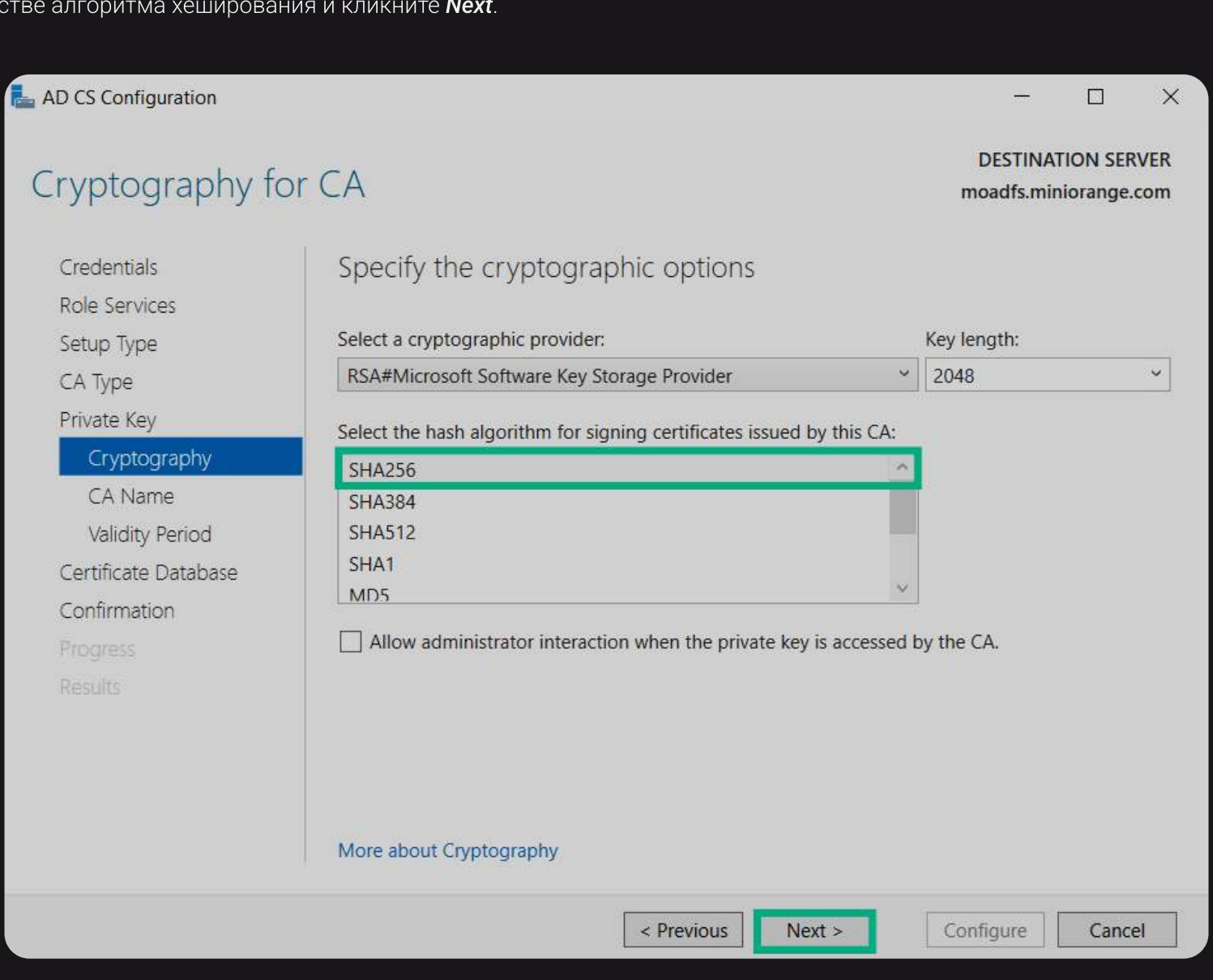
○ Select an existing private key on this computer
Select this option if you have retained private keys from a previous installation or want to use a private key from an alternate source.

More about Private Key

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

Приватный ключ

17. Выберите ***SHA256*** в качестве алгоритма хеширования и кликните ***Next***.


AD CS Configuration

Cryptography for CA

DESTINATION SERVER
moadfs.miniorange.com

Specify the cryptographic options

Select a cryptographic provider: RSA#Microsoft Software Key Storage Provider

Key length: 2048

Select the hash algorithm for signing certificates issued by this CA:

SHA256
SHA384
SHA512
SHA1
MD5

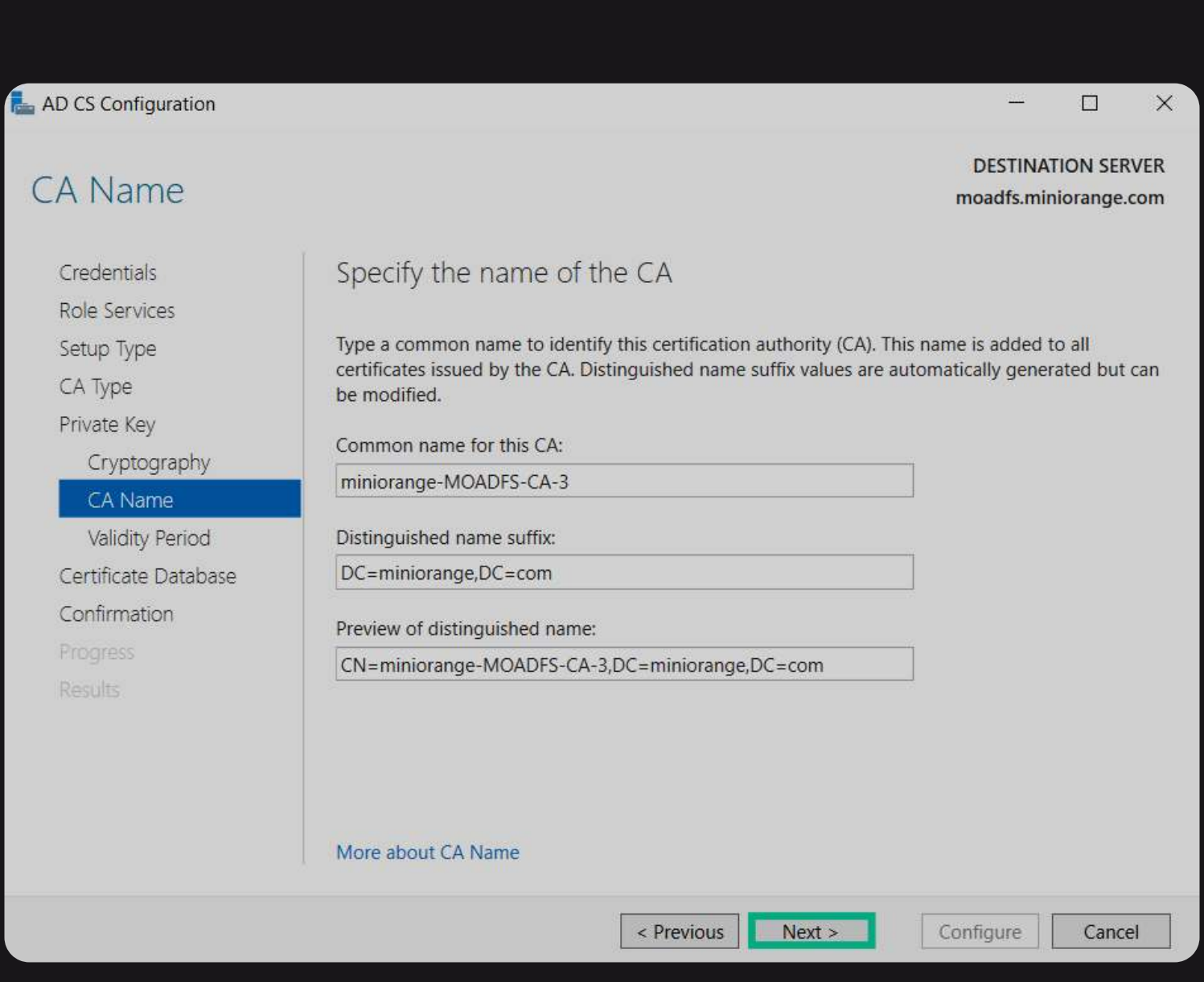
☐ Allow administrator interaction when the private key is accessed by the CA.

More about Cryptography

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

Параметры шифрования

18. Кликните ***Next***.


AD CS Configuration

CA Name

DESTINATION SERVER
moadfs.miniorange.com

Specify the name of the CA

Type a common name to identify this certification authority (CA). This name is added to all certificates issued by the CA. Distinguished name suffix values are automatically generated but can be modified.

Common name for this CA: miniorange-MOADFS-CA-3

Distinguished name suffix: DC=miniorange,DC=com

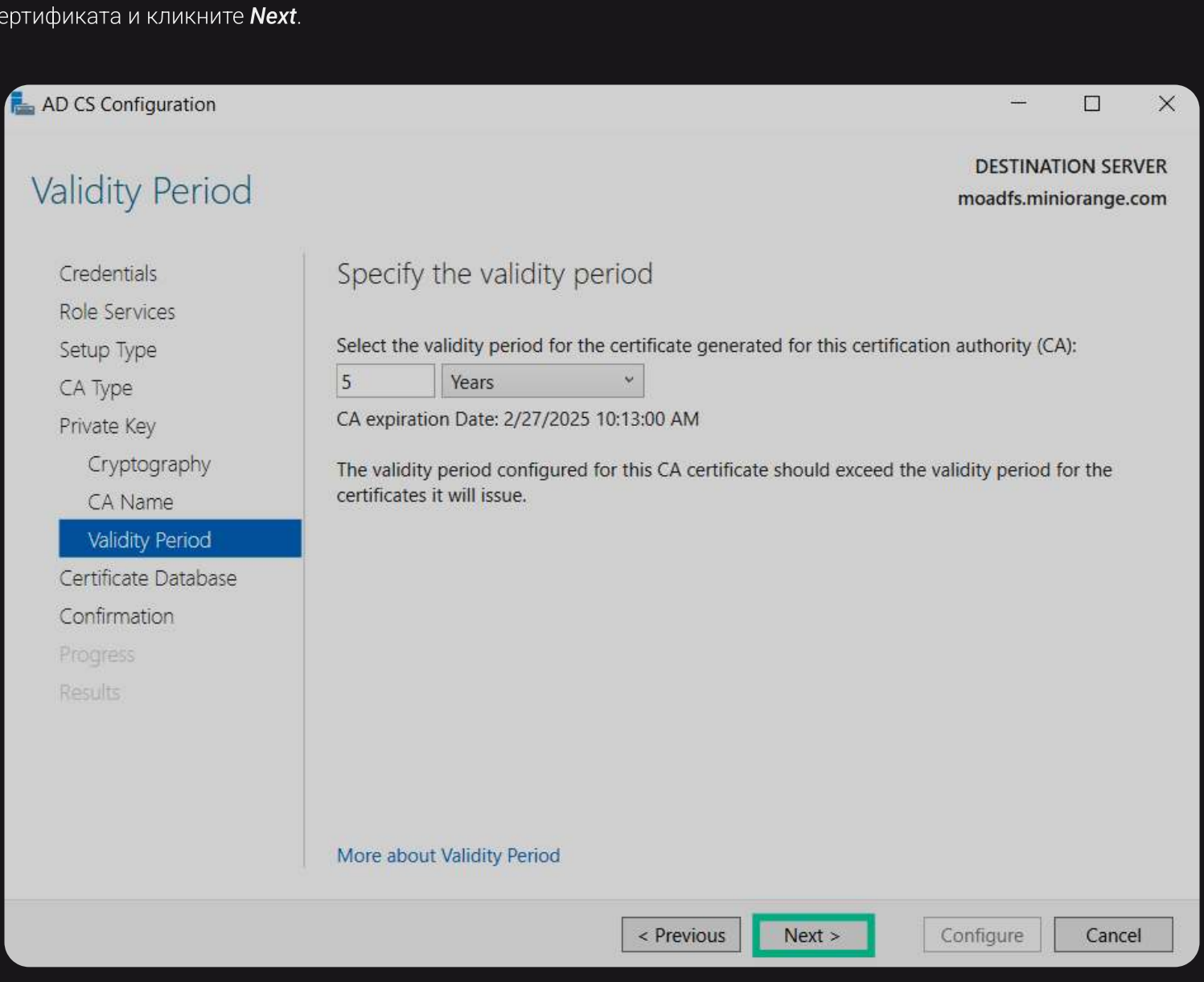
Preview of distinguished name: CN=miniorange-MOADFS-CA-3,DC=miniorange,DC=com

More about CA Name

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

Имя центра сертификации

19. Задайте срок действия сертификата и кликните ***Next***.


AD CS Configuration

Validity Period

DESTINATION SERVER
moadfs.miniorange.com

Specify the validity period

Select the validity period for the certificate generated for this certification authority (CA):

5 Years

CA expiration Date: 2/27/2025 10:13:00 AM

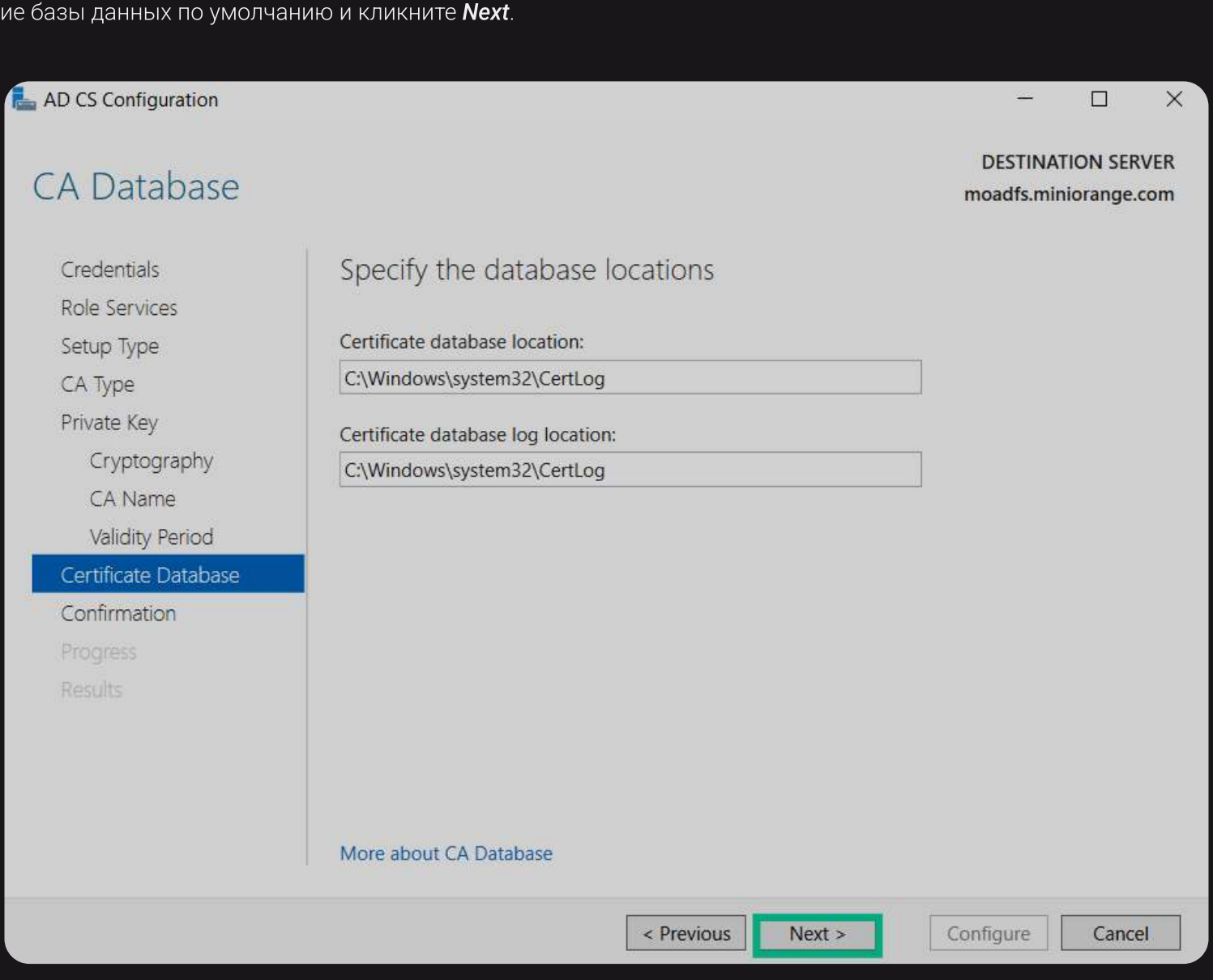
The validity period configured for this CA certificate should exceed the validity period for the certificates it will issue.

More about Validity Period

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

Срок действия

20. Выберите местоположение базы данных по умолчанию и кликните ***Next***.


AD CS Configuration

CA Database

DESTINATION SERVER
moadfs.miniorange.com

Specify the database locations

Certificate database location: C:\Windows\system32\CertLog

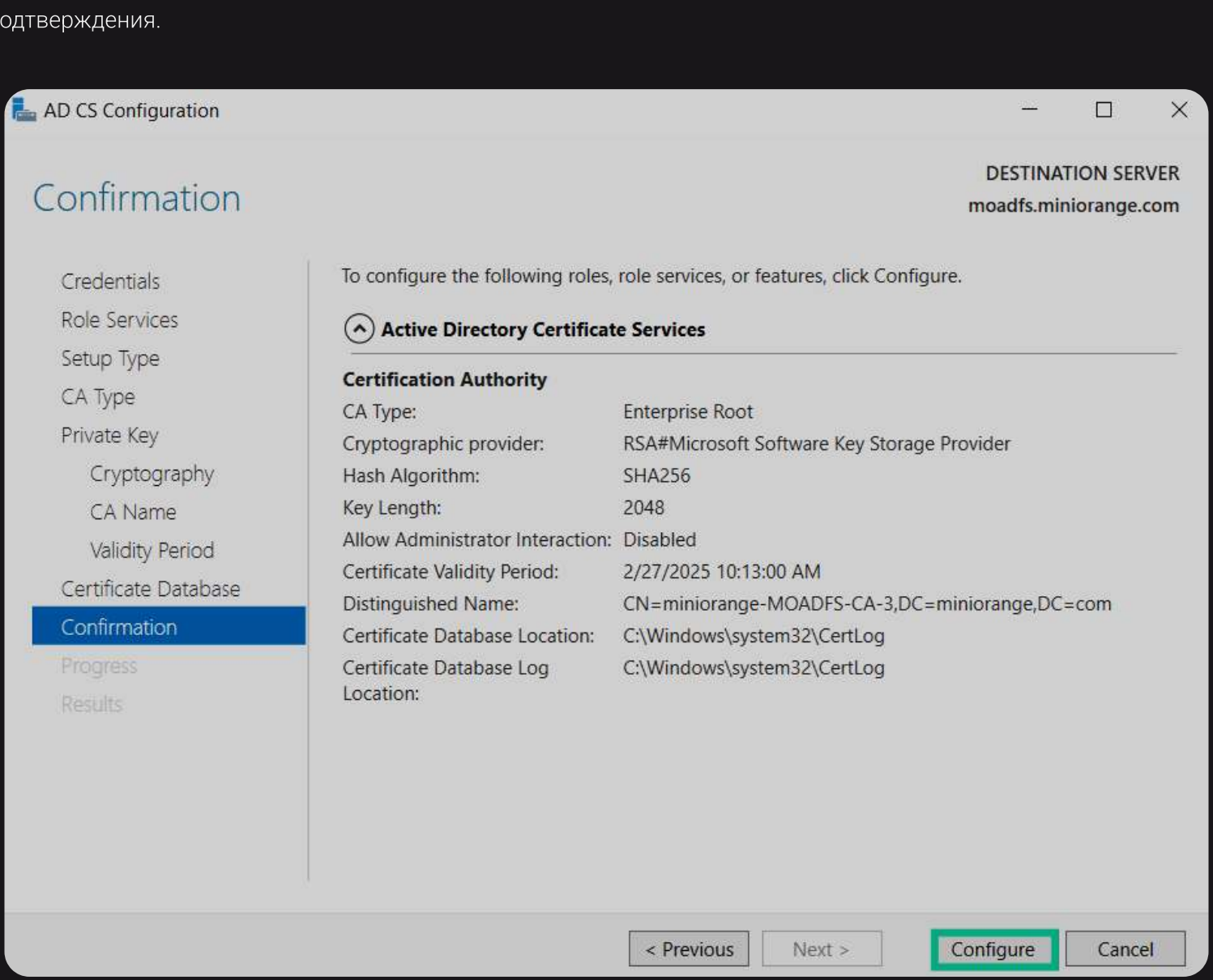
Certificate database log location: C:\Windows\system32\CertLog

More about CA Database

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

База данных сертификата

21. Кликните ***Configure*** для подтверждения.


AD CS Configuration

Confirmation

DESTINATION SERVER
moadfs.miniorange.com

To configure the following roles, role services, or features, click Configure.

Active Directory Certificate Services

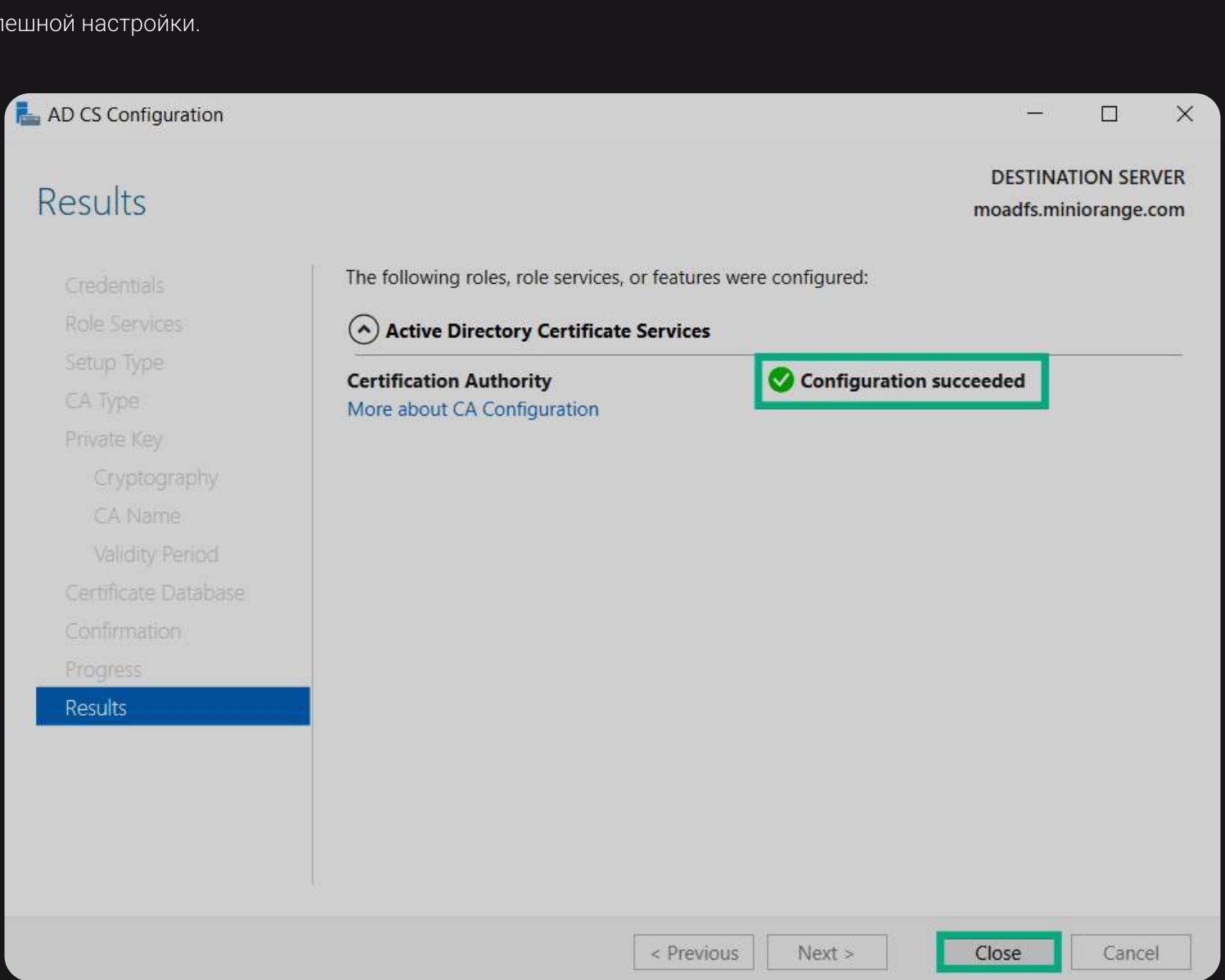
Certification Authority

CA Type:	Enterprise Root
Cryptographic provider:	RSA#Microsoft Software Key Storage Provider
Hash Algorithm:	SHA256
Key Length:	2048
Allow Administrator Interaction:	Disabled
Certificate Validity Period:	2/27/2025 10:13:00 AM
Distinguished Name:	CN=miniorange-MOADFS-CA-3,DC=miniorange,DC=com
Certificate Database Location:	C:\Windows\system32\CertLog
Certificate Database Log Location:	C:\Windows\system32\CertLog

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Configure
Cancel

Подтверждение

22. Кликните ***Close*** после успешной настройки.


AD CS Configuration

Results

DESTINATION SERVER
moadfs.miniorange.com

The following roles, role services, or features were configured:

Active Directory Certificate Services

Certification Authority — ✓ **Configuration succeeded**

More about CA Configuration

Credentials
Role Services
Setup Type
CA Type
Private Key
Cryptography
CA Name
Validity Period
Certificate Database
Confirmation
Progress
Results
< Previous
Next >
Close
Cancel

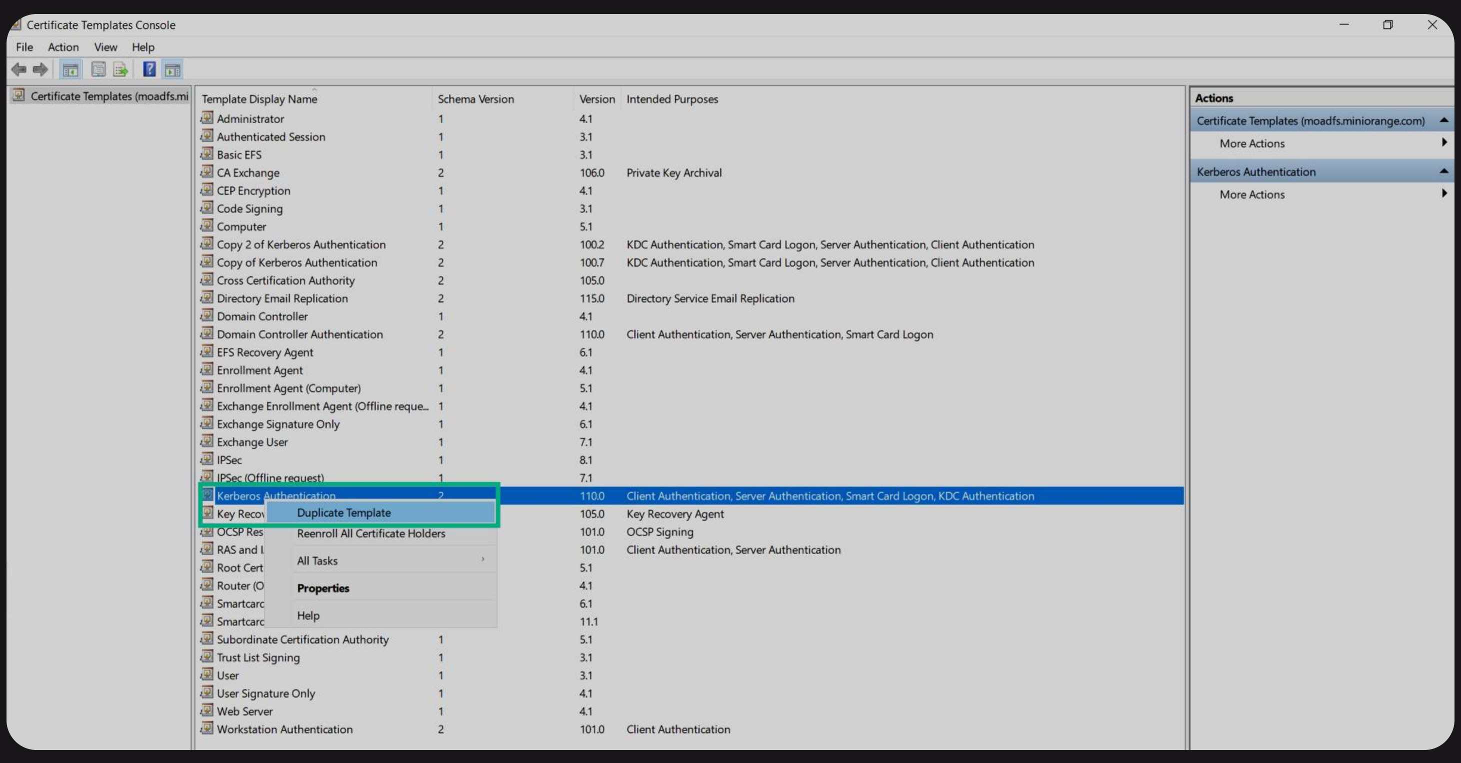
Результат настройки AD CS

Создание шаблона сертификата

Константин Алпашкин

Назад к содержанию

1. Нажмите `Win+R` и выполните команду `certtmpl.msc`.
2. Кликните правой кнопкой мыши на шаблон ***Kerberos Authentication*** и выберите ***Duplicate Template***.



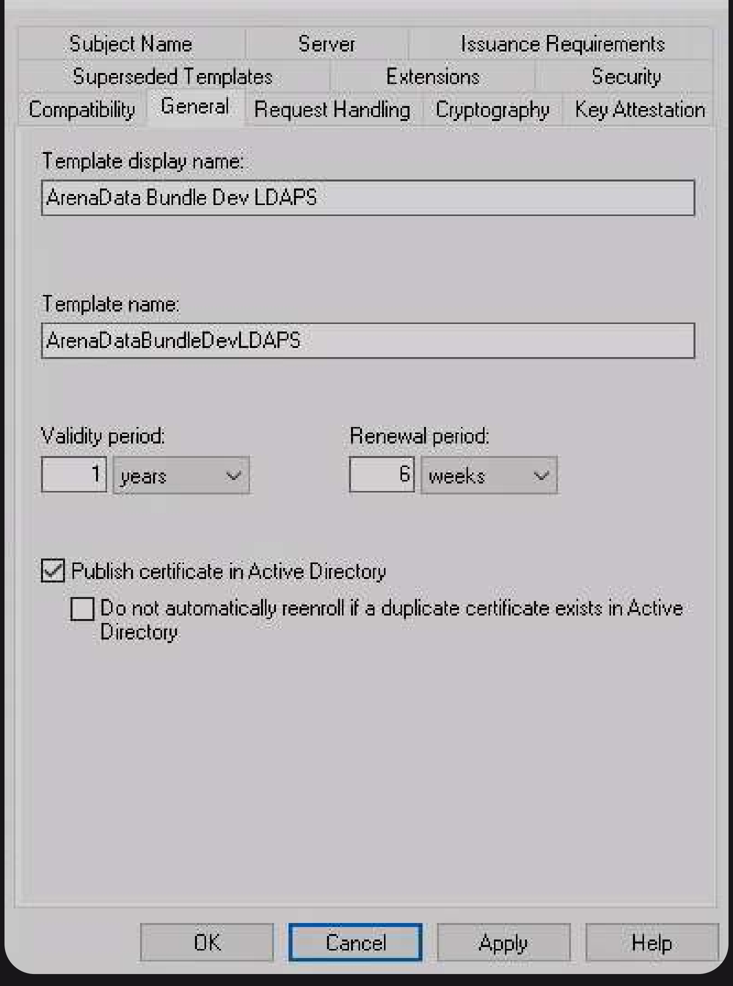
Certificate Templates Console

Template Display Name	Schema Version	Version	Intended Purposes
Administrator	1	4.1	
Authenticated Session	1	3.1	
Basic EFS	1	3.1	
CA Exchange	2	106.0	Private Key Archival
CEP Encryption	1	4.1	
Code Signing	1	3.1	
Computer	1	5.1	
Copy 2 of Kerberos Authentication	2	100.2	KDC Authentication, Smart Card Logon, Server Authentication, Client Authentication
Copy of Kerberos Authentication	2	100.7	KDC Authentication, Smart Card Logon, Server Authentication, Client Authentication
Cross Certification Authority	2	105.0	
Directory Email Replication	2	115.0	Directory Service Email Replication
Domain Controller	1	4.1	
Domain Controller Authentication	2	110.0	Client Authentication, Server Authentication, Smart Card Logon
EFS Recovery Agent	1	6.1	
Enrollment Agent	1	4.1	
Enrollment Agent (Computer)	1	5.1	
Exchange Enrollment Agent (Offline reque...	1	4.1	
Exchange Signature Only	1	6.1	
Exchange User	1	7.1	
IPSec	1	8.1	
IPSec (Offline request)	1	7.1	
Kerberos Authentication	2	110.0	Client Authentication, Server Authentication, Smart Card Logon, KDC Authentication
Key Recov		105.0	Key Recovery Agent
OCSP Res		101.0	OCSP Signing
RAS and I		101.0	Client Authentication, Server Authentication
Root Cert		5.1	
Router (O		4.1	
Smartcarc		6.1	
Smartcarc		11.1	
Subordinate Certification Authority	1	5.1	
Trust List Signing	1	3.1	
User	1	3.1	
User Signature Only	1	4.1	
Web Server	1	4.1	
Workstation Authentication	2	101.0	Client Authentication

File
Action
View
Help
Certificate Templates (moadfs.mi
Duplicate Template
Reenroll All Certificate Holders
All Tasks
Properties
Help
Actions
Certificate Templates (moadfs.miniorange.com)
More Actions
Kerberos Authentication
More Actions

Дублирование шаблона

3. В свойствах шаблона перейдите на вкладку ***General***, переименуйте отображаемое имя шаблона и включите опцию ***Publish certificate in Active Directory***.



Properties of New Template

Template display name: ArenaData Bundle Dev LDAPS

Template name: ArenaDataBundleDevLDAPS

Validity period: 1 years

Renewal period: 6 weeks

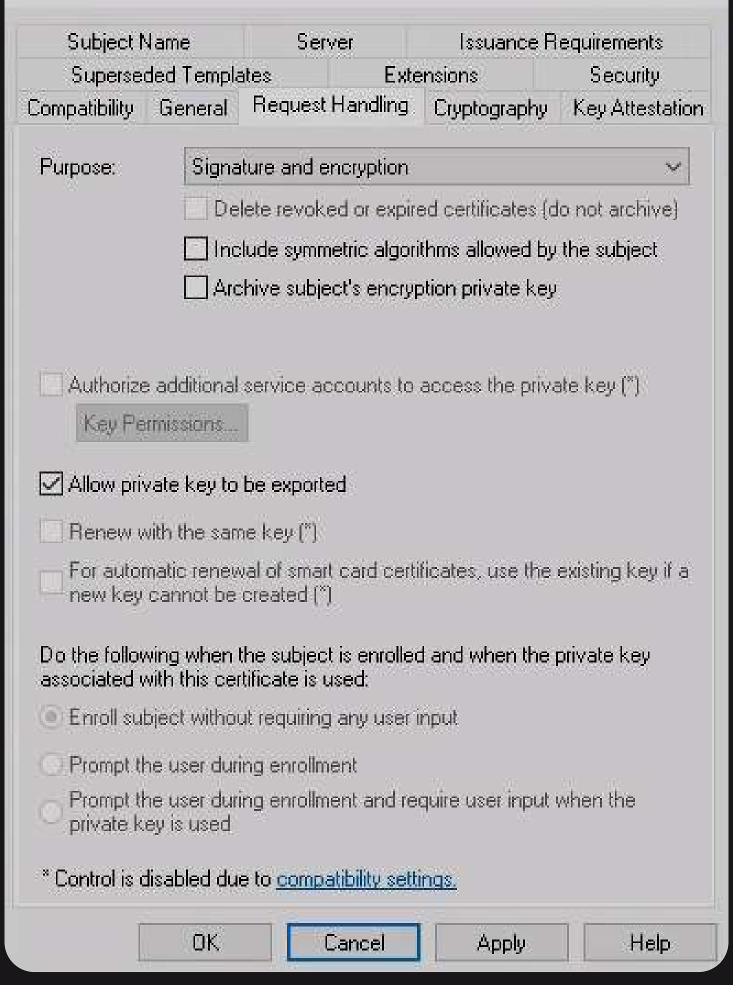
☑ Publish certificate in Active Directory

☐ Do not automatically reenroll if a duplicate certificate exists in Active Directory

Subject Name
Server
Issuance Requirements
Superseded Templates
Extensions
Security
Compatibility
General
Request Handling
Cryptography
Key Attestation
OK
Cancel
Apply
Help

Свойства нового шаблона

4. Перейдите на вкладку ***Request Handling*** и включите опцию ***Allow private key to be exported***.



Properties of New Template

Purpose: Signature and encryption

☐ Delete revoked or expired certificates (do not archive)

☐ Include symmetric algorithms allowed by the subject

☐ Archive subject's encryption private key

☐ Authorize additional service accounts to access the private key (*)

☑ Allow private key to be exported

☐ Renew with the same key (*)

☐ For automatic renewal of smart card certificates, use the existing key if a new key cannot be created (*)

Do the following when the subject is enrolled and when the private key associated with this certificate is used:

● Enroll subject without requiring any user input

○ Prompt the user during enrollment

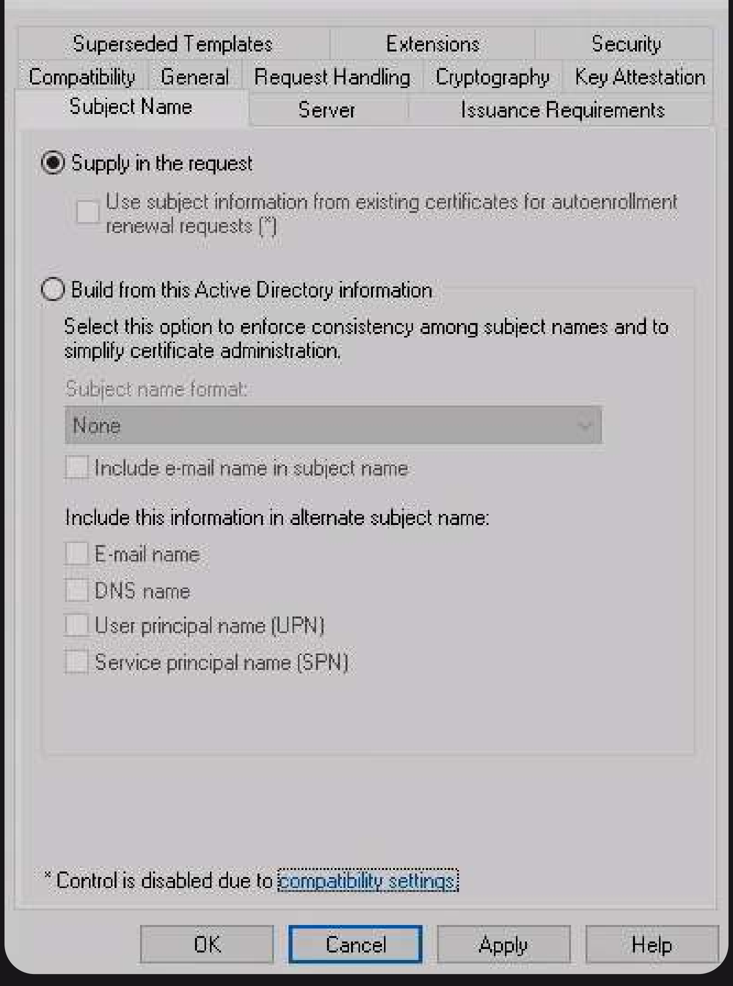
○ Prompt the user during enrollment and require user input when the private key is used

* Control is disabled due to compatibility settings.

Subject Name
Server
Issuance Requirements
Superseded Templates
Extensions
Security
Compatibility
General
Request Handling
Cryptography
Key Attestation
Key Permissions...
OK
Cancel
Apply
Help

Разрешение экспорта приватного ключа

5. Перейдите на вкладку ***Subject Name*** и выберите опцию ***Supply in the request***.



Properties of New Template

● Supply in the request

☐ Use subject information from existing certificates for autoenrollment renewal requests (*)

○ Build from this Active Directory information

Select this option to enforce consistency among subject names and to simplify certificate administration.

Subject name format: None

☐ Include e-mail name in subject name

Include this information in alternate subject name:

☐ E-mail name

☐ DNS name

☐ User principal name (UPN)

☐ Service principal name (SPN)

* Control is disabled due to compatibility settings.

Superseded Templates
Extensions
Security
Compatibility
General
Request Handling
Cryptography
Key Attestation
Subject Name
Server
Issuance Requirements
OK
Cancel
Apply
Help

Выбор имени субъекта

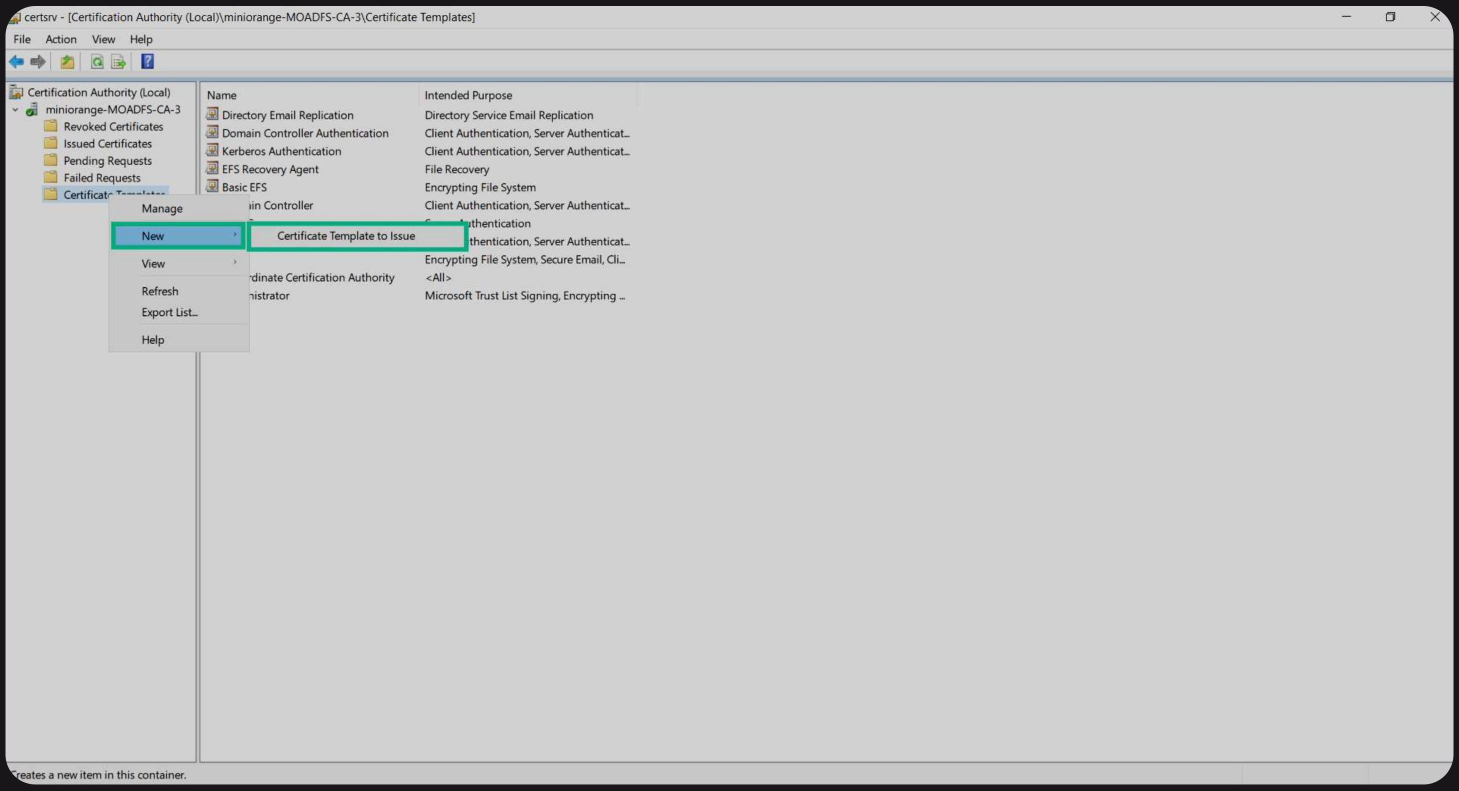
6. Кликните ***Apply***, а затем ***OK***.

Создание нового шаблона сертификата

Константин Алпашкин

Назад к содержанию

1. Перейдите в ***Start*** → ***Certification Authority***.
2. Кликните правой кнопкой ***Certificate Templates*** и выберите ***New*** → ***Certificate Template to Issue***.

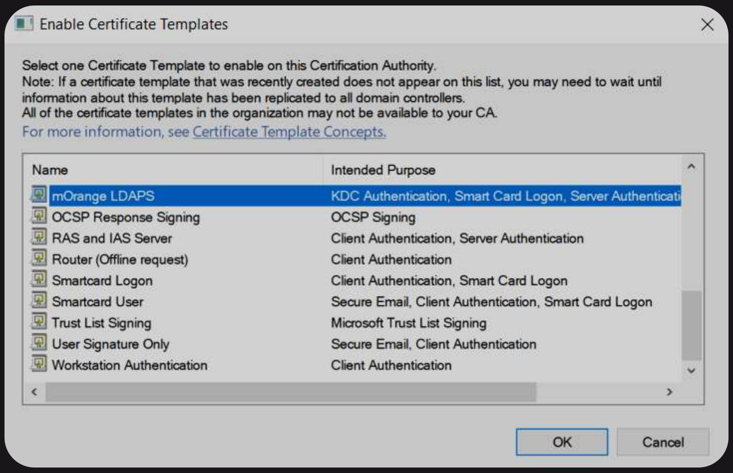

certsrv - [Certification Authority (Local)\miniorange-MOADFS-CA-3\Certificate Templates]
File
Action
View
Help
Certification Authority (Local)
miniorange-MOADFS-CA-3
Revoked Certificates
Issued Certificates
Pending Requests
Failed Requests
Certificate Templates
Manage
New
Certificate Template to Issue
View
Refresh
Export List...
Help

Name	Intended Purpose
Directory Email Replication	Directory Service Email Replication
Domain Controller Authentication	Client Authentication, Server Authenticat...
Kerberos Authentication	Client Authentication, Server Authenticat...
EFS Recovery Agent	File Recovery
Basic EFS	Encrypting File System
in Controller	Client Authentication, Server Authenticat...
	uthentication
	thentication, Server Authenticat...
	Encrypting File System, Secure Email, Cli...
rdinate Certification Authority	<All>
nistrator	Microsoft Trust List Signing, Encrypting ...

reates a new item in this container.

Новый шаблон сертификата о выдаче

3. Выберите шаблон сертификата, который вы только что создали, и кликните ***OK***


Enable Certificate Templates
OK
Cancel

Select one Certificate Template to enable on this Certification Authority.
Note: If a certificate template that was recently created does not appear on this list, you may need to wait until information about this template has been replicated to all domain controllers.
All of the certificate templates in the organization may not be available to your CA.
For more information, see Certificate Template Concepts.

Name	Intended Purpose
mOrange LDAPS	KDC Authentication, Smart Card Logon, Server Authenticati
OCSP Response Signing	OCSP Signing
RAS and IAS Server	Client Authentication, Server Authentication
Router (Offline request)	Client Authentication
Smartcard Logon	Client Authentication, Smart Card Logon
Smartcard User	Secure Email, Client Authentication, Smart Card Logon
Trust List Signing	Microsoft Trust List Signing
User Signature Only	Secure Email, Client Authentication
Workstation Authentication	Client Authentication

Выбор сертификата

Запрос нового сертификата

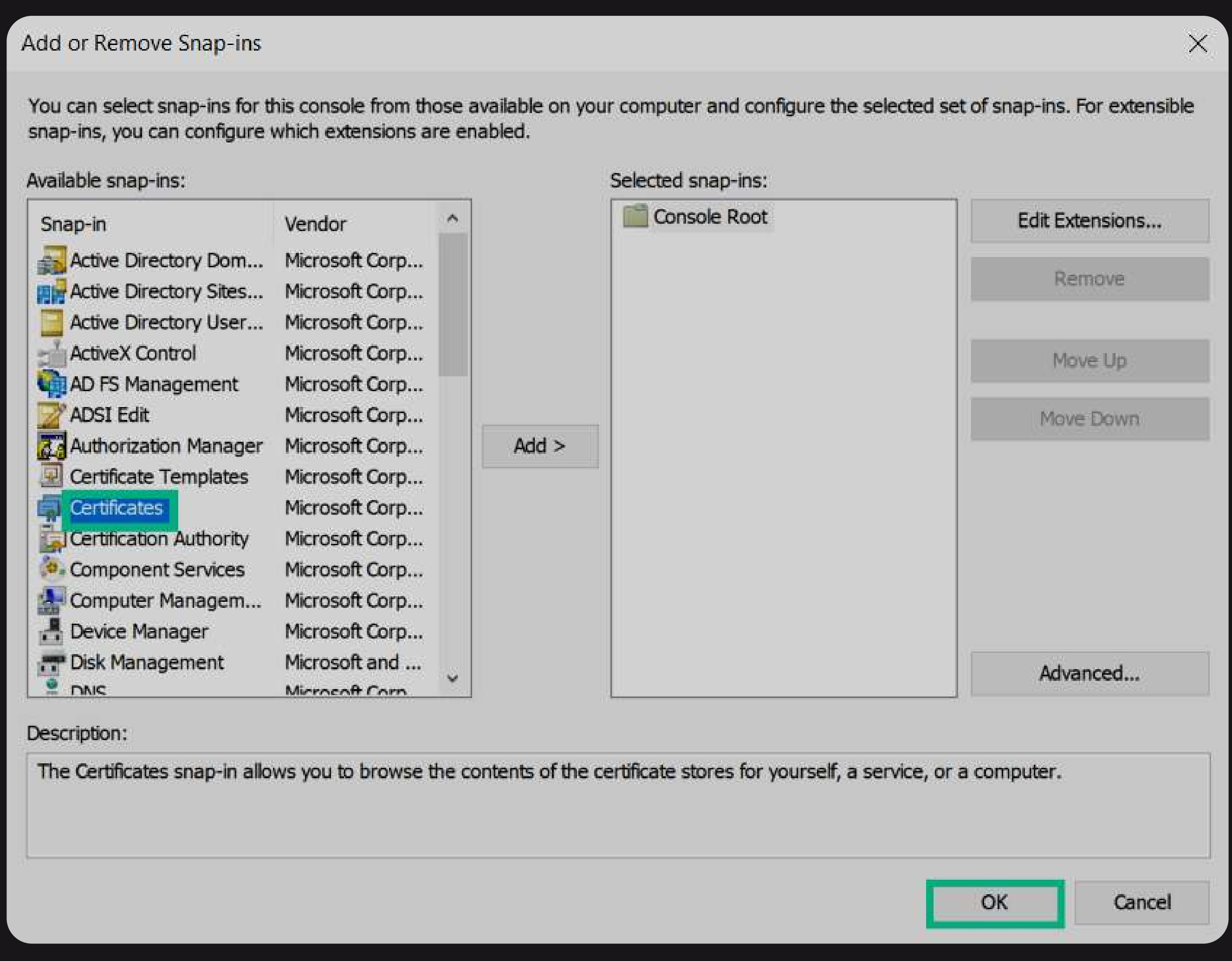
Константин Алпашкин

Назад к содержанию

1. Нажмите `Win+R` и выполните команду `mmc`.

2. Кликните ***File*** → ***Add or Remove Snap-ins***.

3. Выберите ***Certificates*** и кликните ***Add***, затем кликните ***OK***.



Add or Remove Snap-ins

You can select snap-ins for this console from those available on your computer and configure the selected set of snap-ins. For extensible snap-ins, you can configure which extensions are enabled.

Available snap-ins:

Snap-in	Vendor
Active Directory Dom...	Microsoft Corp...
Active Directory Sites...	Microsoft Corp...
Active Directory User...	Microsoft Corp...
ActiveX Control	Microsoft Corp...
AD FS Management	Microsoft Corp...
ADSI Edit	Microsoft Corp...
Authorization Manager	Microsoft Corp...
Certificate Templates	Microsoft Corp...
Certificates	Microsoft Corp...
Certification Authority	Microsoft Corp...
Component Services	Microsoft Corp...
Computer Managem...	Microsoft Corp...
Device Manager	Microsoft Corp...
Disk Management	Microsoft and ...

Selected snap-ins: Console Root

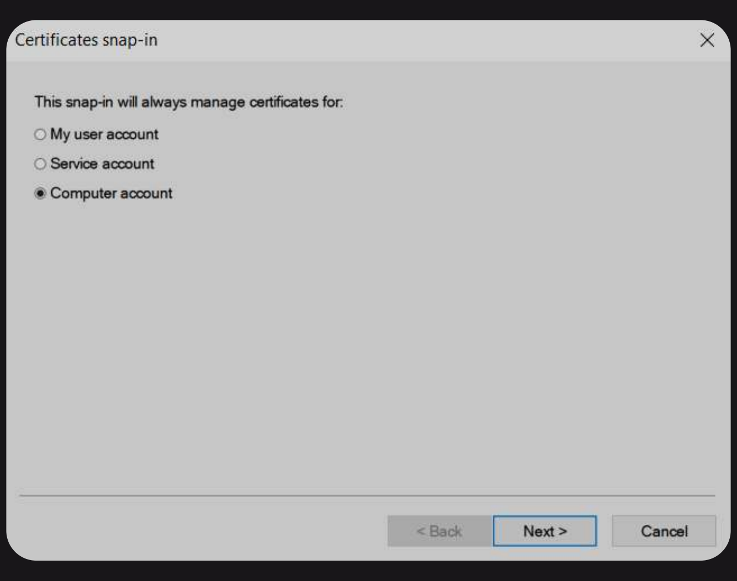
Description:

The Certificates snap-in allows you to browse the contents of the certificate stores for yourself, a service, or a computer.

Add >
Edit Extensions...
Remove
Move Up
Move Down
Advanced...
OK
Cancel

Добавление оснастки сертификатов

4. Выберите ***Computer account*** и кликните ***Next***.



Certificates snap-in

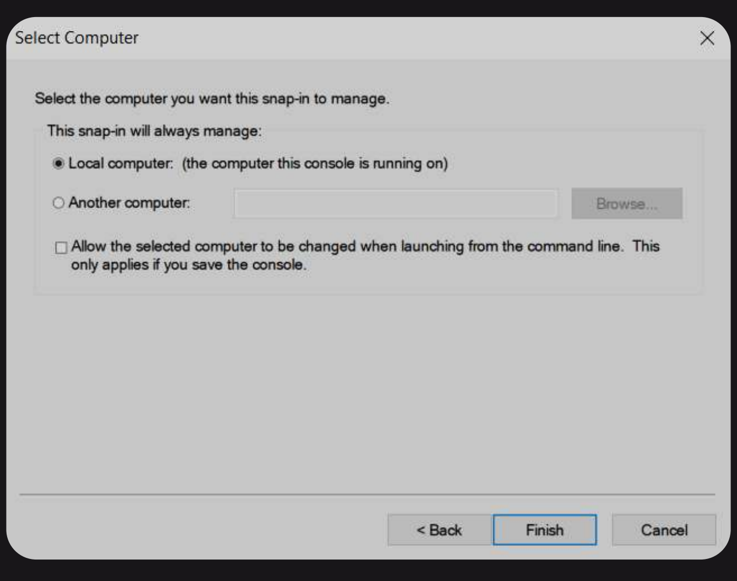
This snap-in will always manage certificates for:

○ My user account
○ Service account
● Computer account

< Back
Next >
Cancel

Тип аккаунта

5. Выберите ***Local computer*** и кликните ***Finish***.



Select Computer

Select the computer you want this snap-in to manage.

This snap-in will always manage:

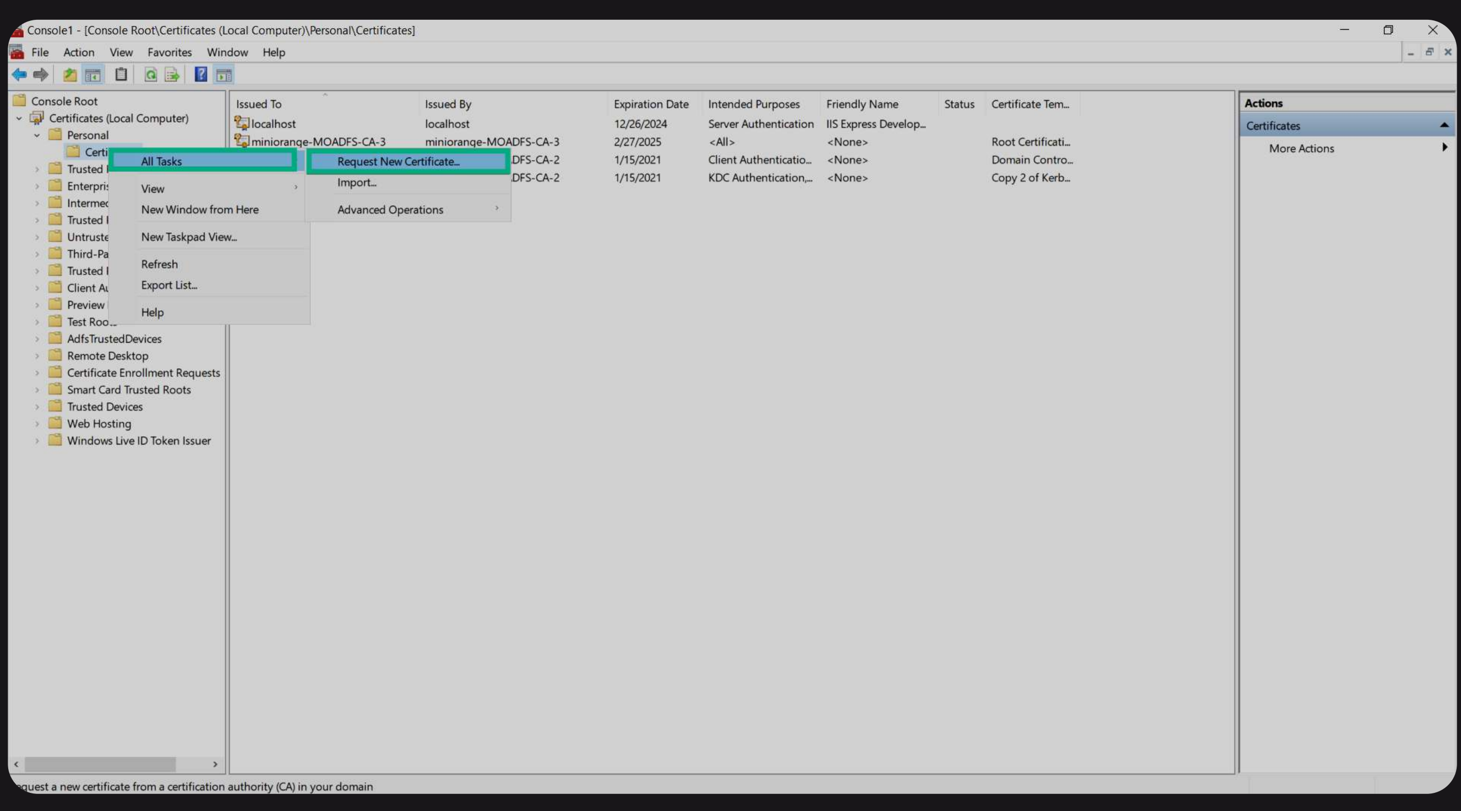
● Local computer: (the computer this console is running on)
○ Another computer:

☐ Allow the selected computer to be changed when launching from the command line. This only applies if you save the console.

Browse...
< Back
Finish
Cancel

Выбор компьютера

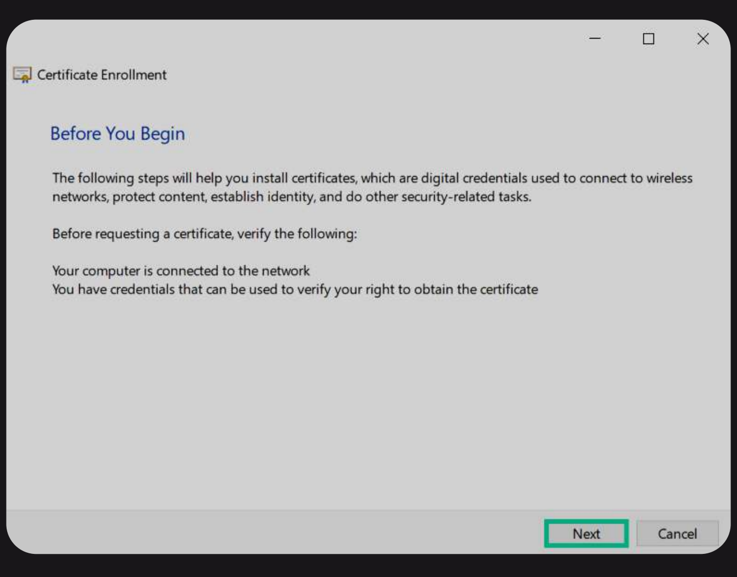
6. Кликните правой кнопкой на ***Certificates***, выберите ***All Tasks***, а затем кликните ***Request New Certificate***.



Console1 - [Console Root\Certificates (Local Computer)\Personal\Certificates]
File
Action
View
Favorites
Window
Help
Console Root
Certificates (Local Computer)
Personal
All Tasks
View
New Window from Here
New Taskpad View...
Refresh
Export List...
Help
Request New Certificate...
Import...
Advanced Operations
Issued To
Issued By
Expiration Date
Intended Purposes
Friendly Name
Status
Certificate Tem...
Actions
Certificates
More Actions
AdfsTrustedDevices
Remote Desktop
Certificate Enrollment Requests
Smart Card Trusted Roots
Trusted Devices
Web Hosting
Windows Live ID Token Issuer

Запрос нового сертификата

7. Кликните ***Next***.



Certificate Enrollment

Before You Begin

The following steps will help you install certificates, which are digital credentials used to connect to wireless networks, protect content, establish identity, and do other security-related tasks.

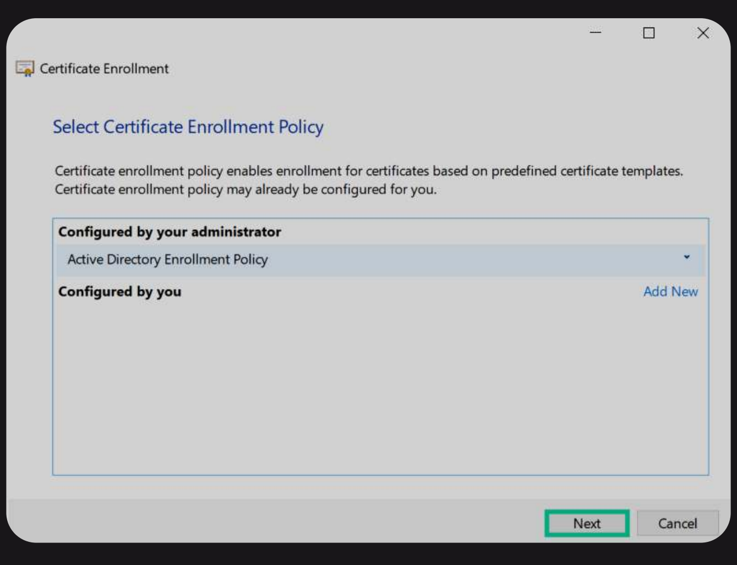
Before requesting a certificate, verify the following:

Your computer is connected to the network
You have credentials that can be used to verify your right to obtain the certificate

Next
Cancel

Регистрация сертификатов

8. Кликните ***Next***.



Certificate Enrollment

Select Certificate Enrollment Policy

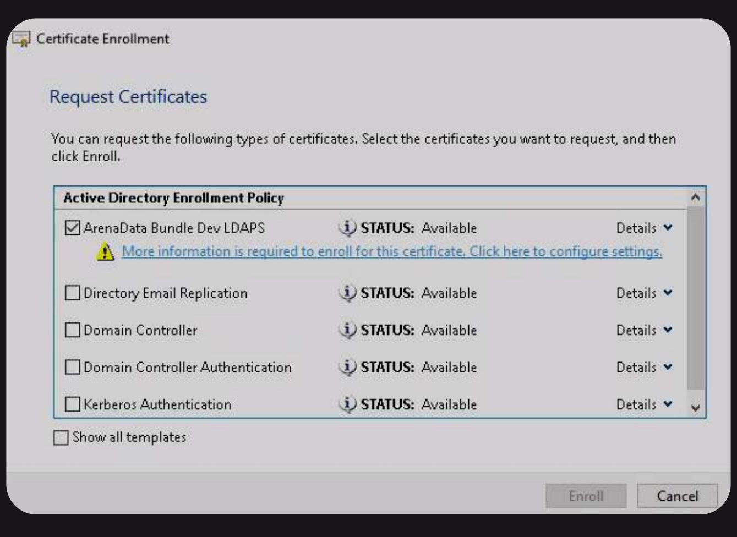
Certificate enrollment policy enables enrollment for certificates based on predefined certificate templates. Certificate enrollment policy may already be configured for you.

Configured by your administrator
Active Directory Enrollment Policy
Configured by you

Add New
Next
Cancel

Выбор политики регистрации

9. Выберите ранее созданный шаблон и кликните ***More information is required…***.



Certificate Enrollment

Request Certificates

You can request the following types of certificates. Select the certificates you want to request, and then click Enroll.

Active Directory Enrollment Policy

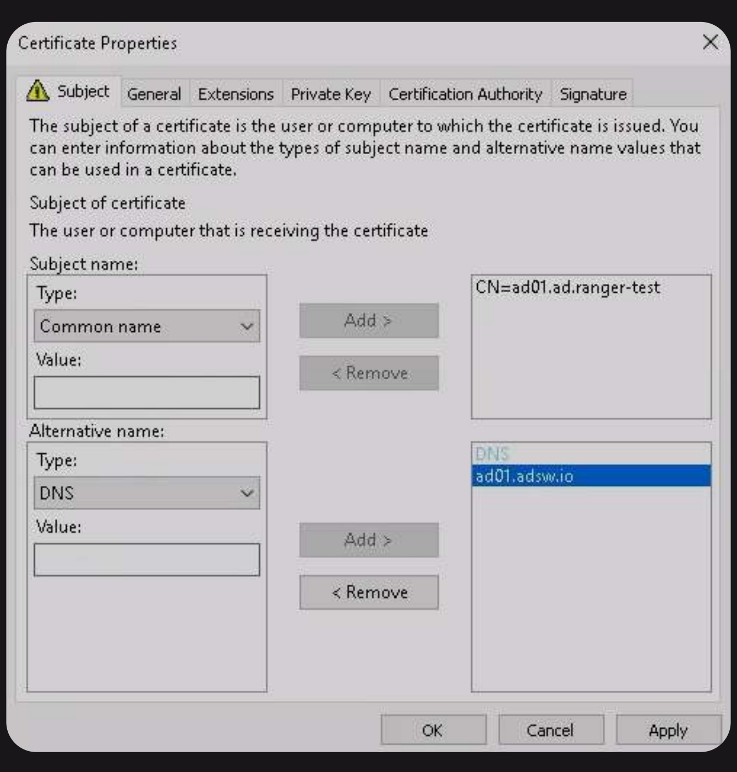
☑ ArenaData Bundle Dev LDAPS — **STATUS:** Available
More information is required to enroll for this certificate. Click here to configure settings.
☐ Directory Email Replication — **STATUS:** Available
☐ Domain Controller — **STATUS:** Available
☐ Domain Controller Authentication — **STATUS:** Available
☐ Kerberos Authentication — **STATUS:** Available

☐ Show all templates

Details
Enroll
Cancel

Дополнительная информация о сертификате

10. Заполните поля ***AD domain*** и ***DNS name***, например, `CN=ad01.ad.ranger-test` и `DNS=ad01.adsw.io`. Кликните ***Apply*** и затем ***OK***.



Certificate Properties

The subject of a certificate is the user or computer to which the certificate is issued. You can enter information about the types of subject name and alternative name values that can be used in a certificate.

Subject of certificate

The user or computer that is receiving the certificate

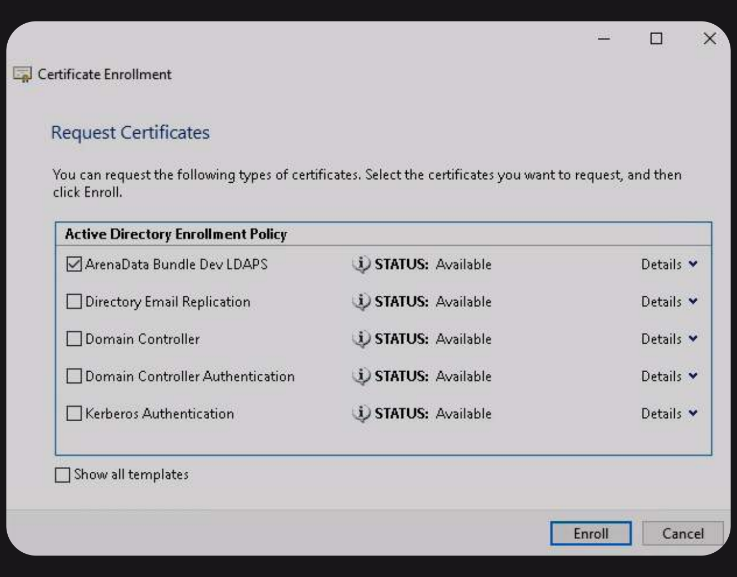
Subject name:
Type: Common name
CN=ad01.ad.ranger-test

Alternative name:
Type: DNS
DNS
ad01.adsw.io

Subject
General
Extensions
Private Key
Certification Authority
Signature
Value:
Add >
< Remove
OK
Cancel
Apply

Свойства сертификата

11. Кликните ***Enroll***.



Certificate Enrollment

Request Certificates

You can request the following types of certificates. Select the certificates you want to request, and then click Enroll.

Active Directory Enrollment Policy

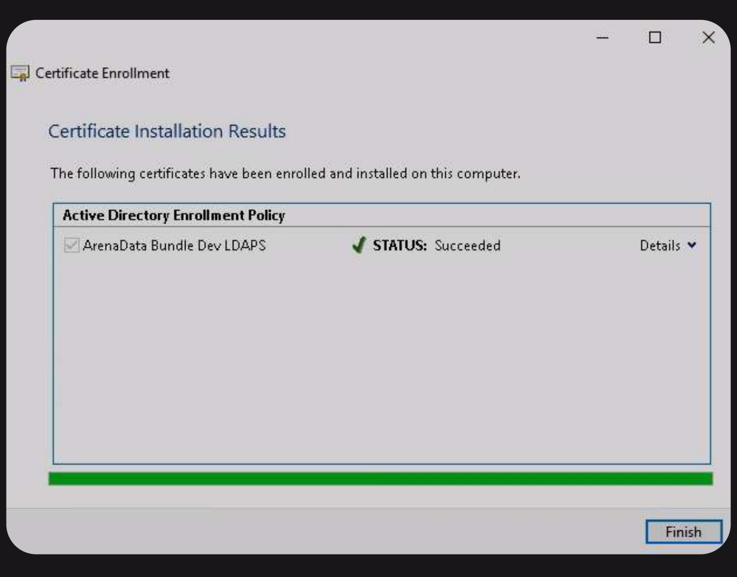
☑ ArenaData Bundle Dev LDAPS — **STATUS:** Available
☐ Directory Email Replication — **STATUS:** Available
☐ Domain Controller — **STATUS:** Available
☐ Domain Controller Authentication — **STATUS:** Available
☐ Kerberos Authentication — **STATUS:** Available

☐ Show all templates

Details
Enroll
Cancel

Регистрация сертификата

12. Кликните ***Finish***.



Certificate Enrollment

Certificate Installation Results

The following certificates have been enrolled and installed on this computer.

Active Directory Enrollment Policy

☑ ArenaData Bundle Dev LDAPS — **STATUS:** Succeeded

Details
Finish

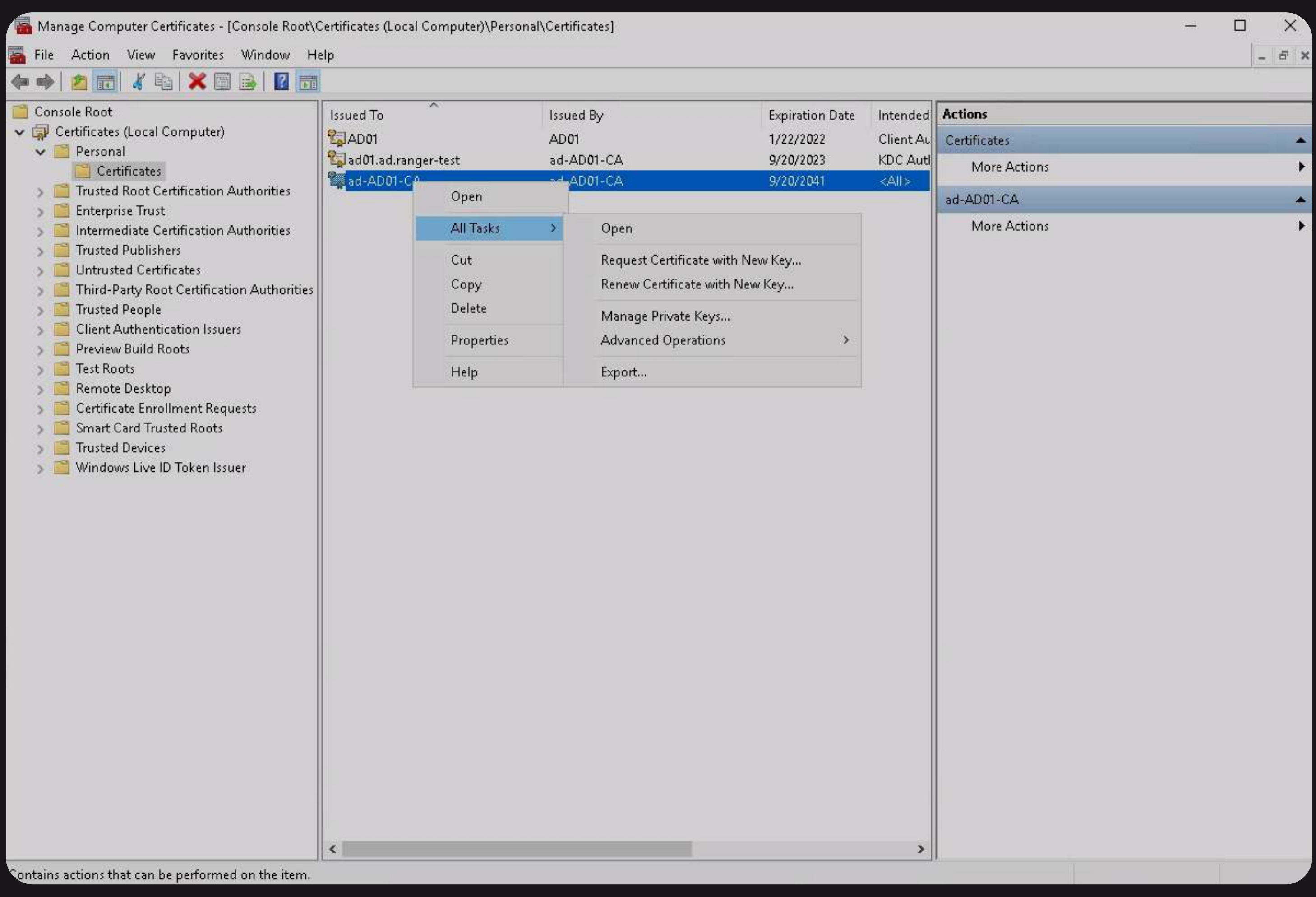
Результаты установки сертификата

Экспорт сертификата ЦС

Константин Алпашкин

Назад к содержанию

1. Откройте оснастку управления сертификатами `mmc` (`Win+R` → выполнить `mmc` → ***File*** → ***Add or Remove Snap-ins***).
2. Кликните правой кнопкой мыши на сертификате ЦС и выберите ***All Tasks*** → ***Export***.



Manage Computer Certificates - [Console Root\Certificates (Local Computer)\Personal\Certificates]

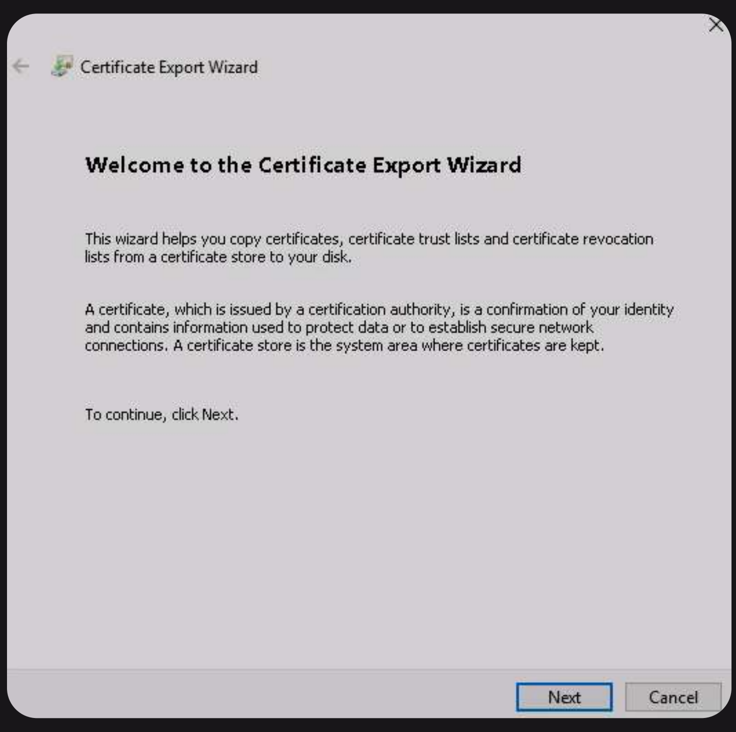
Issued To	Issued By	Expiration Date	Intended
AD01	AD01	1/22/2022	Client Au
ad01.ad.ranger-test	ad-AD01-CA	9/20/2023	KDC Autl
ad-AD01-CA	ad-AD01-CA	9/20/2041	<All>

Contains actions that can be performed on the item.

File
Action
View
Favorites
Window
Help
Console Root
Certificates (Local Computer)
Personal
Certificates
Trusted Root Certification Authorities
Enterprise Trust
Intermediate Certification Authorities
Trusted Publishers
Untrusted Certificates
Third-Party Root Certification Authorities
Trusted People
Client Authentication Issuers
Preview Build Roots
Test Roots
Remote Desktop
Certificate Enrollment Requests
Smart Card Trusted Roots
Trusted Devices
Windows Live ID Token Issuer
Open
All Tasks
Cut
Copy
Delete
Properties
Help
Open
Request Certificate with New Key...
Renew Certificate with New Key...
Manage Private Keys...
Advanced Operations
Export...
Actions
Certificates
More Actions
ad-AD01-CA
More Actions

Экспорт сертификата

3. Кликните ***Next***.



Certificate Export Wizard

Welcome to the Certificate Export Wizard

This wizard helps you copy certificates, certificate trust lists and certificate revocation lists from a certificate store to your disk.

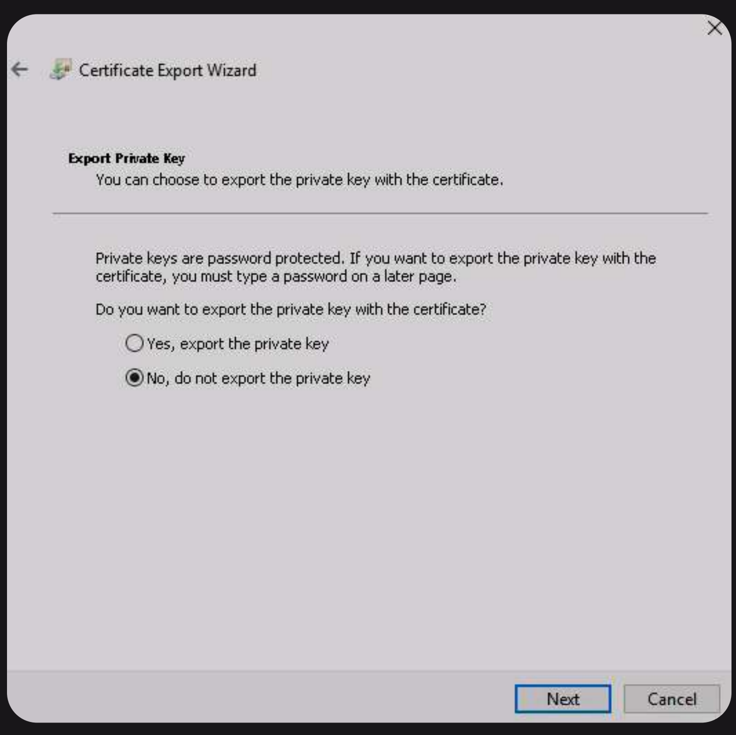
A certificate, which is issued by a certification authority, is a confirmation of your identity and contains information used to protect data or to establish secure network connections. A certificate store is the system area where certificates are kept.

To continue, click Next.

Next
Cancel

Мастер экспорта сертификатов

4. Выберите ***No, don't export the private key***, затем кликните ***Next***.



Certificate Export Wizard

Export Private Key
You can choose to export the private key with the certificate.

Private keys are password protected. If you want to export the private key with the certificate, you must type a password on a later page.

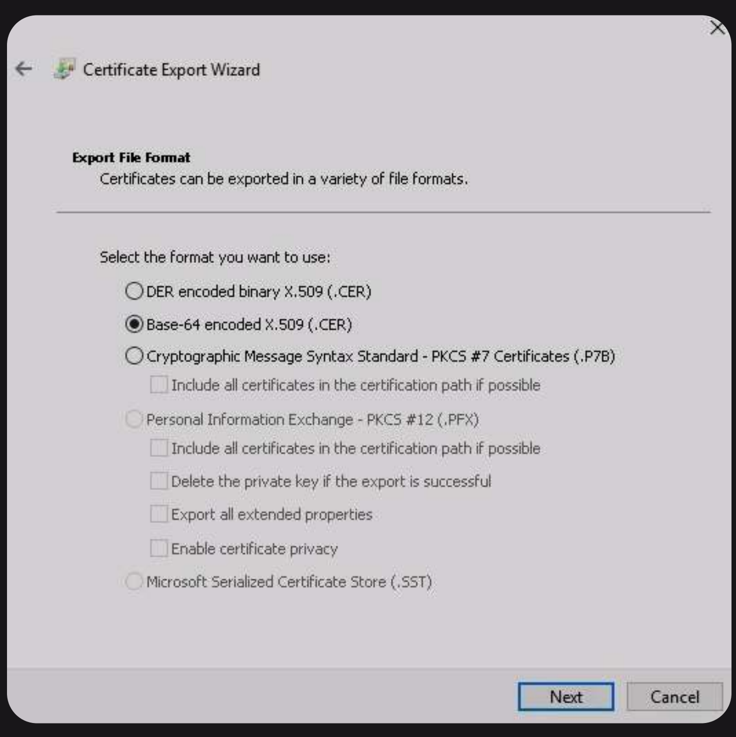
Do you want to export the private key with the certificate?

○ Yes, export the private key
● No, do not export the private key

Next
Cancel

Экспорт приватного ключа

5. Выберите формат ***Base-64 encoded X.509***, а затем кликните ***Next***.



Certificate Export Wizard

Export File Format
Certificates can be exported in a variety of file formats.

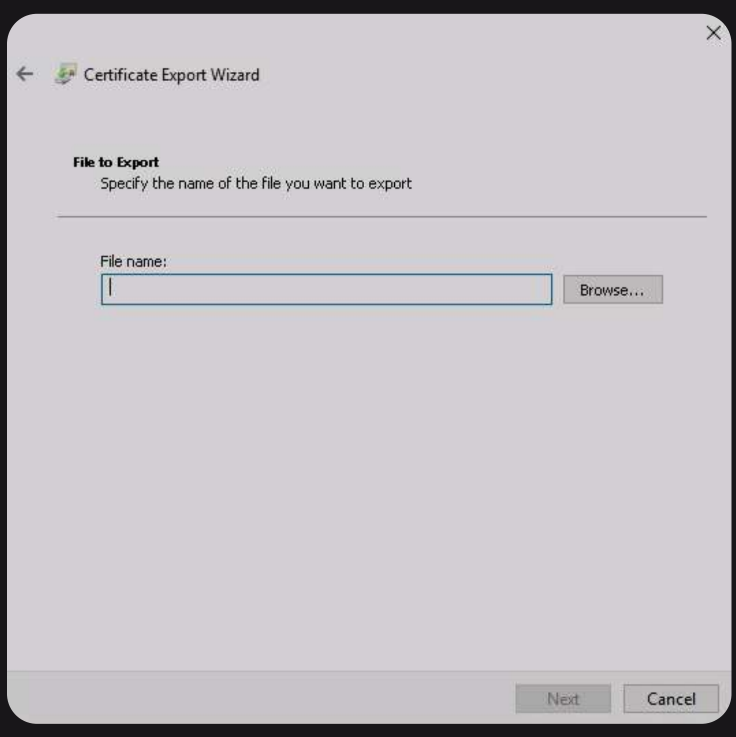
Select the format you want to use:

○ DER encoded binary X.509 (.CER)
● Base-64 encoded X.509 (.CER)
○ Cryptographic Message Syntax Standard - PKCS #7 Certificates (.P7B)
  ☐ Include all certificates in the certification path if possible
○ Personal Information Exchange - PKCS #12 (.PFX)
  ☐ Include all certificates in the certification path if possible
  ☐ Delete the private key if the export is successful
  ☐ Export all extended properties
  ☐ Enable certificate privacy
○ Microsoft Serialized Certificate Store (.SST)

Next
Cancel

Формат экспортируемого файла

6. Введите имя файла для вашего сертификата, а затем кликните ***Next***.



Certificate Export Wizard

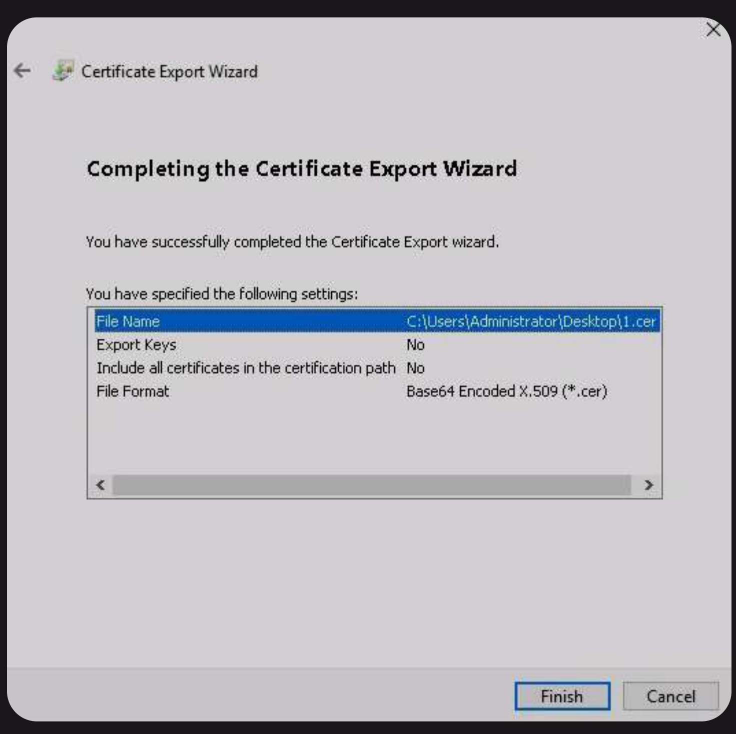
File to Export
Specify the name of the file you want to export

File name:

Browse...
Next
Cancel

Имя файла для экспорта

7. Кликните ***Finish***.



Certificate Export Wizard

Completing the Certificate Export Wizard

You have successfully completed the Certificate Export wizard.

You have specified the following settings:

File Name	C:\Users\Administrator\Desktop\1.cer
Export Keys	No
Include all certificates in the certification path	No
File Format	Base64 Encoded X.509 (*.cer)

Finish
Cancel

Завершение экспорта сертификата

Настройка использования Active Directory для Kerberos-аутентификации с помощью ADCM

Константин Алпашкин, Сергей Тихомиров

Содержание

Назад к содержанию

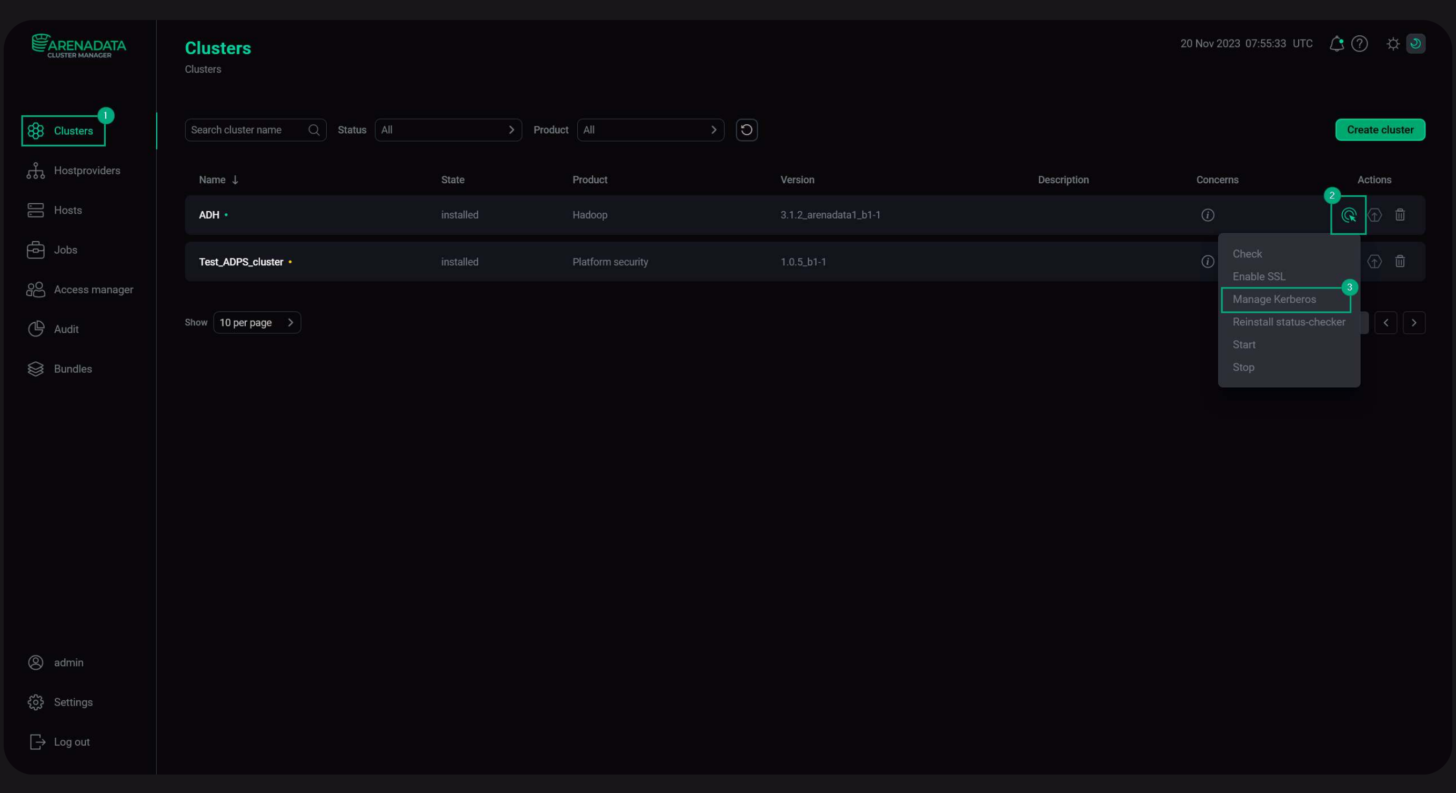
Обзор

Параметры Active Directory

Обзор

Для керберизации кластера с помощью Active Directory выполните описанные ниже шаги:

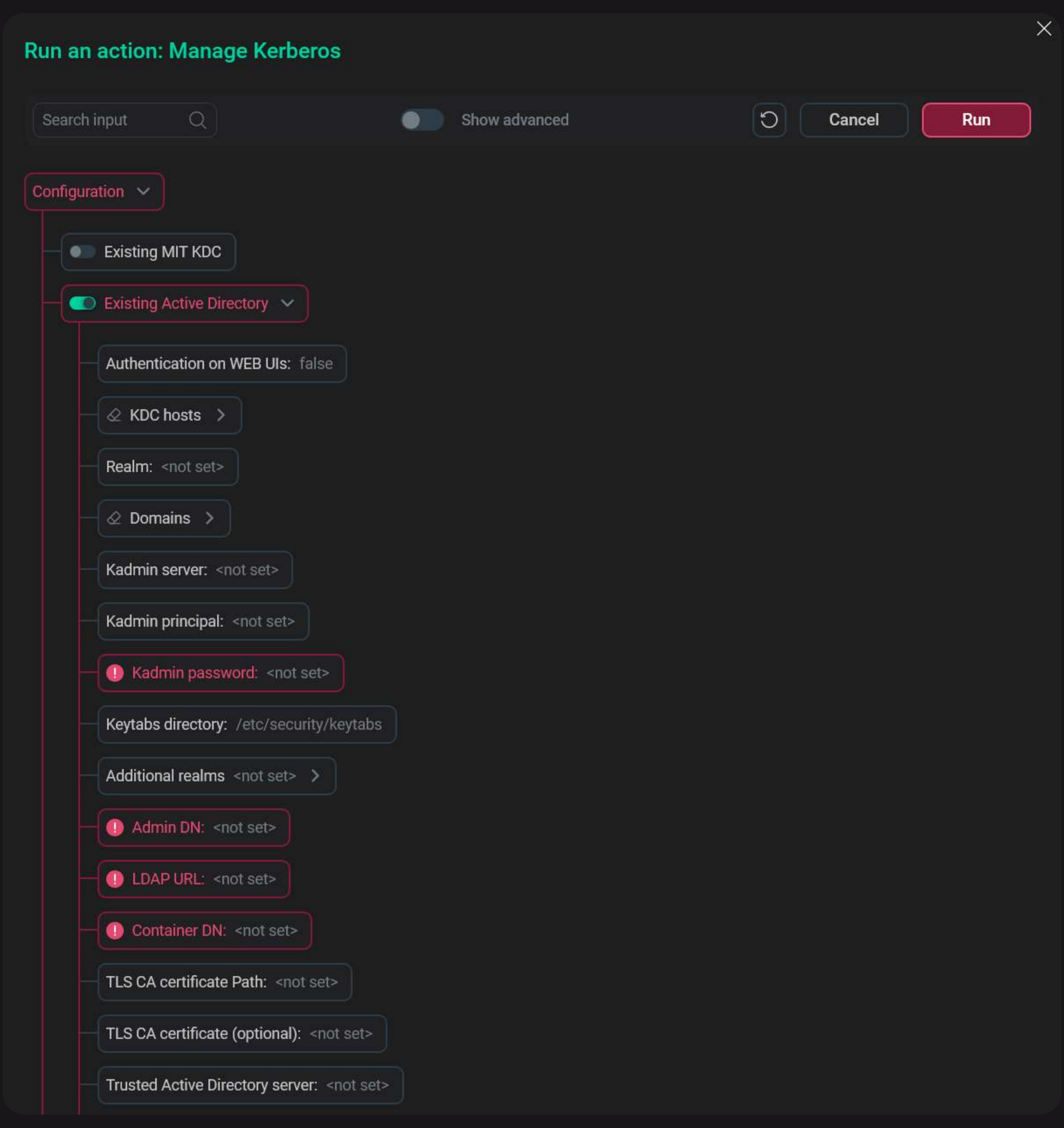
1. В веб-интерфейсе ADCM перейдите на страницу ***Clusters***. Выберите установленный и подготовленный кластер ADH, и запустите действие ***Manage Kerberos***.


ARENADATA
CLUSTER MANAGER
Clusters
Clusters
20 Nov 2023 07:55:33 UTC
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status
All
Product
All
Create cluster
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
Show
10 per page
1
2
3

Name	State	Product	Version	Description	Concerns	Actions
ADH	installed	Hadoop	3.1.2_arenadata1_b1-1			
Test_ADPS_cluster	installed	Platform security	1.0.5_b1-1			

Manage Kerberos

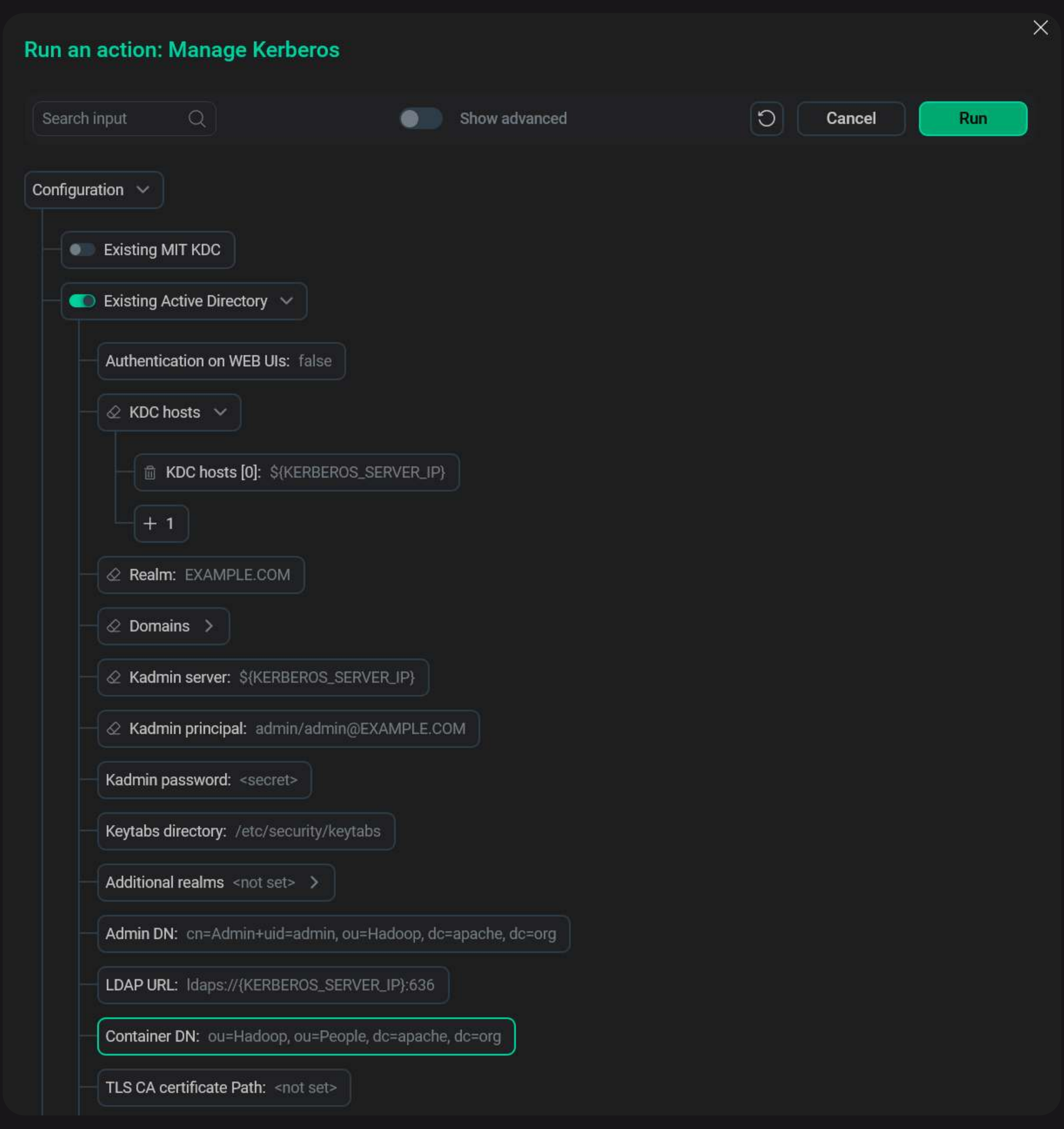
2. В появившемся окне выберите опцию ***Existing Active Directory***.


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
KDC hosts
Domains

Authentication on WEB UIs: false
Realm: <not set>
Kadmin server: <not set>
Kadmin principal: <not set>
Kadmin password: <not set>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
Admin DN: <not set>
LDAP URL: <not set>
Container DN: <not set>
TLS CA certificate Path: <not set>
TLS CA certificate (optional): <not set>
Trusted Active Directory server: <not set>

Выбор соответствующей опции

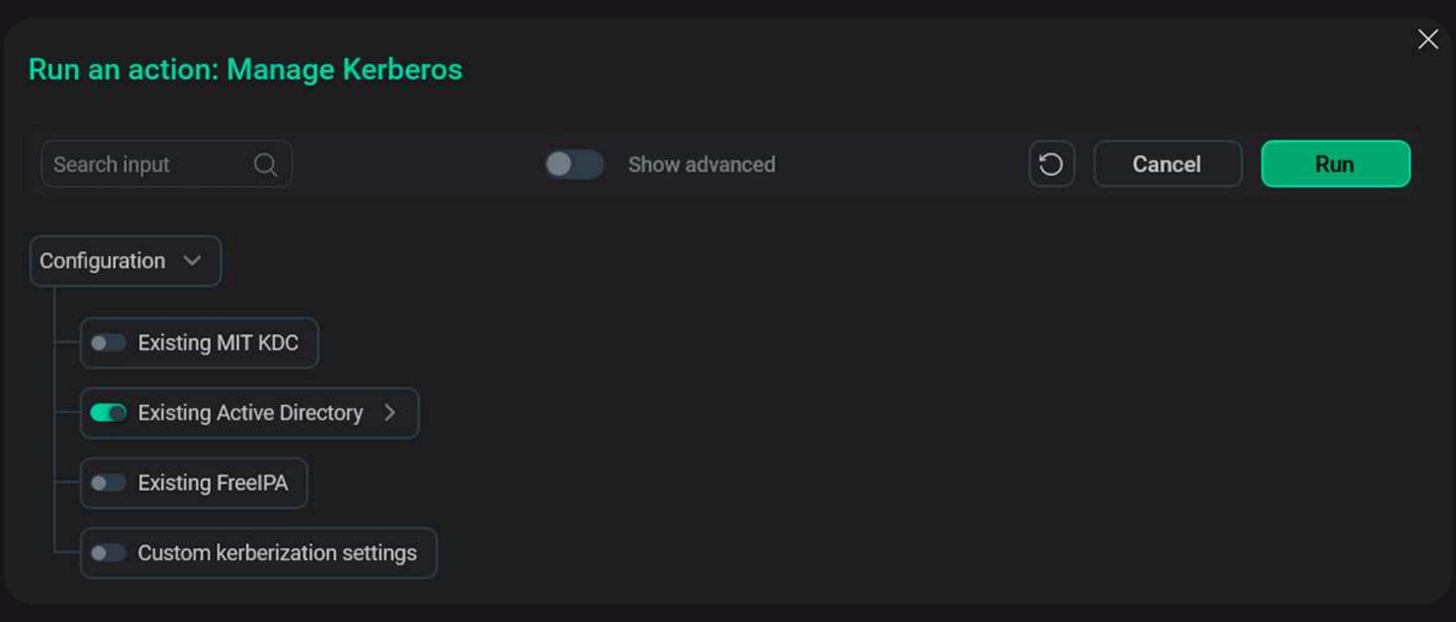
3. Заполните параметры Active Directory.


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
KDC hosts
+ 1
Domains

Authentication on WEB UIs: false
KDC hosts [0]: ${KERBEROS_SERVER_IP}
Realm: EXAMPLE.COM
Kadmin server: ${KERBEROS_SERVER_IP}
Kadmin principal: admin/admin@EXAMPLE.COM
Kadmin password: <secret>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
Admin DN: cn=Admin+uid=admin, ou=Hadoop, dc=apache, dc=org
LDAP URL: ldaps://{KERBEROS_SERVER_IP}:636
Container DN: ou=Hadoop, ou=People, dc=apache, dc=org
TLS CA certificate Path: <not set>

Параметры Active Directory

4. Нажмите ***Run*** и дождитесь окончания действия. Затем приступайте к настройке Kerberos в кластере.


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Custom kerberization settings

Запуск действия

Параметры Active Directory

Параметр	Описание
Authentication on WEB UIs	Включает аутентификацию Kerberos для веб-интерфейсов
KDC hosts	Один или более MIT KDC-хостов
Realm	Kerberos realm — сеть, содержащая KDC-хосты и множество клиентов
Domains	Домены, в которых находятся хосты
Kadmin server	Адрес хоста, на котором запущен `kadmin`
Kadmin principal	Имя принципала для подключения через `kadmin`, например `admin@RU-CENTRAL1.INTERNAL`
Kadmin password	Пароль принципала для подключения через `kadmin`
Keytabs directory	Директория keytab-файла, содержащего записи одного или нескольких принципалов и их ключи
Additional realms	Дополнительные сети Kerberos realm
Admin DN	Полное отличительное имя пользователя с правами создания, редактирования, удаления, изменения пароля пользователей в целевом OU
LDAP URL	LDAP URL состоит из префикса `ldap://` или `ldaps://`, имени хоста или IP-адреса и порта AD-сервера
Container DN	Отличительное имя контейнера
TLS CA certificate Path	Путь к CA-сертификату в файловой системе хоста. Файл по данному пути должен существовать заранее, иначе он будет создан с данными из поля `TLS CA certificate (optional)`
TLS CA certificate (optional)	Текст CA-сертификата, который запишется в файл по указанному в поле `TLS CA certificate Path` пути
Trusted Active Directory server	Сервер с Active Directory для односторонней доверенности от MIT Kerberos KDC внутри realm
Trusted Active Directory realm	Realm с Active Directory для односторонней доверенности от MIT Kerberos KDC внутри realm

Керберизация с предустановленной Active Directory при отсутствии прав администратора

Сергей Тихомиров

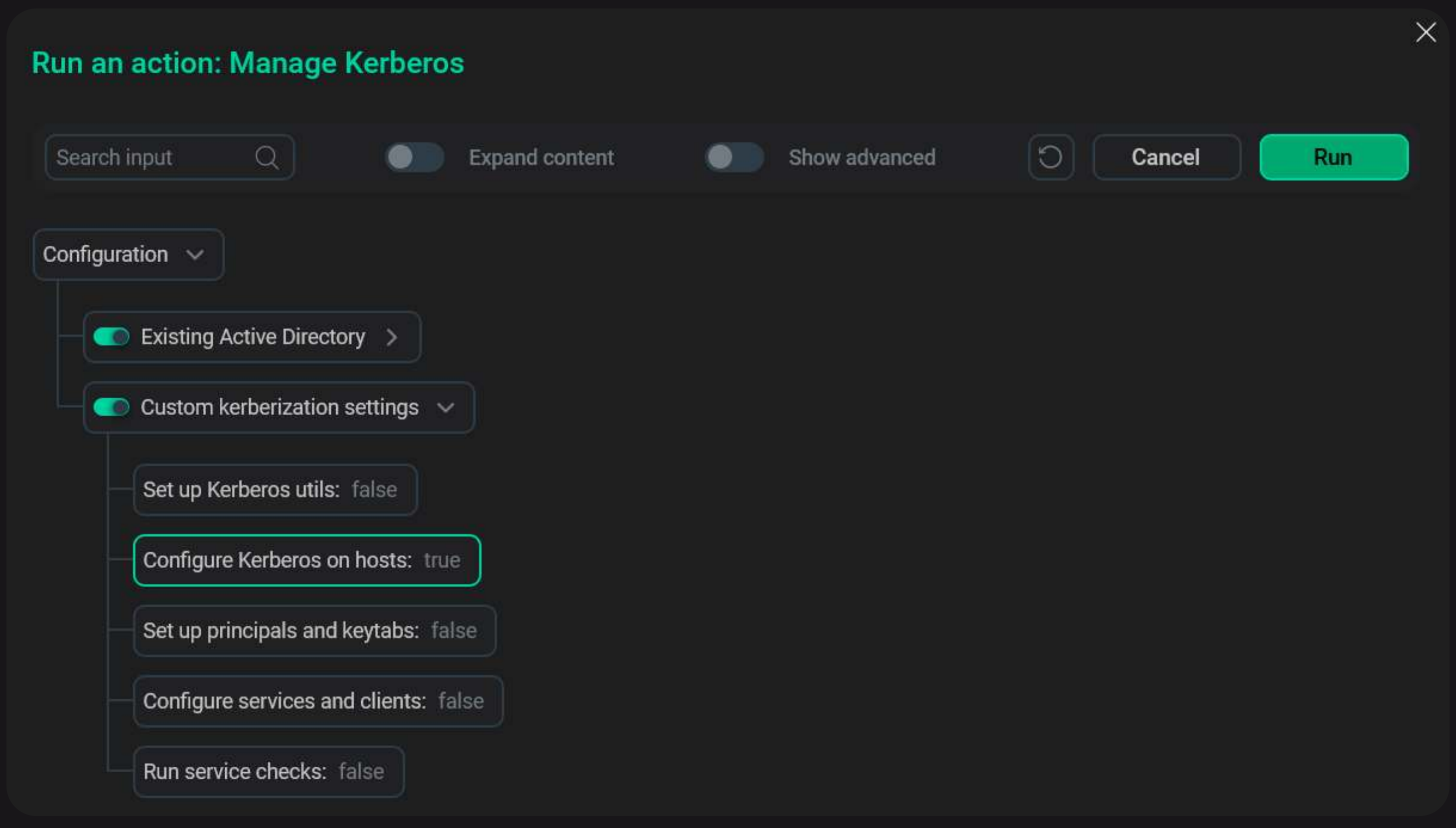
Содержание

Назад к содержанию

Подготовка хостов кластера
Активация Kerberos

Подготовка хостов кластера

1. Проверьте, что кластер ADH установлен и не имеет ошибок.
2. Добавьте файл *krb5.conf* в директорию */etc/*. Это можно сделать самостоятельно, но также можно запустить действие ***Manage Kerberos*** с выбранной опцией ***Existing Active Directory*** и установленным флагом ***Configure Kerberos on hosts*** в меню ***Custom kerberization settings***. При заполнении полей в форме ***Existing Active Directory*** указывайте данные пользователя, не являющегося администратором.



Run an action: Manage Kerberos

Configuration
- Existing Active Directory
- Custom kerberization settings
  - Set up Kerberos utils: false
  - Configure Kerberos on hosts: true
  - Set up principals and keytabs: false
  - Configure services and clients: false
  - Run service checks: false

Создание файла конфигурации

3. Добавьте принципала для каждого сервиса. В качестве списка сервисов, для которых нужно добавить принципала, используйте столбец ***Keytab-файл*** в таблице сопоставления owner-keytab.

```
$ ldapadd -H <ldap_link> -x -D <username>@<realm> -w <password> -f <ldif_file>.ldif
```

где:

- `<ldap_link>` — URL LDAP-сервера.
- `<username>` — имя пользователя LDAP.
- `<realm>` — realm пользователя.
- `<password>` — пароль для `<username>`.
- `<ldif_file>` — имя файла с информацией о принципале. Пример содержимого файла приведен ниже.

```
DN: CN=HTTP/<principal>,OU=Hadoop,DC=fresh,DC=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
accountExpires: 0
userPrincipalName: HTTP/<principal>@<realm>
servicePrincipalName: HTTP/<principal>
distinguishedName: CN=HTTP/<principal>,OU=Hadoop,DC=fresh,DC=com
userAccountControl: 514
```

где `<principal>` — имя принципала для сервиса.

Удачное добавление принципала должно завершиться сообщением, похожим на представленное ниже:

```
adding new entry "CN=HTTP/<principal>,OU=kerberos,ou=adh,DC=ad,DC=ranger-test"
```

Для более подробного вывода используйте опцию `-v` при запуске ldapadd, включающую режим verbose.

Правильность добавления принципалов можно проверить с помощью ldapsearch:

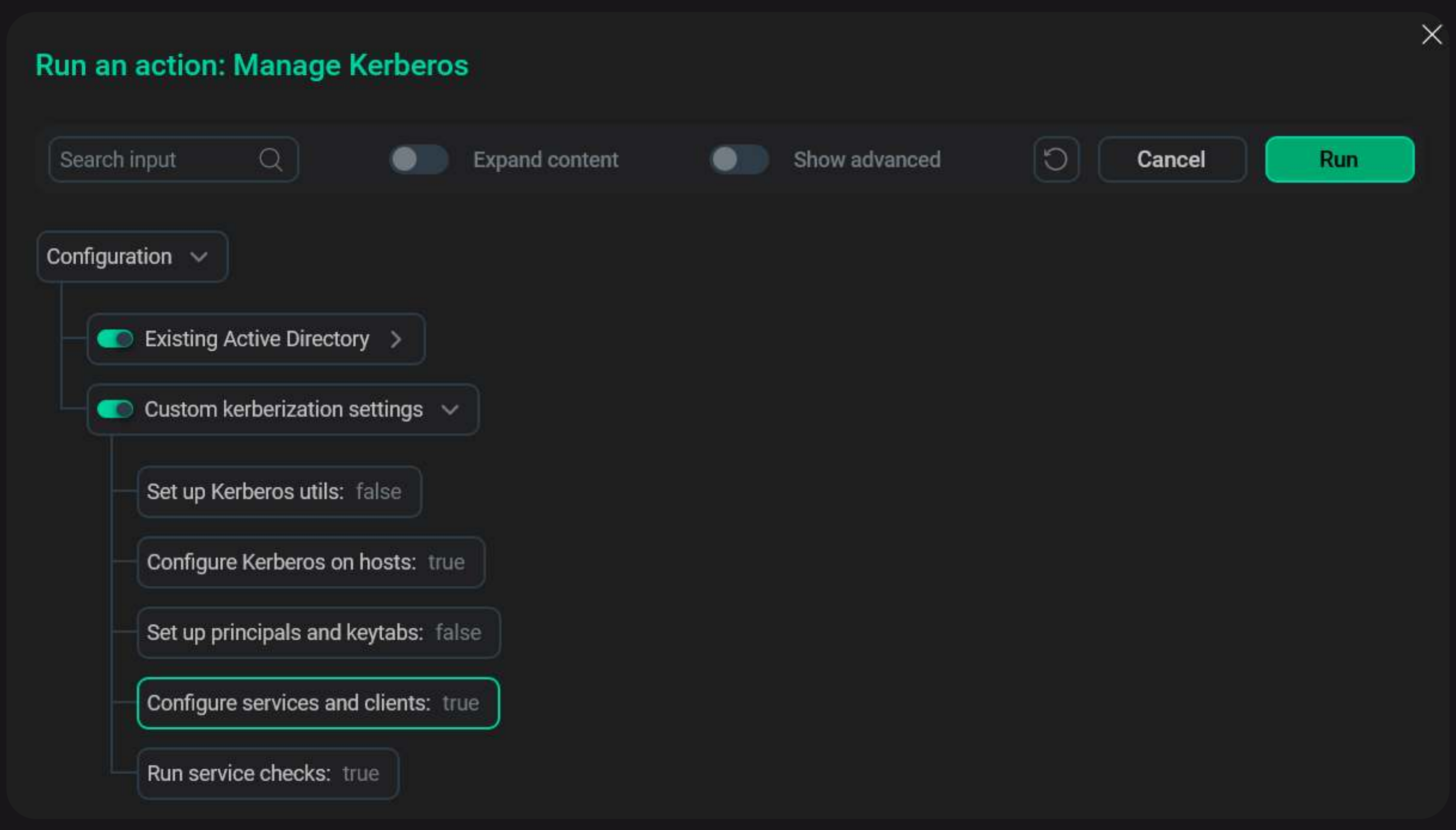
```
$ ldapsearch -x -H <ldap_link> -D <username>@<realm> -w <password> -b <search_base> <search_filter>
```

где:

- `<search_base>` — база для поиска.
- `<search_filter>` — поисковой фильтр.

Активация Kerberos

Запустите действие ***Manage Kerberos*** с выбранной опцией ***Existing Active Directory*** и установленными флагами ***Configure Kerberos on hosts***, ***Configure services and clients*** и ***Run Service checks***.



Run an action: Manage Kerberos

Configuration
- Existing Active Directory
- Custom kerberization settings
  - Set up Kerberos utils: false
  - Configure Kerberos on hosts: true
  - Set up principals and keytabs: false
  - Configure services and clients: true
  - Run service checks: true

Завершение кербе√изации

Частые проблемы при настройке AD Kerberos

Константин Алпашкин, Сергей Тихомиров

Содержание

Назад к содержанию

- Ошибка создания HTTP-принципала на хосте в домене AD
  - Как исправить
  - Создание HTTP-принципала
- Принципал уже существует, но в другом контейнере
  - Как исправить
  - Пример
- У администратора нет прав на base_dn в кастомном ldap.conf
  - Как исправить

В данной статье описаны частые проблемы, возникающие при настройке AD Kerberos, а также способы их решения. Для удобства работы и визуализации структур LDAP-сервера можно использовать Apache Directory Studio либо иной аналогичный инструмент.

> **ПРИМЕЧАНИЕ**
>
> В этой статье термин **контейнер** означает запись LDAP, которая является base DN (search base) для сервисных принципалов текущего пользователя. Такая запись может содержать Organization Unit (OU) и другие элементы DIT, а ее значение используется в поле ***Container DN*** при выполнении операции Manage Kerberos.

Некоторые известные кейсы с возможными решениями приведены ниже.

Ошибка создания HTTP-принципала на хосте в домене AD

Такая ошибка возникает в процессе включения Kerberos, если на хосте AD установлено обновление KB5008382 и хосты кластера находятся в одном домене с хостом AD. Ниже представлен пример сообщения о данной ошибке.

```
######################## STDOUT ########################
adding new entry "CN=HTTP/dl-dev-h3-kdc.adp.local,OU=svc,OU=EIAP,DC=adp,DC=local"

######################## STDERR ########################
ldap_add: Constraint violation (19)
        additional info: 000021C7: AtrErr: DSID-03200E89, #1:
        0: 000021C7: DSID-03200E89, problem 1005 (CONSTRAINT_ATT_TYPE), data 0, Att 90303 (servicePrincipalName)
```

Как исправить

Существует два решения:

- Удалить хосты из домена AD, провести керберизацию, а потом вернуть хосты.
- Создать HTTP-принципал с правами администратора AD перед керберизацией.

Создание HTTP-принципала

1. Авторизуйтесь на сервере с AD как администратор AD с правами на управление хостами AD. Это можно сделать двумя способами:
   - Подключитесь к Windows-серверу с AD и перейдите в раздел ***AD Users and Computers***. Создайте принципал `HTTP/<fqdn>@<realm>` для каждого хоста.
   - Подготовьте среду (см. Полезная информация), затем подключитесь через консоль с помощью ldap-клиента и запустите следующую команду:

   ```
   ldapadd -H "ldaps://kru-win-new.fresh.com" -x -D administrator@FRESH.COM -w "mypassword" -f adduser.ldif
   ```

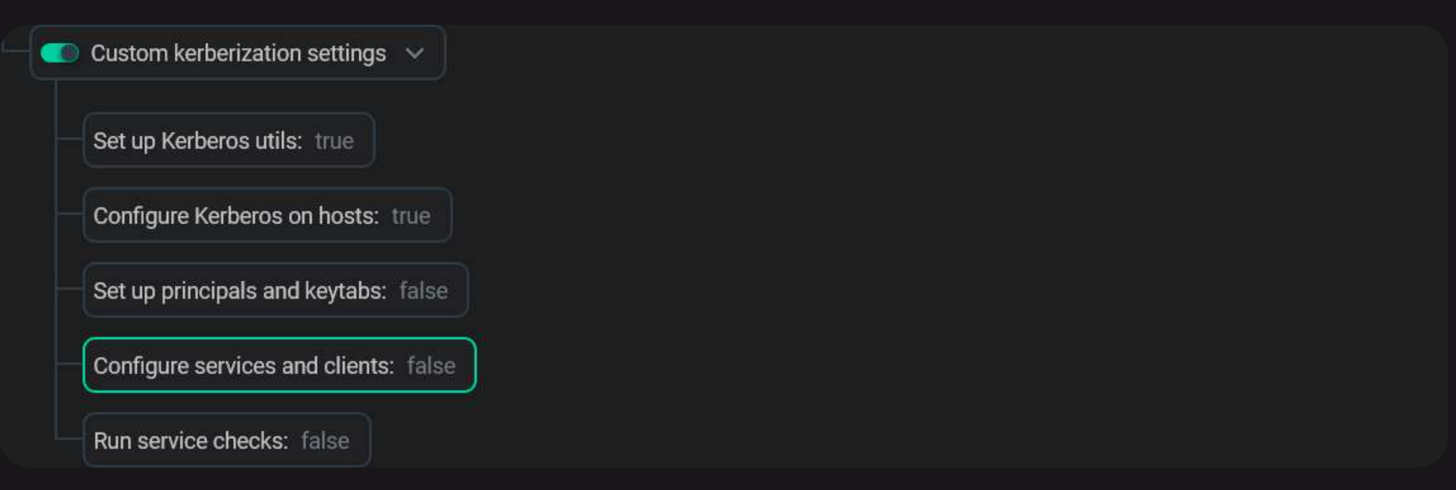
   Пример конфигурационного файла *adduser.ldif* представлен ниже.

   ```
   DN: CN=HTTP/kru-test-adps.ru-central1.internal,OU=Hadoop,DC=fresh,DC=com
   changetype: add
   objectClass: top
   objectClass: person
   objectClass: organizationalPerson
   objectClass: user
   accountExpires: 0
   userPrincipalName: HTTP/kru-test-adps.ru-central1.internal@FRESH.COM
   servicePrincipalName: HTTP/kru-test-adps.ru-central1.internal
   distinguishedName: CN=HTTP/kru-test-adps.ru-central1.internal,OU=Hadoop,DC=fresh,DC=com
   userAccountControl: 514
   ```

2. Запустите действие Manage Kerberos. Если у вас нет локальных прав администратора, то отключите флаг ***Set up principals and keytabs***, чтобы пропустить создание принципалов и keytab-файлов. В таком случае вам нужно создать все принципалы и keytab-файлы самостоятельно.

Полезная информация

- Для подготовки среды Kerberos на хосте вы можете установить ldap-клиент со всеми настройками кластера используя ***Manage Kerberos*** со включенной опцией ***Custom kerberization settings*** и установленными флагами ***Setup Kerberos Clients*** и ***Cluster configuration***.



Set up Kerberos utils: true
Configure Kerberos on hosts: true
Set up principals and keytabs: false
Configure services and clients: false
Run service checks: false

Customize action

- Важно разместить HTTP-принципал в той же директории, которая указана в поле ***Container DN*** в окне, открывающемся при запуске ***Manage Kerberos***, так как локальный администратор должен иметь права на управление принципалом.

Так, Kerberos будет включен с использованием существующего HTTP-принципала. Пароль принципала будет сгенерирован, а keytab-файл будет записан на все хосты.

Принципал уже существует, но в другом контейнере

ADCM не может выполнять поиск/удаление/создание сервисных принципалов в контейнерах, если имя контейнера отличается от значения Container DN, указанного при включении/отключении Kerberos. Поскольку текущий администратор может не иметь прав на весь домен AD, такие принципалы, существующие в разных контейнерах с одинаковыми именами, приводят к конфликту:

```
######################## STDERR ########################
ldap_add: Constraint violation (19)
    additional info: 000021C8: AtrErr: DSID-03200BE8, #1:
    0: 000021C8: DSID-03200BE8, problem 1005 (CONSTRAINT_ATT_TYPE), data 0, Att 90290 (userPrincipalName).
```

Как исправить

Удалите существующие принципалы из старого контейнера вручную. Пример, как это сделать, показан ниже.

Пример

1. Проверьте существующие принципалы во всех контейнерах домена, используя следующий фильтр:

```
ldapsearch -H ldap://{ldap-server-url} -o ldif-wrap=no -x -W -D 'cn=userA,ou=kerberos,ou=adh,dc=foo,dc=bar' -b 'dc=foo,dc=bar'
name={principal-name} | grep dn
```

   Где:

   - `-b 'dc=foo,dc=bar'` состоит из частей доменного имени (например, 'dc=com,dc=hostname'), соответствующего Kerberos REALM.
   - `-o ldif-wrap=no` отключает перенос строк (results wrapping) для результатов ldapsearch. Если не указывать параметр, результаты ldapsearch могут быть обрезаны и `grep` не найдет полное значение DN, необходимое для удаления принципала.

   Вызов ldapsearch возвращает Container DN(s), например:

```
# Пример 1:
dn: CN=hdfs/test-principal-name.ru-central1.internal,OU=e_ivanov_krb1_ou,OU=kerberos,OU=adh,DC=ad,DC=ranger-test

# Пример 2:
dn: CN=del_me,CN=Users,DC=adh-sec,DC=com
```

2. Удалите существующие конфликтные принципалы. ADCM пересоздаст принципалы с актуальным значением Container DN. Например:

```
ldapdelete -H ldap://<ldap-server-url> -x -W -D 'cn=userA,ou=kerberos,ou=adh,dc=ad,dc=ranger-test' CN=hive/test-principal-name2.
ru-central1.internal,OU=userB,OU=kerberos,OU=adh,DC=ad,DC=ranger-test
```

3. После удаления всех сервисных принципалов из контейнера выполните действие ***(Re)Enable Kerberos***. Действие должно выполниться без конфликтов.

У администратора нет прав на base_dn в кастомном ldap.conf

В этом случае при расширении компонента (expand) на новый хост или при добавлении новых сервисов возможна следующая ошибка:

```
<timestamp> DEBUG adcm_check ansible adcm_check: title: Hdfs-Datanode: Add principal, title: <principal-name>,
message:######################## STDOUT ########################

######################## STDERR ########################
ldap_sasl_bind(SIMPLE): Can't contact LDAP server (-1)
```

Как исправить

Проверьте, соответствует ли `base_dn` в файле *ldap.conf* актуальному значению Container DN для текущего пользователя. Если это не так, измените `base_dn` в файле *ldap.conf* или выполните действие ***Re-enable Kerberos*** с корректным значением Container DN.

Обзор FreeIPA

Константин Алпашкин

Назад к содержанию

FreeIPA — это бесплатная система управления идентификацией пользователей для Linux/UNIX с открытым исходным кодом. FreeIPA включает в себя такие компоненты, как Linux (Fedora), 389 Directory Server, MIT Kerberos, NTP, DNS, Dogtag (система сертификатов). Основной задачей FreeIPA является централизованное управление учетными данными и аутентификация.

Для обеспечения безопасности Arenadata Hyperwave использует Kerberos, LDAP, SSL-сертификаты/ключи и другие смежные технологии. Для такого набора инструментов FreeIPA отлично подходит в качестве системы управления идентификацией.

ПРИМЕЧАНИЕ

FreeIPA поддерживается начиная с версии ADH 2.1.6.b1.

Для керберизации кластера с помощью FreeIPA используется ADCM. В процессе керберизации ADCM устанавливает IPA-клиент на каждый хост кластера и устанавливает соединение с сервером FreeIPA, используя заданные параметры Kerberos для FreeIPA.

В данном руководстве для демонстрации примеров используется тестовый FreeIPA-сервер, развернутый на единственном хосте (CentOS). Этот сервер представляет собой базовую конфигурацию `yum install ipa-server` с параметрами установки по умолчанию.

Настройка использования FreeIPA для Kerberos-аутентификации с помощью ADCM

Константин Алпашкин, Сергей Тихомиров

Содержание

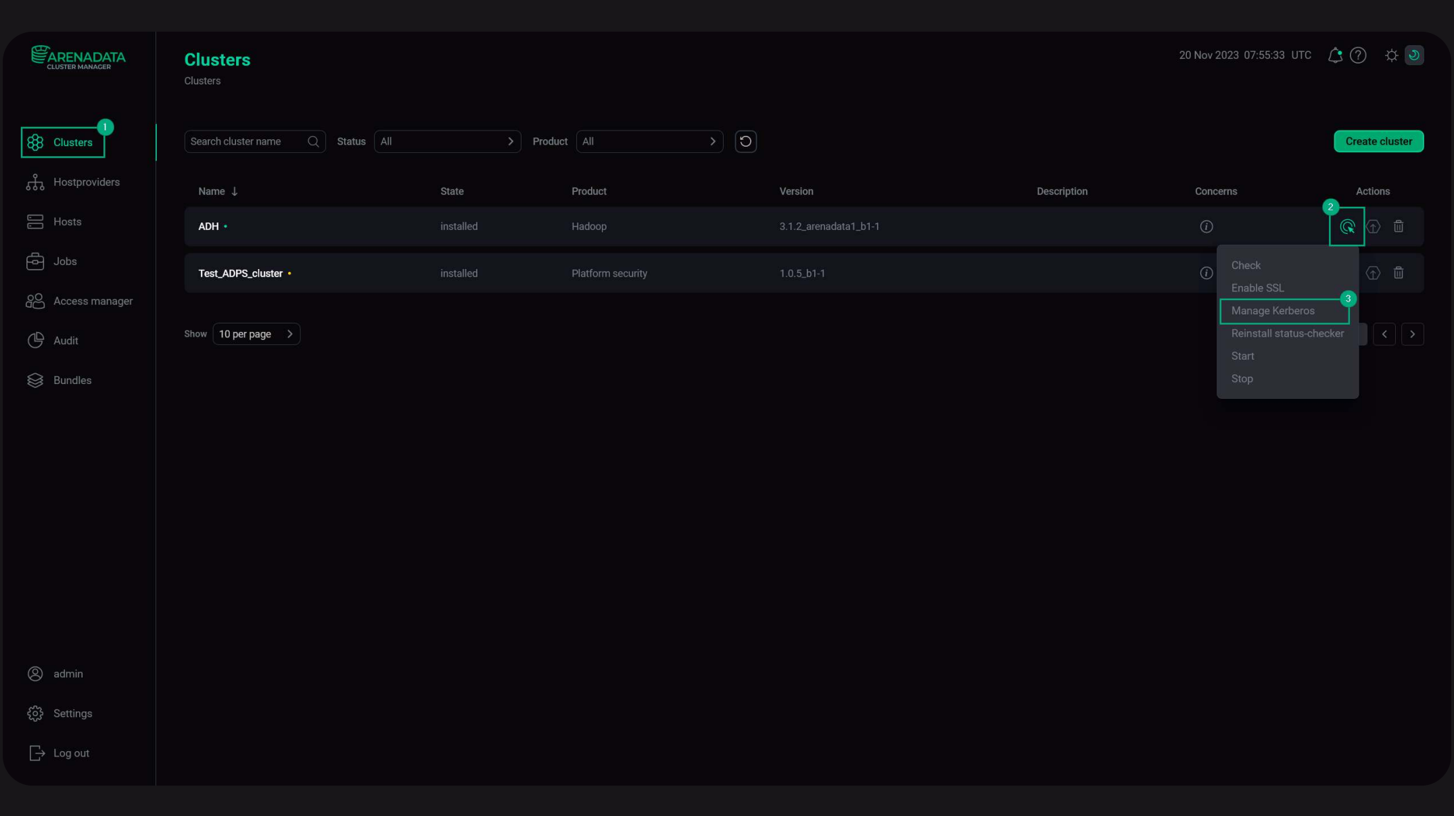
Назад к содержанию

- Обзор
- Kerberos-параметры FreeIPA
- Проверка интеграции с FreeIPA
  - Доступ к HDFS до керберизации
  - Доступ к HDFS после керберизации с FreeIPA

Обзор

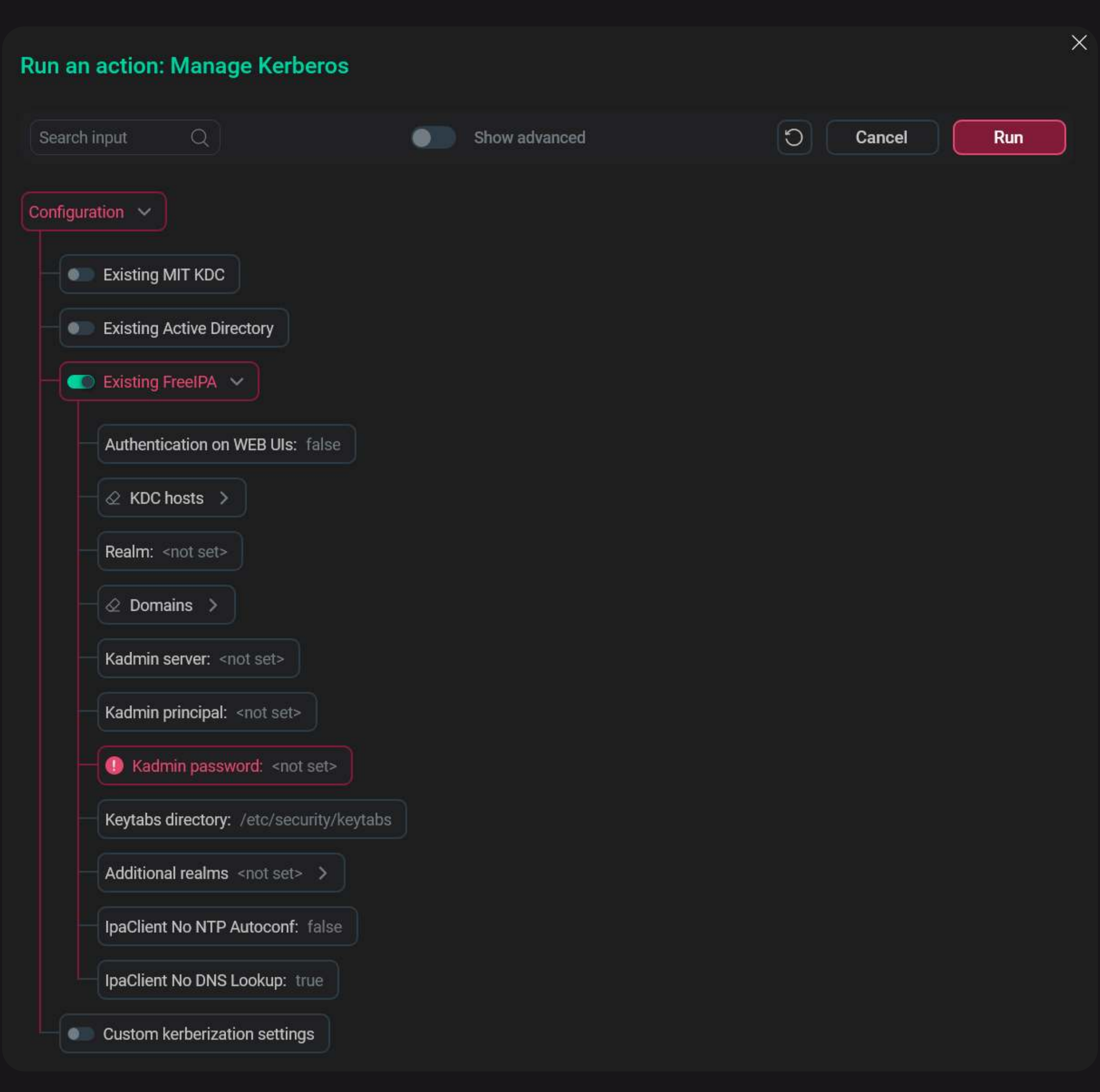
Для керберизации кластера с помощью FreeIPA выполните описанные ниже шаги:

1. В веб-интерфейсе ADCM перейдите на страницу ***Clusters***. Выберите установленный и подготовленный кластер ADH, и запустите действие ***Manage Kerberos***.


ARENADATA CLUSTER MANAGER
Clusters
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
20 Nov 2023 07:55:33 UTC
Search cluster name
Status
All
Product
All
Create cluster
Name
State
Product
Version
Description
Concerns
Actions
ADH
installed
Hadoop
3.1.2_arenadata1_b1-1
Test_ADPS_cluster
installed
Platform security
1.0.5_b1-1
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
Show
10 per page
1
2
3

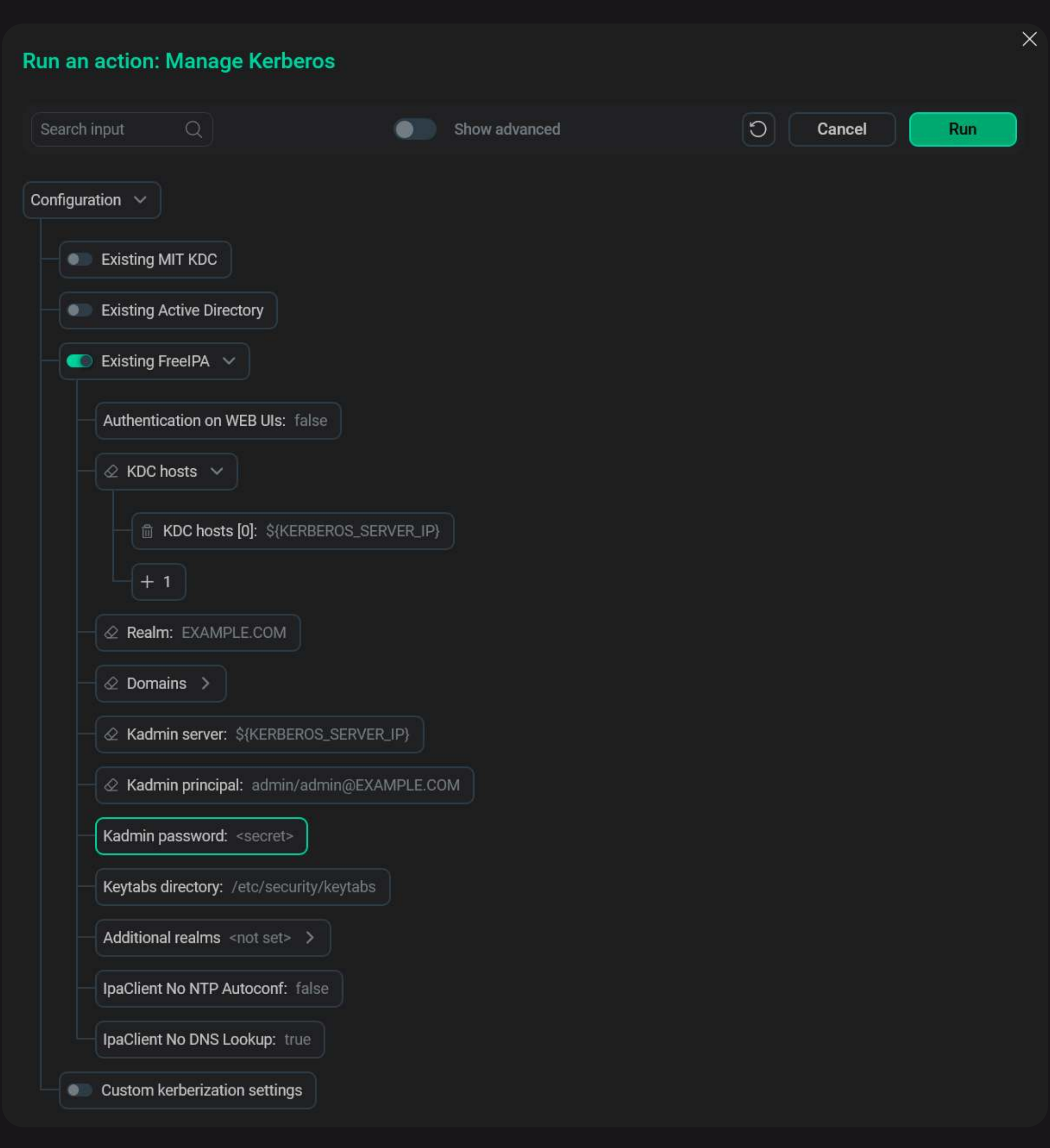
Manage Kerberos

2. В появившемся окне выберите опцию ***Existing FreeIPA***.


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Authentication on WEB UIs: false
KDC hosts
Realm: <not set>
Domains
Kadmin server: <not set>
Kadmin principal: <not set>
Kadmin password: <not set>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
IpaClient No NTP Autoconf: false
IpaClient No DNS Lookup: true
Custom kerberization settings

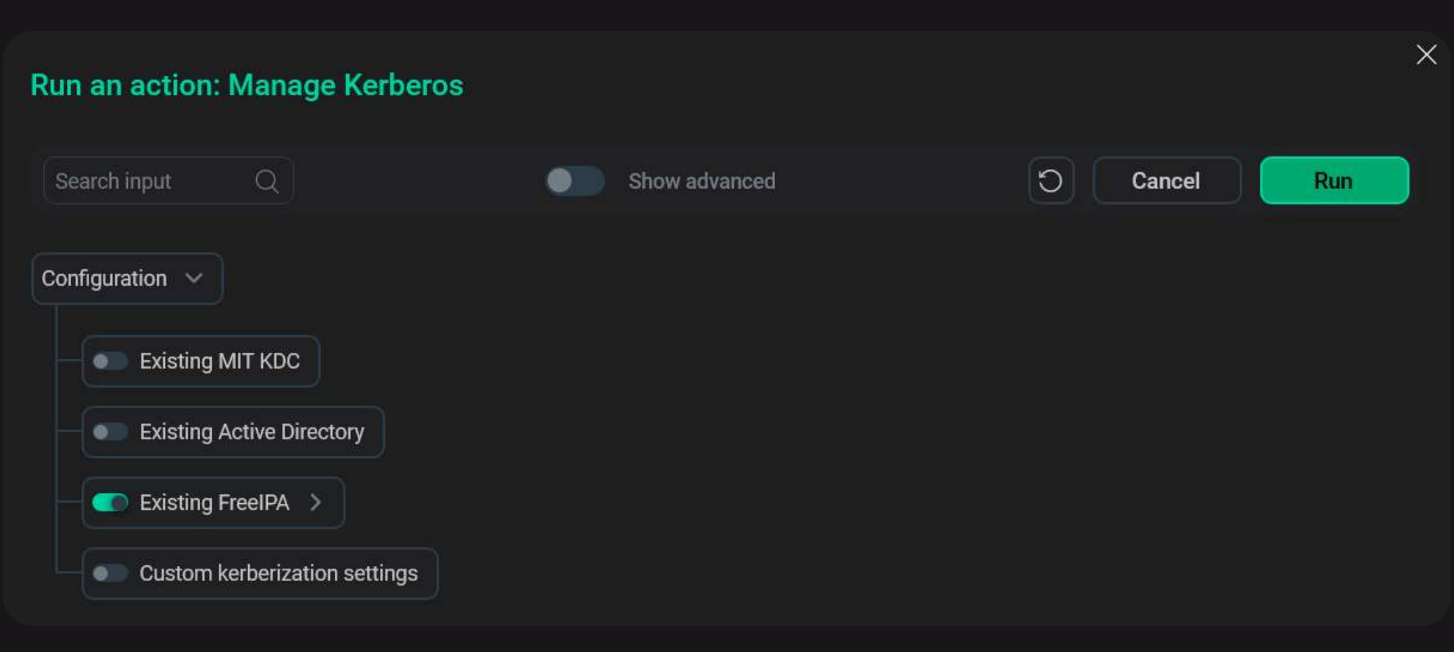
Выбор соответствующей опции

3. Заполните Kerberos-параметры FreeIPA и приступайте к настройке Kerberos в кластере.


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Authentication on WEB UIs: false
KDC hosts
KDC hosts [0]: ${KERBEROS_SERVER_IP}
+ 1
Realm: EXAMPLE.COM
Domains
Kadmin server: ${KERBEROS_SERVER_IP}
Kadmin principal: admin/admin@EXAMPLE.COM
Kadmin password: <secret>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
IpaClient No NTP Autoconf: false
IpaClient No DNS Lookup: true
Custom kerberization settings

Kerberos-параметры FreeIPA

4. Нажмите ***Run*** и дождитесь окончания действия. Затем приступайте к настройке Kerberos в кластере.


Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Custom kerberization settings

Запуск действия

Kerberos-параметры FreeIPA

Следующая таблица содержит описания параметров, необходимых для керберизации ADH-кластера с помощью FreeIPA. Актуальные значения для этих параметров можно получить, выполнив команду `ipa user-find <ipa_admin>` на FreeIPA-сервере, где `<ipa_admin>` — имя пользователя IPA Admin.

Параметр	Описание
Authentication on WEB UIs	Включает аутентификацию Kerberos для веб-интерфейсов
KDC hosts	Один или более хостов, на которых запущен(ы) FreeIPA-сервер(а). Допустим только FQDN
Realm	Kerberos realm для подключения к FreeIPA-серверу
Domains	Один или более доменов, привязанных к FreeIPA
Kadmin server	Адрес хоста, на котором запущен `kadmin`. Допустим только FQDN
Kadmin principal	Имя принципала для подключения через `kadmin`, например `admin@RU-CENTRAL1.INTERNAL`
Kadmin password	Пароль пользователя IPA Admin
Keytabs directory	Директория keytab-файла, содержащего записи одного или нескольких принципалов и их ключи
Additional realms	Дополнительные сети Kerberos realm
IpaClient No NTP Autoconf	Отключает настройку NTP при установке IPA-клиентов
IpaClient No DNS Lookup	Отключает поиск сервера FreeIPA по DNS при установке IPA-клиентов
Number of retries for kinit invocation attempts	Количество повторных попыток вызова kinit

Проверка интеграции с FreeIPA

После успешной керберизации выполнение hadoop-команд на узлах кластера возможно только после получения соответствующего тикета Kerberos. Ниже приведены примеры доступа к незащищенному кластеру, а также к кластеру, керберизованному с помощью FreeIPA.

Доступ к HDFS до керберизации

HDFS CLI

Листинг корневой директории HDFS:

```
$ hdfs dfs -ls /
```

Пример вывода:

```
Found 4 items
drwxrwxrwt   - yarn hadoop          0 2023-01-31 16:46 /logs
drwxr-xr-x   - hdfs hadoop          0 2023-01-31 16:43 /system
drwxrwxrwx   - hdfs hadoop          0 2023-01-31 21:46 /tmp
drwxr-xr-x   - hdfs hadoop          0 2023-01-31 17:00 /user
```

HDFS HttpFS

Запрос curl к HttpFS на листинг корневой директории HDFS:

```
$ curl "http://ka-adh-2.ru-central1.internal:14000/webhdfs/v1/?op=LISTSTATUS&user=admin"
```

Пример HTTP-ответа:

```
{"FileStatuses":{"FileStatus":
[{"pathSuffix":"logs","type":"DIRECTORY","length":0,"owner":"yarn","group":"hadoop","permission":"1777","accessTime":0,"modificationTime":1675183594965,"blockSize":0,"replication":0},...
```

Доступ к HDFS после керберизации с FreeIPA

HDFS CLI

Листинг корневой директории HDFS:

```
$ hdfs dfs -ls /
```

Пример вывода (доступ запрещен):

```
2023-02-01 21:15:51,819 WARN ipc.Client: Exception encountered while connecting to the server : org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
2023-02-01 21:15:51,831 WARN ipc.Client: Exception encountered while connecting to the server : org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS]
2023-02-01 21:15:51,832 INFO retry.RetryInvocationHandler: java.io.IOException: DestHost:destPort ka-adh-3.ru-central1.internal:8020 , LocalHost:localPort ka-adh-1.ru-central1.internal/10.92.17.141:0. Failed on local exception: java.io.IOException: org.apache.hadoop.security.AccessControlException: Client cannot authenticate via:[TOKEN, KERBEROS], while invoking ClientNamenodeProtocolTranslatorPB.getFileInfo over ka-adh-3.ru-central1.internal/10.92.16.90:8020 after 1 failover attempts. Trying to failover after sleeping for 518ms
...
```

Чтобы пройти аутентификацию, UNIX-пользователь, от имени которого выполняется команда `hdfs dfs …` (`admin` в нашем случае), должен получить тикет Kerberos:

```
$ kinit admin
```

После получения тикета команда `hdfs dfs -ls /` выводит содержимое корневой директории HDFS, что подтверждает успешную аутентификацию:

```
Found 4 items
drwxrwxrwt   - yarn hadoop          0 2023-01-31 16:46 /logs
drwxr-xr-x   - hdfs hadoop          0 2023-01-31 16:43 /system
drwxrwxrwx   - hdfs hadoop          0 2023-01-31 21:46 /tmp
drwxr-xr-x   - hdfs hadoop          0 2023-01-31 17:00 /user
```

HDFS HttpFS

Запрос curl к HttpFS:

```
$ curl "http://ka-adh-2.ru-central1.internal:14000/webhdfs/v1/?op=LISTSTATUS&user=admin"
```

HTTP-ответ (доступ запрещен):

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8"/>
<title>Error 401 Authentication required</title>
</head>
<body><h2>HTTP ERROR 401</h2>
<p>Problem accessing /webhdfs/v1/. Reason:
<pre>    Authentication required</pre></p>
</body>
</html>
```

Чтобы пройти аутентификацию, UNIX-пользователь, от имени которого запускается curl (`admin` в нашем случае), должен получить тикет Kerberos.

```
$ kinit admin
```

Запрос curl к HttpFS:

```
$ curl --negotiate -u : "http://ka-adh-2.ru-central1.internal:14000/webhdfs/v1/?op=LISTSTATUS&user=admin"
```

> **РЕКОМЕНДАЦИЯ**
>
> Флаг `--negotiate` активирует SPNEGO для доступа curl к ресурсу, защищенному Kerberos.

HTTP-ответ (доступ разрешен):

```
{"FileStatuses":{"FileStatus":
[{"pathSuffix":"logs","type":"DIRECTORY","length":0,"owner":"yarn","group":"hadoop","permission":"1777","accessTime":0,"modificationTime":1675183594965,"blockSize":0,"replication":0},
{"pathSuffix":"system","type":"DIRECTORY","length":0,"owner":"hdfs","group":"hadoop","permission":"755","accessTime":0,"modificationTime":1675183407542,"blockSize":0,"replication":0},...
```

Керберизация с предустановленным FreeIPA при отсутствии прав администратора

Сергей Тихомиров

Содержание

Назад к содержанию

Создание пользователя IPA без прав администратора

Подготовка хостов кластера

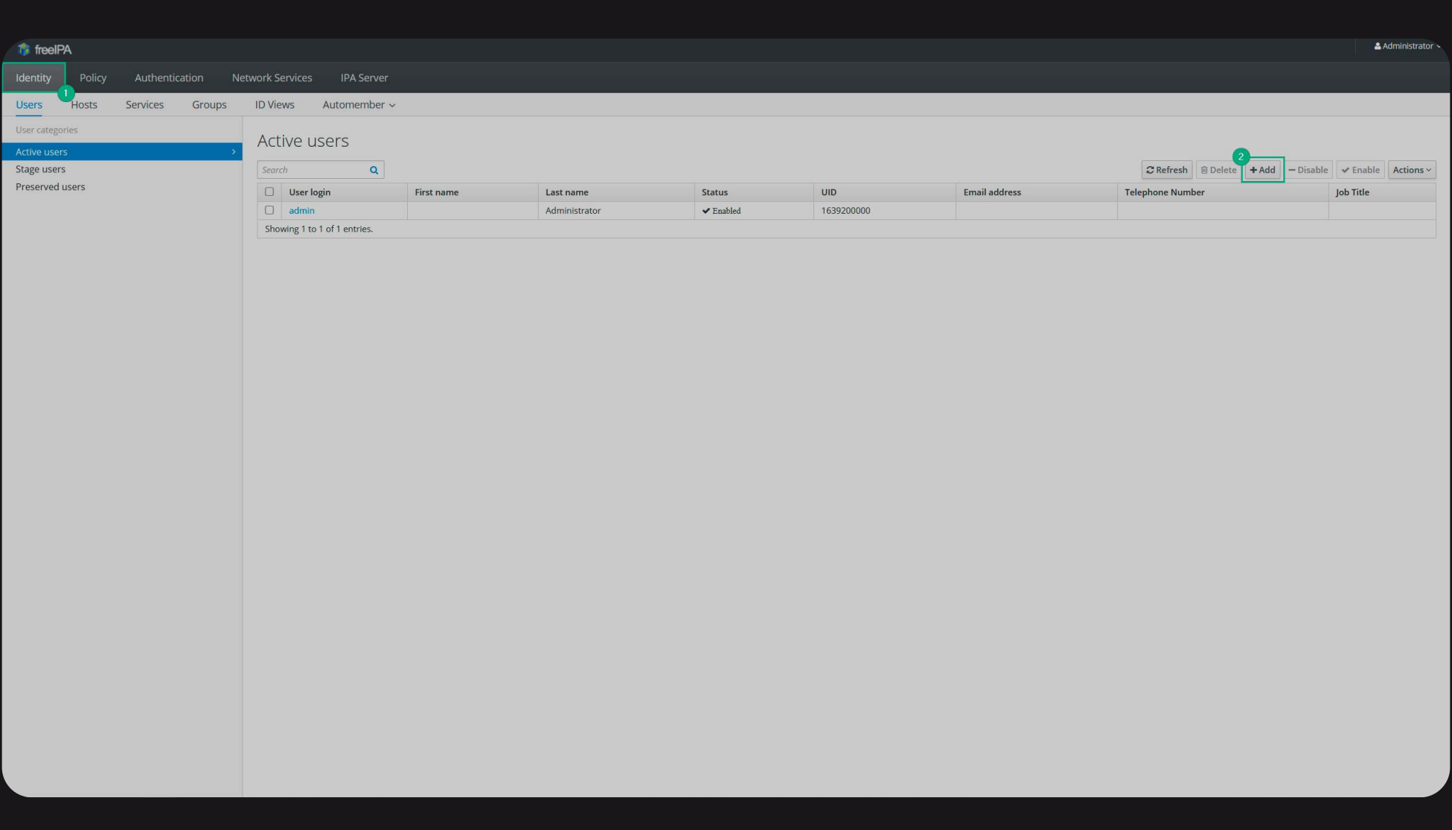
Активация Kerberos

Сопроводительная информация

Создание пользователя IPA без прав администратора

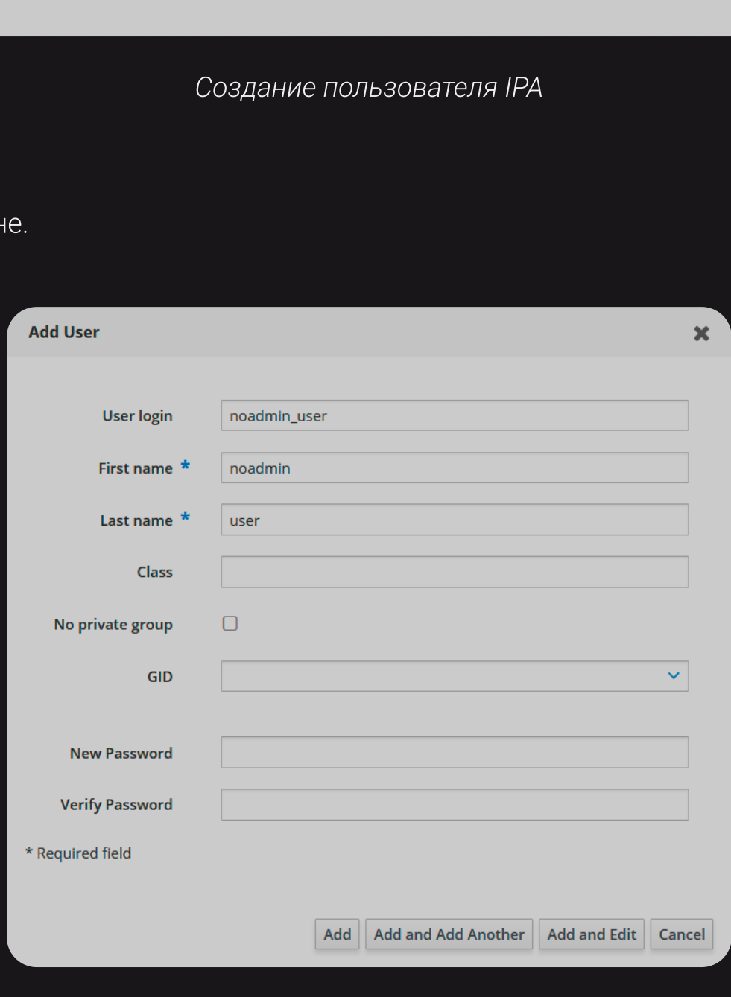
Если у вас нет пользователя IPA без прав администратора, то его можно создать в FreeIPA UI. Следуйте шагам ниже:

1. Залогиньтесь на ваш IPA-сервер как администратор, перейдите на вкладку ***Identity*** и нажмите ***Add***.



Создание пользователя IPA

2. Заполните данные пользователя в открывшемся окне.



User login: noadmin_user

First name *: noadmin

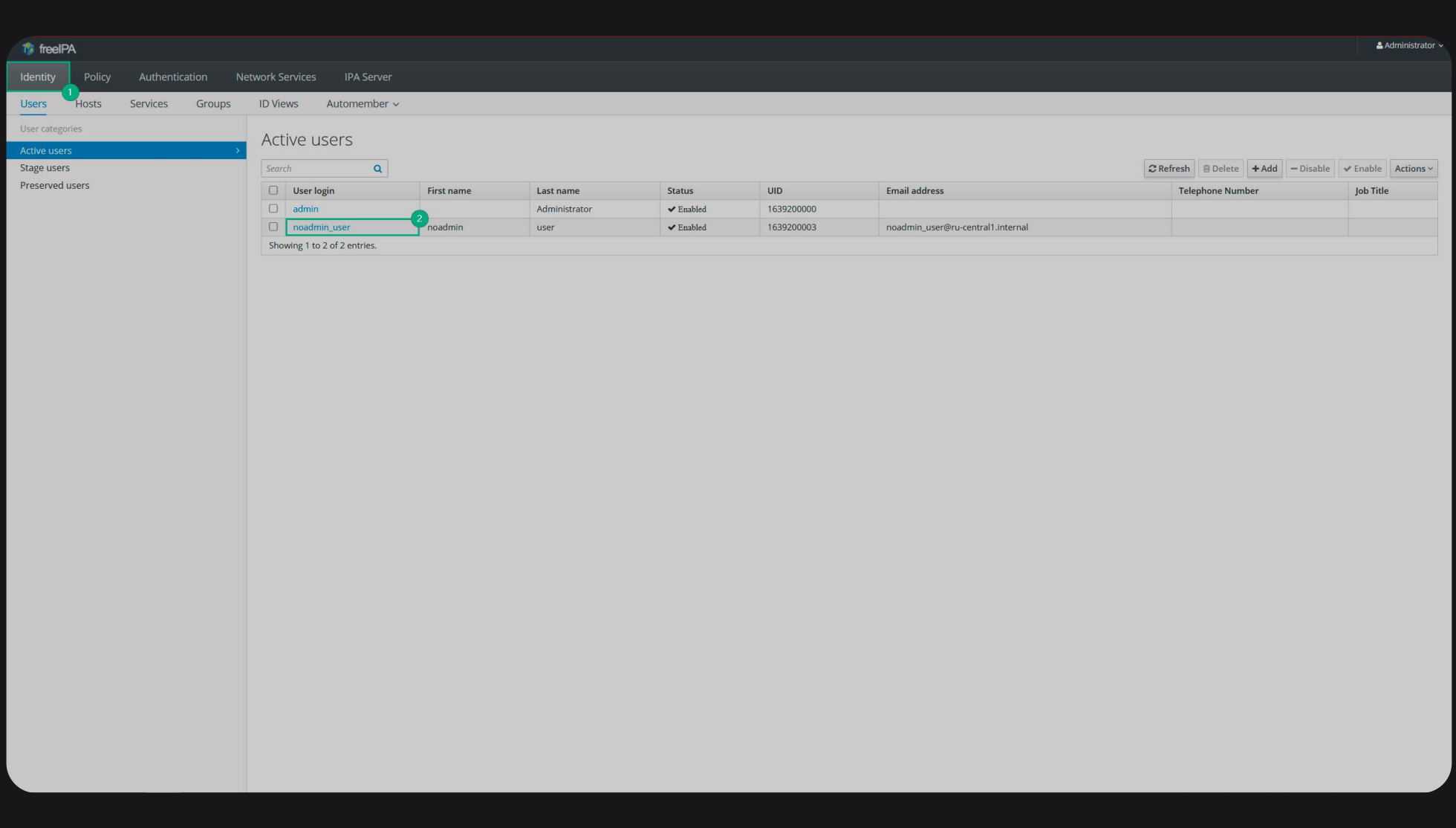
Last name *: user

* Required field

Данные пользователя IPA

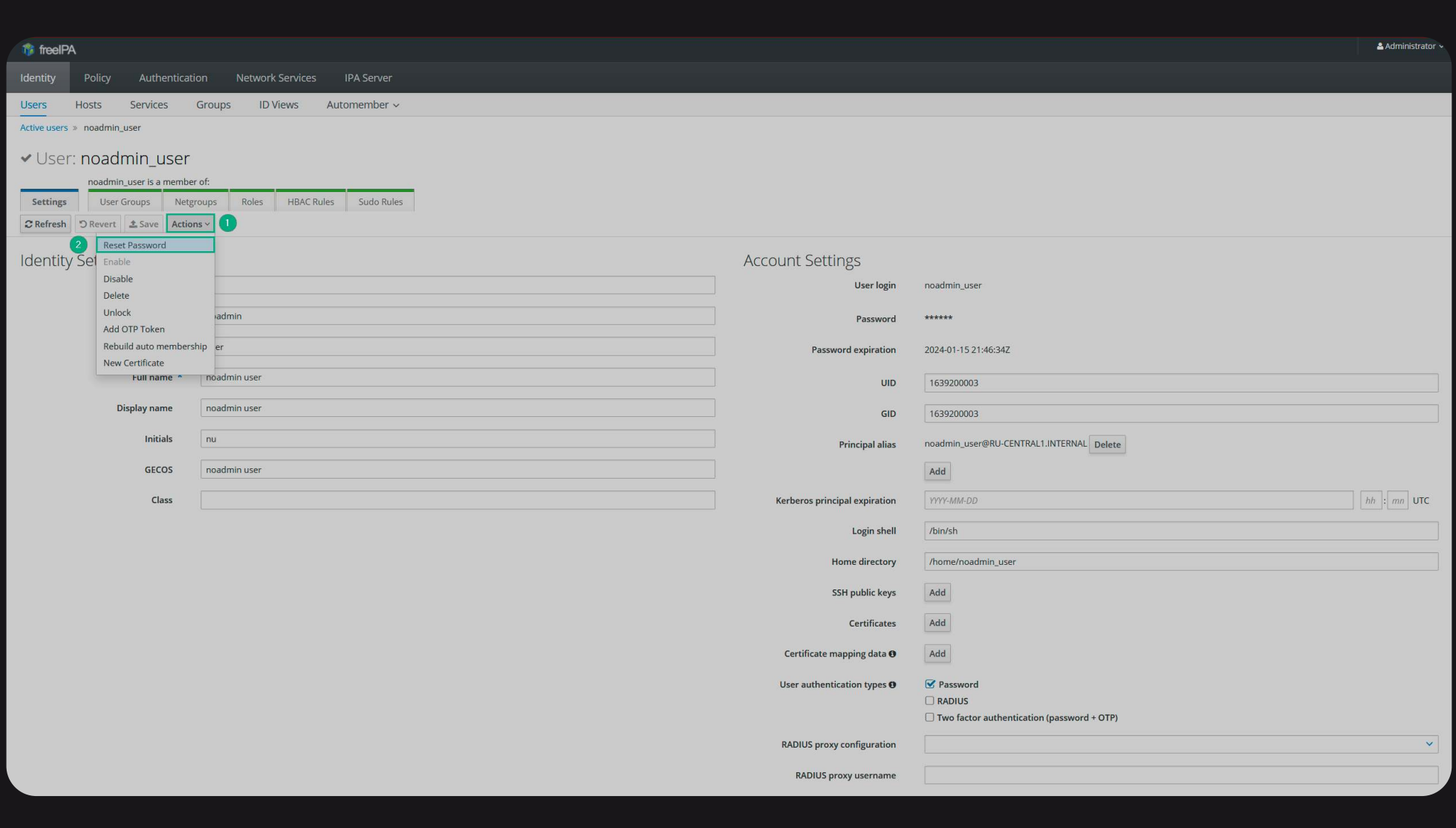
3. Установите для пользователя пароль:

   - На вкладке ***Identity*** нажмите на имя нового пользователя в колонке ***User login***.



Пользователи IPA

   - Нажмите ***Actions → Reset password***.



Установка пароля

   В открывшемся окне введите новый пароль и подтвердите его.

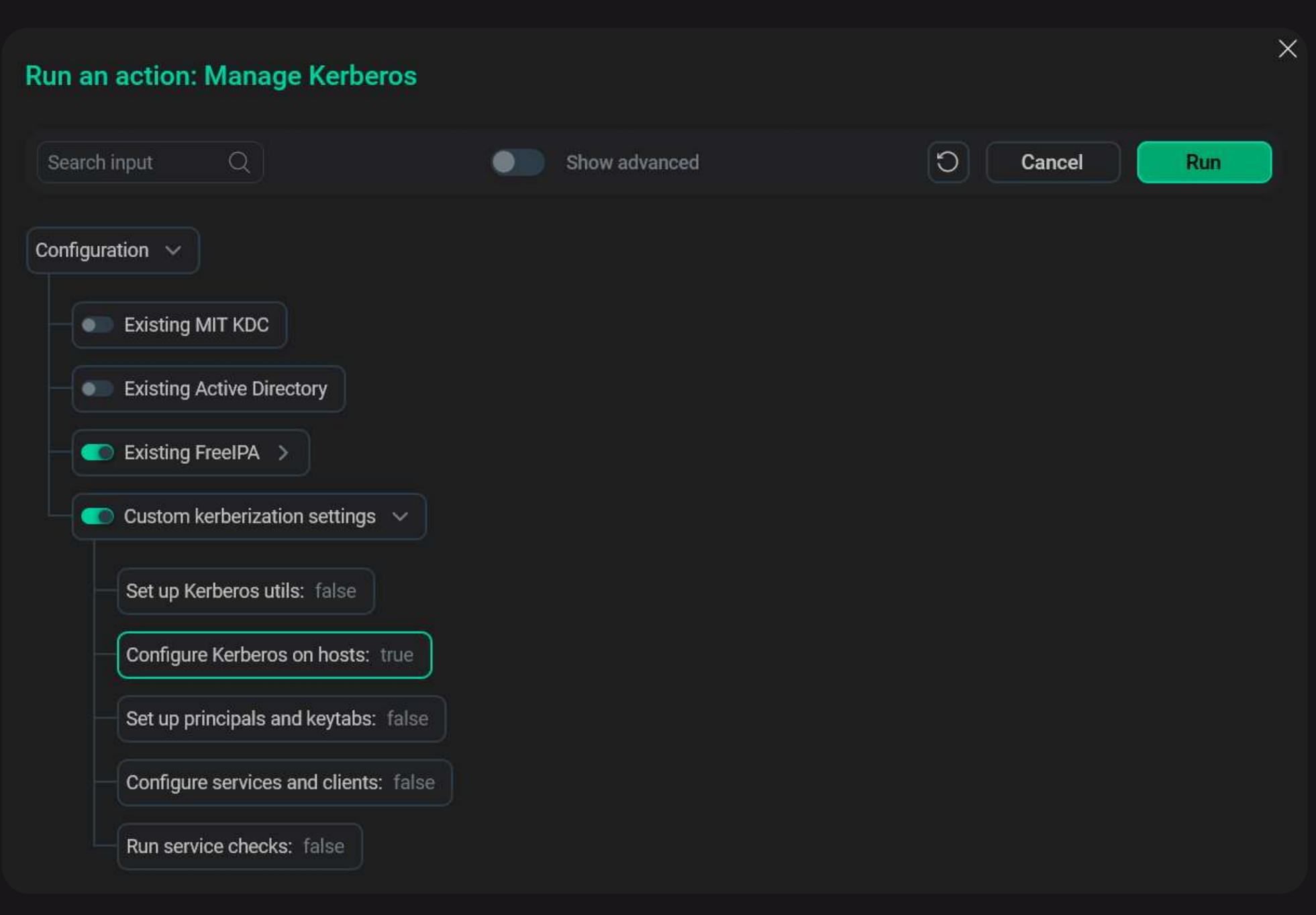
Подготовка хостов кластера

ВАЖНО

Хотя шаги 2—8 можно выполнить вручную, рекомендуется использовать скрипт, приведенный ниже.

На каждом хосте кластера выполните следующие действия:

1. Добавьте файл *krb5.conf* в директорию */etc/*. Это можно сделать самостоятельно, но также можно запустить действие Manage Kerberos с выбранной опцией ***Existing FreeIPA*** и установленным флагом ***Configure Kerberos on hosts*** в меню ***Custom kerberization settings***. При заполнении полей в форме ***Existing FreeIPA*** указывайте данные пользователя, не входящего в группу `admins`.



Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Custom kerberization settings

Set up Kerberos utils: false

Configure Kerberos on hosts: true

Set up principals and keytabs: false

Configure services and clients: false

Run service checks: false

Создание конфигурационного файла

2. Установите клиент FreeIPA с помощью команды ниже:

```
$ sudo ipa-client-install -U -p <ipa_user> -w <ipa_pass> --server=<ipa_server> --domain=<domain> --realm=<realm>
```

   где:

   - `<ipa_user>` — имя пользователя без прав администратора.
   - `<ipa_pass>` — пароль для `<ipa_user>`.
   - `<ipa_server>` — FQDN сервера с FreeIPA.
   - `<domain>` — домен FreeIPA, указанный при конфигурации FreeIPA.
   - `<realm>` — realm FreeIPA, указанный при конфигурации FreeIPA.

3. Вызовите команду `kinit` с данными пользователя без прав администратора:

```
$ kinit <username>
```

   где `<username>` — имя пользователя IPA без прав администратора.

4. Добавьте сервисы в IPA. В качестве списка сервисов, которые нужно добавить, используйте столбец ***Keytab-файл*** в таблице сопоставления owner-keytab.

```
$ ipa service-add <service_name>/<fqdn>
```

   где:

   - `<service_name>` — название компонента сервиса.
   - `<fqdn>` — FQDN хоста, на котором запускается команда.

5. Добавьте keytab-файлы для каждого сервиса и разместите их в директории */etc/security/keytabs*. В качестве списка keytab-файлов, которые нужно добавить, используйте столбец ***Keytab-файл*** в таблице сопоставления owner-keytab.

```
$ ipa-getkeytab -p <service_name>/<fqdn> -k <service_name>.service.keytab
```

   где:

   - `<service_name>` — название компонента сервиса.
   - `<fqdn>` — FQDN хоста, на котором запускается команда.

6. Добавьте пользователя для каждого сервиса, если его еще не существует. В качестве списка пользователей, которых нужно добавить, воспользуйтесь столбцом ***Пользователь owner*** в таблице сопоставления owner-keytab.

```
$ adduser <service_name>
```

   где `<service_name>` — название сервиса.

7. Измените owner для каждого keytab-файла, руководствуясь таблицей сопоставления owner-keytab.

```
$ chown <user>:<group> <keytab_name>
```

   где:

   - `<user>` — имя пользователя, который станет owner.
   - `<group>` — имя группы, которая станет owner.
   - `<keytab_name>` — имя keytab-файла.

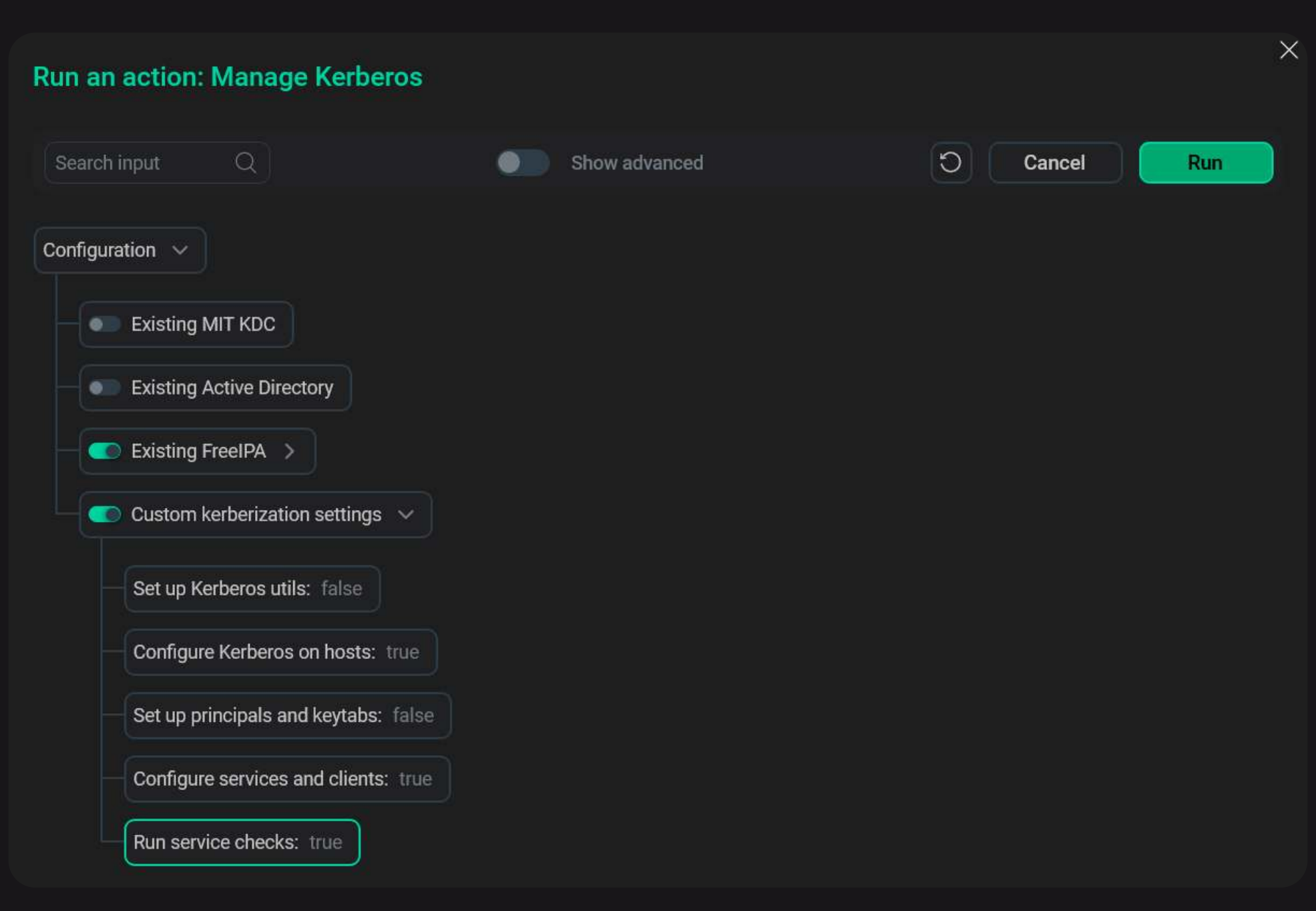
8. Установите права 640 keytab-файлам для сервсиов Spark, HDFS, HBase, Hive, HTTP и YARN:

```
$ chmod 640 <keytab_name>
```

   где `<keytab_name>` — имя keytab-файла.

Активация Kerberos

Запустите действие ***Manager Kerberos*** с выбранной опцией ***Existing FreeIPA*** и установленными флагами ***Configure Kerberos on hosts***, ***Configure services and clients***, ***Run service checks***.



Run an action: Manage Kerberos
Search input
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Custom kerberization settings

Set up Kerberos utils: false

Configure Kerberos on hosts: true

Set up principals and keytabs: false

Configure services and clients: true

Run service checks: true

Активация Kerberos

Сопроводительная информация

Сопоставление owner-keytab

Пользователь owner	Группа owner	Keytab-файл	Права
airflow	airflow	airflow.service.keytab	600
flink	flink	flink.service.keytab	600
hbase	hadoop	hbase.service.keytab	640
	hbase	hbase-thrift2.service.keytab	600
hdfs	hadoop	HTTP.service.keytab	640
		hdfs-datanode.service.keytab	600
		hdfs-journalnode.service.keytab	600
		hdfs-namenode.service.keytab	600
		hdfs-zkfc.service.keytab	600
		hdfs.service.keytab	640
hive	hive	hive.service.keytab	640
httpfs	httpfs	httpfs.service.keytab	600
livy	hadoop	livy.service.keytab	600
mapred	hadoop	mapreduce-historyserver.service.keytab	600
phoenix	phoenix	hbase-phoenix_queryserver.service.keytab	600
solr	solr	solr.service.keytab	600
spark	spark	spark.service.keytab	640
yarn	hadoop	yarn-nodemanager.service.keytab	600
		yarn-resourcemanager.service.keytab	600
		yarn-timelineserver.service.keytab	600
		yarn.service.keytab	640
zeppelin	zeppelin	zeppelin.service.keytab	600
zookeeper	zookeeper	zookeeper.service.keytab	600
impala	impala	impala.service.keytab	600

Скрипт для подготовки хостов

```
#!/bin/bash

# Script for adding services and keytabs to Cluster hosts
# Edit NUMHOSTS and HOSTS as it will be suitable for your case.
# EDIT your IPA parameters
# Select services according to the bundle

declare -a NUMHOSTS
declare -a HOSTS

SSH_OPTS='-o GlobalKnownHostsFile=/dev/null -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no'
NUMHOSTS=(adh3-1 adh3-2 adh3-3)
HOSTS=$(for i in "${NUMHOSTS[@]}"; do echo "stikhomirov-$i.ru-central1.internal"; done)

#IPA credentials
ipa_user='noadmin_username'
ipa_pass='12345678'
ipa_server='stikhomirov-freeipa.ru-central1.internal'
domain='ru-central1.internal'
realm='RU-CENTRAL1.INTERNAL'

#ADH services
services=(airflow flink hbase-phoenix_queryserver hbase-thrift2 hbase hdfs-datanode hdfs-journalnode hdfs-namenode hdfs-zkfc
hdfs hive httpfs livy mapreduce-historyserver solr spark sqoop yarn-nodemanager yarn-resourcemanager yarn-timelineserver yarn
zeppelin zookeeper impala)
user-groups=(airflow:airflow flink:flink hbase:hadoop hbase:hbase hdfs:hadoop hdfs:hadoop hdfs:hadoop hdfs:hadoop hdfs:hadoop
hdfs:hadoop hive:hive httpfs:httpfs livy:hadoop mapred:hadoop phoenix:phoenix solr:solr spark:spark sqoop:sqoop yarn:hadoop
yarn:hadoop yarn:hadoop yarn:hadoop zeppelin:zeppelin zookeeper:zookeeper impala:impala)
users=(airflow flink hbase hdfs hive httpfs impala livy mapred phoenix solr spark sqoop yarn zeppelin zookeeper)
rwr_services=(spark hdfs hbase hive yarn HTTP)

for HOST in $HOSTS; do
    echo "Installing FreeIPA client"
    ssh $SSH_OPTS $HOST sudo ipa-client-install -U -p $ipa_user -w $ipa_pass --server=$ipa_server --domain=$domain --
realm=$realm

    echo "Creating keytab directory"
          ssh $SSH_OPTS $HOST sudo mkdir /etc/security/keytabs

        echo "Launching kinit on $HOST"
          ssh $SSH_OPTS $HOST sudo kinit $ipa_user <<< $ipa_pass

    for user in ${users[@]}; do
        echo "Adding $user user"
            ssh $SSH_OPTS $HOST sudo adduser $user
    done

    for service in ${!services[@]}; do
        echo "Adding $service service to IPA"
            ssh $SSH_OPTS $HOST sudo ipa service-add ${services[i]}/$HOST
        echo "Get $service keytab"
            ssh $SSH_OPTS $HOST sudo ipa-getkeytab -p ${services[i]}/$HOST -k /etc/security/keytabs/${services[i]}.service.
keytab
        echo "Changing owner for $service keytab"
            ssh $SSH_OPTS $HOST sudo chowner ${user-groups[i]} /etc/security/keytabs/${services[i]}.service.keytab
    done

    for service in ${rwr_services[@]}; do
        echo "Setting 640 permissions for $service"
            ssh $SSH_OPTS $HOST sudo chmod 640 /etc/security/keytabs/$service.service.keytab
    done
done
```

Настройка использования Samba для Kerberos-аутентификации с помощью ADCM

Сергей Тихомиров

Содержание

Назад к содержанию

Обзор
Керберизация в ADCM
Kerberos-параметры Samba

Обзор

Samba выступает контроллером домена Active Directory, поэтому настройка Kerberos-аутентификации с Samba очень похожа на настройку аутентификации с Active Directory.

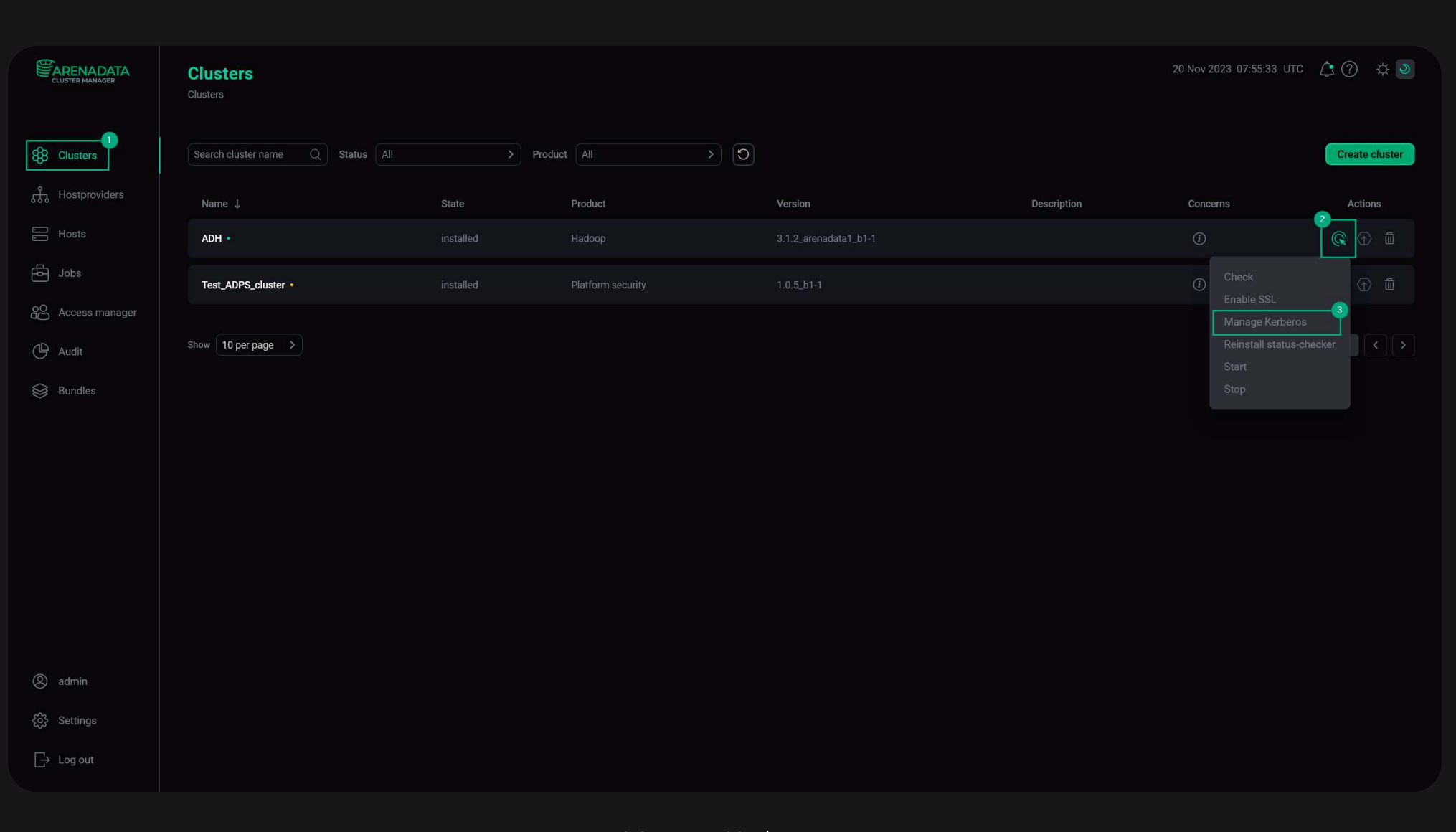
Особенности CentOS 7

На CentOS 7 бандл не генерирует keytab-файлы автоматически, так как стандартный пакет Samba для CentOS 7 не поддерживает создание принципалов на удаленном контроллере. Из-за такой специфики вам необходимо подготовить keytab-файлы вручную. Для этого вы можете изменить переменные в скрипте ниже и использовать его. На других ОС генерация происходит автоматически.

Керберизация в ADCM

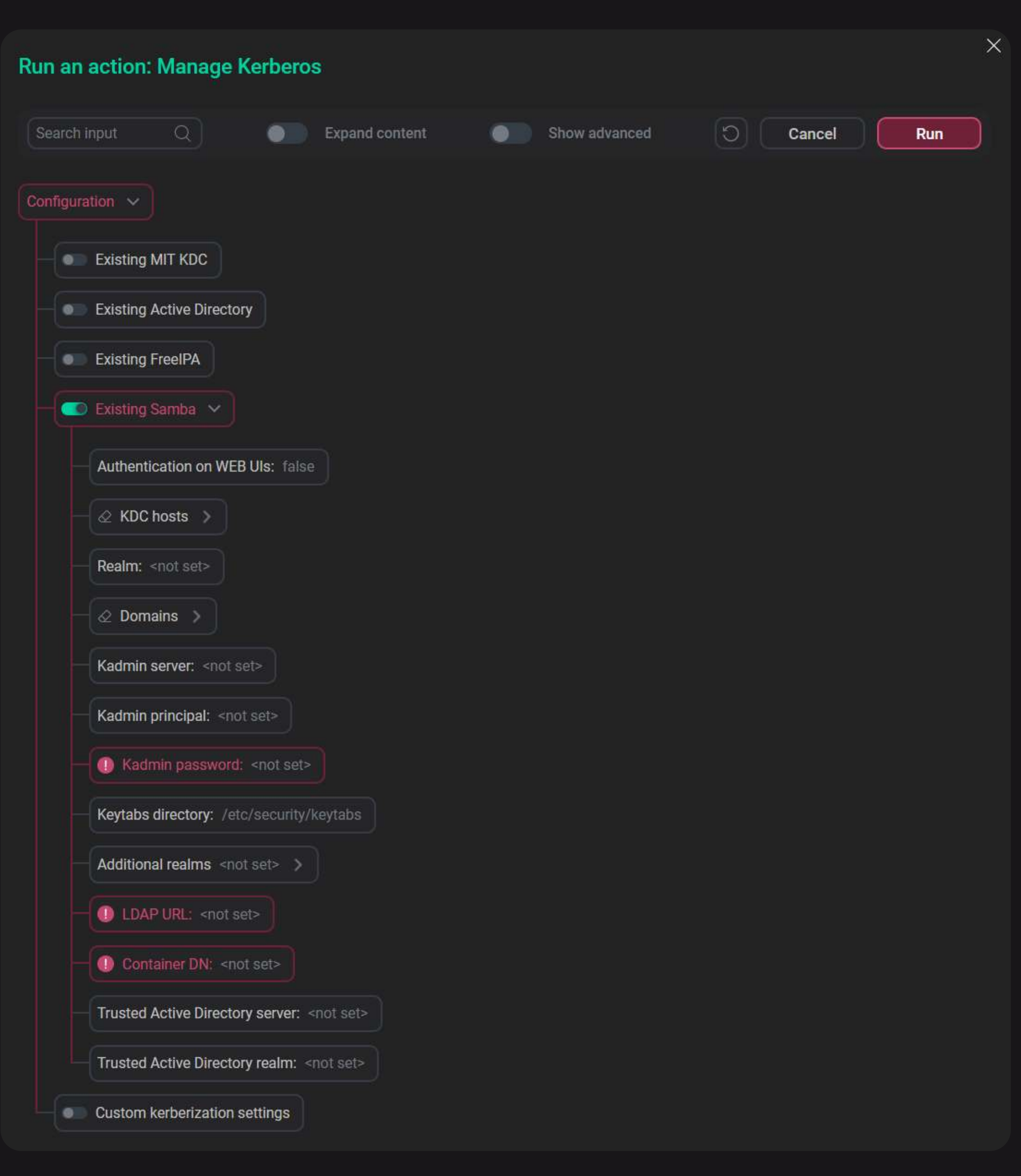
Для керберизации кластера с помощью Samba выполните описанные ниже шаги:

1. В веб-интерфейсе ADCM перейдите на страницу ***Clusters***. Выберите установленный и подготовленный кластер ADH, и запустите действие ***Manage Kerberos***.


ARENADATA
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Search cluster name
Status All
Product All
Create cluster
Name
State
Product
Version
Description
Concerns
Actions
ADH
installed
Hadoop
3.1.2_arenadata1_b1-1
Test_ADPS_cluster
installed
Platform security
1.0.5_b1-1
Check
Enable SSL
Manage Kerberos
Reinstall status-checker
Start
Stop
Show 10 per page
1
2
3

Manage Kerberos

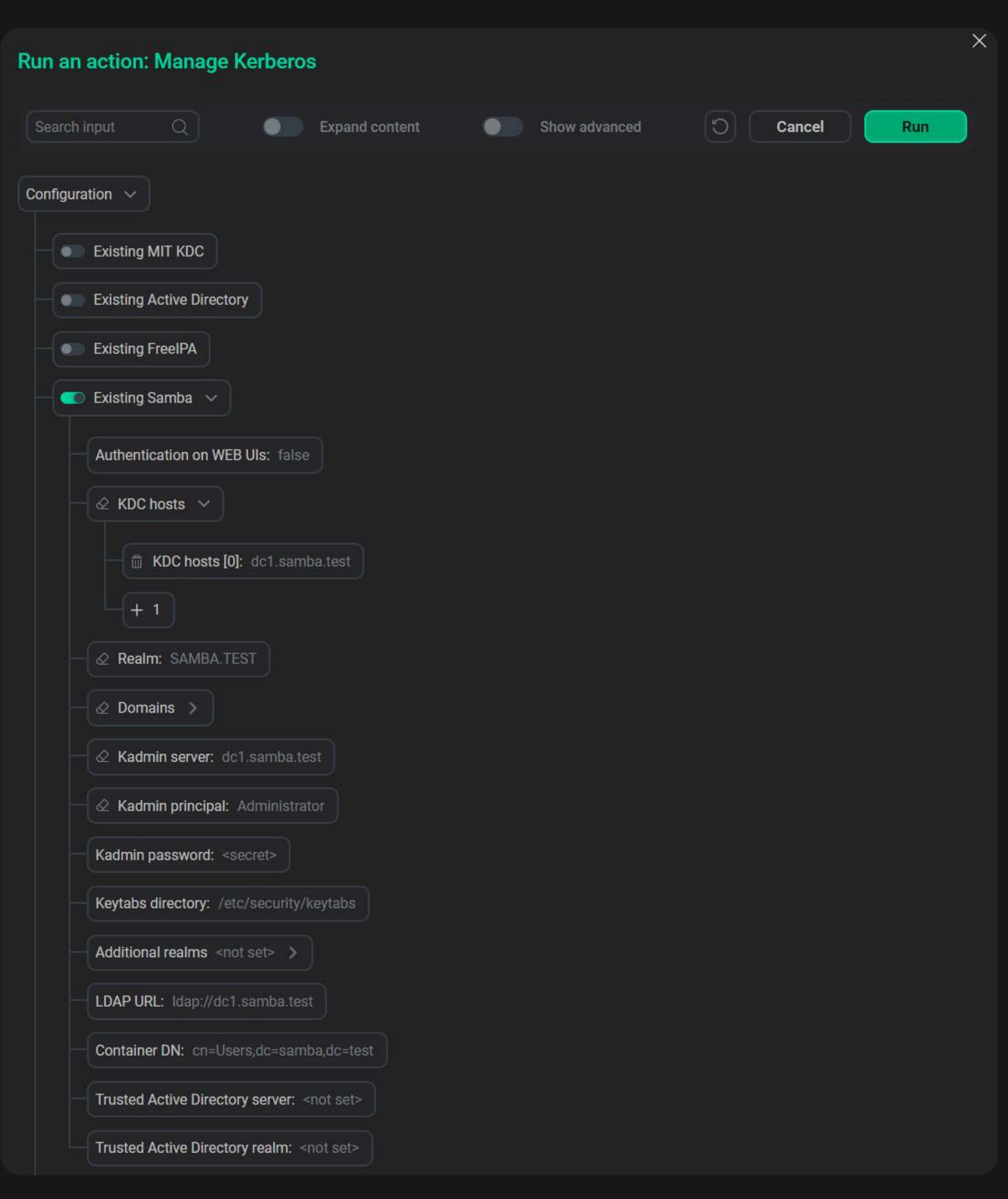
2. В появившемся окне выберите опцию ***Existing Samba***.


Run an action: Manage Kerberos
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Existing Samba
KDC hosts
Domains
Additional realms
Custom kerberization settings

Authentication on WEB UIs: false
Realm: <not set>
Kadmin server: <not set>
Kadmin principal: <not set>
Kadmin password: <not set>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
LDAP URL: <not set>
Container DN: <not set>
Trusted Active Directory server: <not set>
Trusted Active Directory realm: <not set>

Выбор соответствующей опции

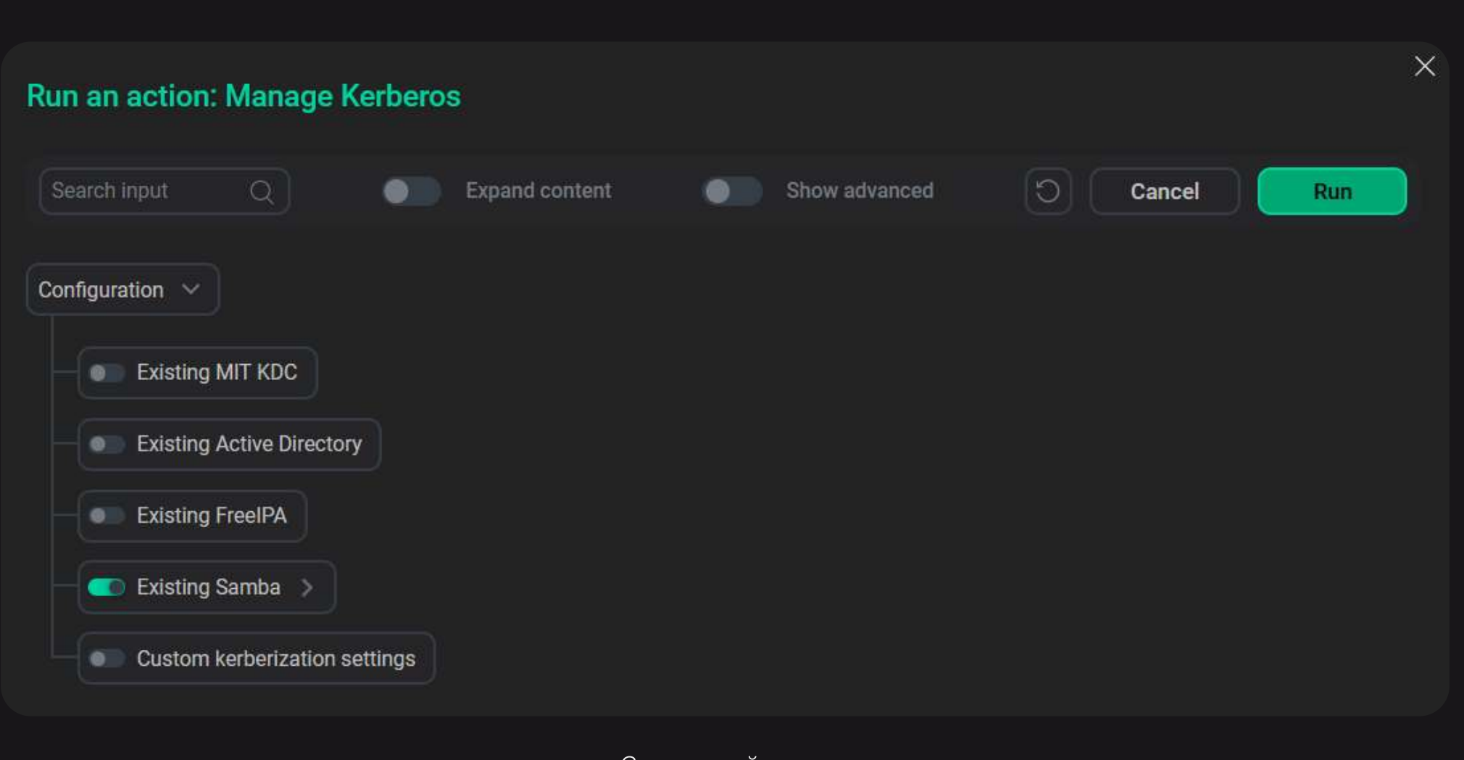
3. Заполните Kerberos-параметры Samba и приступайте к настройке Kerberos в кластере.


Run an action: Manage Kerberos
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Existing Samba
KDC hosts
+ 1
Domains
Additional realms
Custom kerberization settings

Authentication on WEB UIs: false
KDC hosts [0]: dc1.samba.test
Realm: SAMBA.TEST
Kadmin server: dc1.samba.test
Kadmin principal: Administrator
Kadmin password: <secret>
Keytabs directory: /etc/security/keytabs
Additional realms <not set>
LDAP URL: ldap://dc1.samba.test
Container DN: cn=Users,dc=samba,dc=test
Trusted Active Directory server: <not set>
Trusted Active Directory realm: <not set>

Kerberos-параметры Samba

4. Нажмите ***Run*** и дождитесь окончания действия. Затем приступайте к настройке Kerberos в кластере.


Run an action: Manage Kerberos
Search input
Expand content
Show advanced
Cancel
Run
Configuration
Existing MIT KDC
Existing Active Directory
Existing FreeIPA
Existing Samba
Custom kerberization settings

Запуск действия

Для проверки запустите команду `kinit`. Если она работает некорректно со сгенерированными keytab-файлами в вашей системе, добавьте следующие опции шифрования паролей в секцию `libdefaults` кастомного параметра `krb5.conf` в конфигурации кластера в ADCM:

```
[libdefaults]
  permitted_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 arcfour-hmac rc4-hmac arcfour-hmac-md5
  default_tkt_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 rc4-hmac
  default_tgs_enctypes = aes256-cts-hmac-sha1-96 aes128-cts-hmac-sha1-96 rc4-hmac
```

Особенности реализации

В представленном скрипте принципалы создаются особым способом (как и в бандле) по ряду причин:

- Чтобы работала аутентификация по keytab-файлам, в качестве UPN (User Principal Name) требуется передать SPN (Service Principal Name). Это необходимо из-за того, что Samba не позволяет искать принципалов по UPN при выполнении `kinit`.
- Так как существует ограничение UPN в одну запись, приходится создавать SPN для каждого сервиса по количеству хостов.
- По умолчанию Samba при работе с принципалами и keytab-файлами обращается к локальной базе, а не к удаленному контроллеру домена. Чтобы Samba обращалась к контроллеру, при запуске действий нужно использовать аргумент `-H ldap://$SAMBA_SERVER`.

У такого подхода есть свои плюсы: нет необходимости добавлять компьютер в домен и настраивать локальные реплики баз данных Samba для создания сервисных пользователей и принципалов. Также аутентификация принципалов происходит по отдельным keytab-файлам для каждого, а не по одному keytab-файлу на компьютер.

Скрипт для генерации keytab-файлов

```
#!/bin/bash
# Script for adding services and keytabs to Cluster hosts
# Edit  HOSTS as it will be suitable for your case.
# EDIT your Samba parameters
# Select services according to the bundle

# Samba credentials
SAMBA_USER='Administrator'
SAMBA_PASSWORD='adminPassword'
SAMBA_SERVER='<samba_server>'
REALM='<samba_realm>'
SAMBA_HOST='<samba_host>'
USER_PASSWORD='userPassword'
OU_NAME='CN=Users,DC=samba,DC=test'

declare -a HOSTS

SSH_OPTS='-o GlobalKnownHostsFile=/dev/null -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no'

HOSTS=(
    "List your hosts here"
)
encoded_password=$(echo -n "\"$USER_PASSWORD\"" | iconv -f UTF-8 -t UTF-16LE | base64)

# ADH services
SERVICES=(HTTP airflow flink hbase-phoenix_queryserver hbase-thrift2 hbase hdfs-datanode hdfs-journalnode hdfs-namenode
hdfs-zkfc hdfs hive httpfs livy mapreduce-historyserver solr spark sqoop yarn-nodemanager yarn-resourcemanager yarn-
timelineserver yarn zeppelin zookeeper impala ssm ssm-agent kyuubi)

mkdir /tmp/keytabs

for HOST in ${HOSTS[@]}; do
    echo "Set hostname"
        ssh $SSH_OPTS $HOST "sudo hostnamectl --static set-hostname $HOST"
    echo "Create keytab dir"
            ssh $SSH_OPTS $HOST "sudo mkdir -p /etc/security/keytabs"
    for SERVICE in ${SERVICES[@]}; do
        echo "Create service user"
        ssh $SSH_OPTS $SAMBA_HOST "sudo useradd $SERVICE"
        echo "Create service principal"
        echo "Create add.ldif"
        ssh $SSH_OPTS $SAMBA_HOST "cat <<EOF > /tmp/add.ldif
DN: CN=${SERVICE}/${HOST},${OU_NAME}
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
accountExpires: 0
sAMAccountName: ${SERVICE}-${HOST}
userPrincipalName: ${SERVICE}/${HOST}@${REALM}
servicePrincipalName: ${SERVICE}/${HOST}
distinguishedName: CN=${SERVICE}/${HOST},${OU_NAME}
userAccountControl: 514

DN: CN=${SERVICE}/${HOST},${OU_NAME}
changetype: modify
replace: unicodePwd
unicodePwd:: $encoded_password

DN: CN=${SERVICE}/${HOST},${OU_NAME}
changetype: modify
replace: userAccountControl
userAccountControl: 66048

EOF"

        echo "sudo ldbmodify -H ldap://$SAMBA_SERVER /tmp/add.ldif -U $SAMBA_USER --password=$SAMBA_PASSWORD"
        ssh $SSH_OPTS $SAMBA_HOST "sudo ldbmodify -H ldap://$SAMBA_SERVER /tmp/add.ldif -U $SAMBA_USER --
password=$SAMBA_PASSWORD"

        echo "Delete keytab"
        ssh $SSH_OPTS $SAMBA_HOST "sudo rm -f /tmp/$SERVICE.service.keytab"

        echo "Extract keytab"
        TMP_SCRIPT="/tmp/extract_keytab_${SERVICE}_${HOST}.sh"
        KTUTIL_SCRIPT="/tmp/ktutil_commands_${SERVICE}_${HOST}.txt"
        cat <<EOT > $TMP_SCRIPT
#!/bin/bash
cd /tmp
sudo ktutil < $KTUTIL_SCRIPT
EOT

        cat <<EOT > $KTUTIL_SCRIPT
add_entry -password -p ${SERVICE}/${HOST}@${REALM} -k 1 -e aes256-cts-hmac-sha1-96
${USER_PASSWORD}
add_entry -password -p ${SERVICE}/${HOST}@${REALM} -k 1 -e aes128-cts-hmac-sha1-96
${USER_PASSWORD}
add_entry -password -p ${SERVICE}/${HOST}@${REALM} -k 1 -e arcfour-hmac
${USER_PASSWORD}
wkt ${SERVICE}.service.keytab
quit
EOT

        scp $SSH_OPTS $TMP_SCRIPT $SAMBA_HOST:$TMP_SCRIPT
        scp $SSH_OPTS $KTUTIL_SCRIPT $SAMBA_HOST:$KTUTIL_SCRIPT
        ssh $SSH_OPTS $SAMBA_HOST "chmod +x $TMP_SCRIPT && $TMP_SCRIPT && rm $TMP_SCRIPT && rm $KTUTIL_SCRIPT"
        rm $TMP_SCRIPT
        rm $KTUTIL_SCRIPT

        echo "Try kinit"
        ssh $SSH_OPTS $SAMBA_HOST "sudo klist -k /tmp/$SERVICE.service.keytab"
        echo "Change permission"
        ssh $SSH_OPTS $SAMBA_HOST "sudo chmod 777 /tmp/${SERVICE}.service.keytab"
        echo "Export keytab to local"
        scp $SAMBA_HOST:/tmp/${SERVICE}.service.keytab /tmp/keytabs/${SERVICE}.service.keytab
        echo "Export keytab to host"
        scp /tmp/keytabs/${SERVICE}.service.keytab $HOST:/tmp/${SERVICE}.service.keytab
        ssh $SSH_OPTS $HOST "sudo mv /tmp/${SERVICE}.service.keytab /etc/security/keytabs/${SERVICE}.service.keytab"
        echo "Clean up local temporary keytab"
        rm /tmp/keytabs/$SERVICE.service.keytab
    done

    echo "Change owner and group"

    ssh $SSH_OPTS $HOST sudo chown zookeeper:zookeeper /etc/security/keytabs/zookeeper*
    ssh $SSH_OPTS $HOST sudo chown hdfs:hadoop /etc/security/keytabs/hdfs*
    ssh $SSH_OPTS $HOST sudo chown hdfs:hadoop /etc/security/keytabs/HTTP*
    ssh $SSH_OPTS $HOST sudo chown yarn:hadoop /etc/security/keytabs/yarn*
    ssh $SSH_OPTS $HOST sudo chown hbase:hadoop /etc/security/keytabs/hbase*
    ssh $SSH_OPTS $HOST sudo chown flink:flink /etc/security/keytabs/flink*
    ssh $SSH_OPTS $HOST sudo chown sqoop:sqoop /etc/security/keytabs/sqoop*
    ssh $SSH_OPTS $HOST sudo chown zeppelin:zeppelin /etc/security/keytabs/zeppelin*
    ssh $SSH_OPTS $HOST sudo chown spark:spark /etc/security/keytabs/spark*
    ssh $SSH_OPTS $HOST sudo chown hive:hive /etc/security/keytabs/hive*
    ssh $SSH_OPTS $HOST sudo chown livy:hadoop /etc/security/keytabs/livy*
    ssh $SSH_OPTS $HOST sudo chown httpfs:httpfs /etc/security/keytabs/httpfs*
    ssh $SSH_OPTS $HOST sudo chown mapred:hadoop /etc/security/keytabs/mapreduce-historyserver*
    ssh $SSH_OPTS $HOST sudo chown solr:solr /etc/security/keytabs/solr*
    ssh $SSH_OPTS $HOST sudo chown airflow:airflow /etc/security/keytabs/airflow*
    ssh $SSH_OPTS $HOST sudo chown impala:impala /etc/security/keytabs/impala*

    echo "Change permissions"

    ssh $SSH_OPTS $HOST "sudo chmod -R 777 /etc/security/keytabs/"

done
```

Kerberos-параметры Samba

Параметр	Описание
Authentication on WEB UIs	Включает аутентификацию Kerberos для веб-интерфейсов
KDC hosts	Один или несколько хостов контроллера домена
Realm	Kerberos realm — сеть, содержащая KDC-хосты и множество клиентов
Domains	Домены, в которых находятся хосты
Kadmin server	Адрес хоста, на котором запущен `kadmin`
Kadmin principal	Имя принципала для подключения через `kadmin`, например `admin@RU-CENTRAL1.INTERNAL`
Kadmin password	Пароль принципала для подключения через `kadmin`
Keytabs directory	Директория keytab-файла, содержащего записи одного или нескольких принципалов и их ключи
Additional realms	Дополнительные сети Kerberos realm
LDAP URL	LDAP URL состоит из префикса `ldap://` или `ldaps://`, имени хоста или IP-адреса и порта LDAP-сервера
Container DN	Отличительное имя контейнера
Trusted Active Directory server	Сервер с DC для односторонней доверенности
Trusted Active Directory realm	Область доверия между realm

Подключение к керберизованному Hive из DBeaver

Константин Алпашкин

Назад к содержанию

1. Установите DBeaver CE.

ПРИМЕЧАНИЕ

Используйте DBeaver CE версии 21.3.4 и выше.

2. После установки DBeaver CE необходимо скопировать файл */etc/krb5.conf* с Kerberos-сервера в папку *C:\Program Files\DBeaver\jre\conf\security* вашей машины Windows.
3. Отредактируйте файл *krb5.conf*, удалив все символы перед секцией `[libdefaults]` и закомментировав строку `renew_lifetime = 7d`.
4. Чтобы подключиться к Hive, необходимо также передать основной файл *keytab*. Создайте файл *keytab*, запустив утилиту `kadmin` на сервере Kerberos и выполнив следующую команду:

```
$ xst -norandkey -k <file_name> <principal_name>
```

Например:

```
$ xst -norandkey -k hive.keytab hive/hive-kerberos.ru-central1.internal@ADREALM.IO
```

Файл *krb5.conf* должен иметь следующий вид:

krb5.conf

```
[libdefaults]
default_realm = ADREALM.IO
dns_lookup_realm = false
dns_lookup_kdc = false
ticket_lifetime = 24h
#renew_lifetime = 7d
forwardable = true

[realms]
ADREALM.IO = {
kdc = kerberos-server.ru-central1.internal
admin_server = kerberos-server.ru-central1.internal
}

[domain_realm]
.adrealm.io = ADREALM.IO
adrealm.io = ADREALM.IO
.adrealm.io = ADREALM.IO
```

5. Создайте файл *jaas.conf* в папке DBeaver. В этом файле необходимо задать путь к основному файлу *keytab*, используемому для подключения к Hive. Также необходимо указать имя принципала.
Файл *jaas.conf* должен иметь следующий вид:

jaas.conf

```
Client {
com.sun.security.auth.module.Krb5LoginModule required
doNotPrompt=true
useKeyTab=true
keyTab="C:/Users/Administrator/Downloads/hive.keytab"
useTicketCache=false
renewTGT=false
principal="hive/hive-kerberos.ru-central1.internal@ADREALM.IO";
};
```

6. Создайте файл *dbeaver.ini*, расположенный в папке DBeaver, и добавьте пути к файлам *jaas.conf* и *krb5.conf*.

ПРИМЕЧАНИЕ

Файлы *jaas.conf* и *dbeaver.ini* должны находиться в одной папке.

dbeaver.ini

```
-startup
plugins/org.eclipse.equinox.launcher_1.6.200.v20210416-2027.jar
--launcher.library
plugins/org.eclipse.equinox.launcher.win32.win32.x86_64_1.2.200.v20210429-1609
-vmargs
-XX:+IgnoreUnrecognizedVMOptions
--add-modules=ALL-SYSTEM
-Dosgi.requiredJavaVersion=11
-Xms64m
-Xmx1024m
-Djava.security.krb5.conf=C:\Program Files\DBeaver\jre\conf\security\krb5.conf
-Djava.security.auth.login.config=jaas.conf
-Djavax.security.auth.useSubjectCredsOnly=false
```

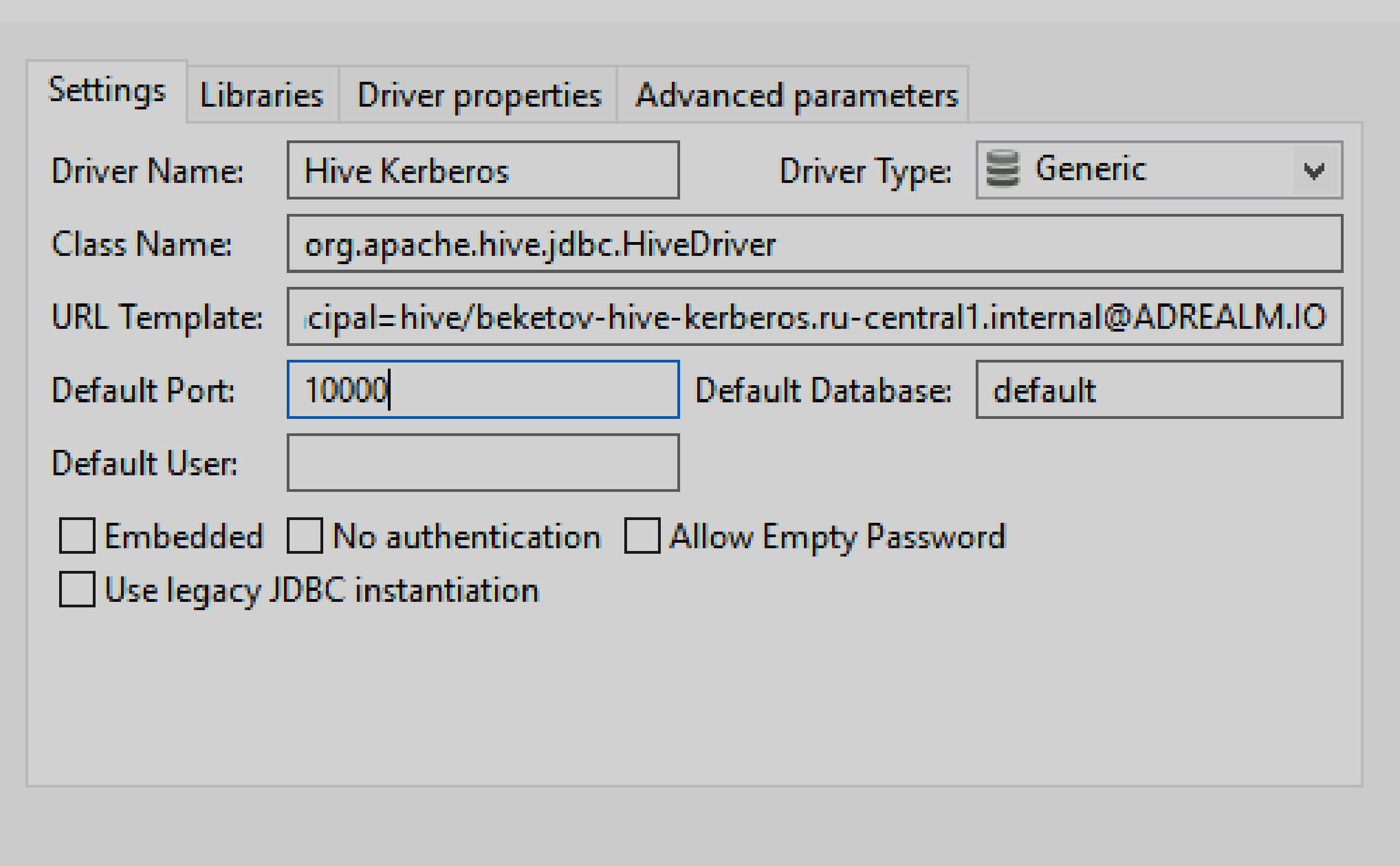
7. Получите аутентификационный тикет, используя файл *keytab*. Для этого перейдите в каталог *C:\Program Files\DBeaver\jre\bin* и выполните следующую команду:

```
./kinit.exe -k -t keytab-file-path principal_name
```

Пример вывода:

```
New ticket is stored in cache file C:\Users\Administrator\krb5cc_Administrator
```

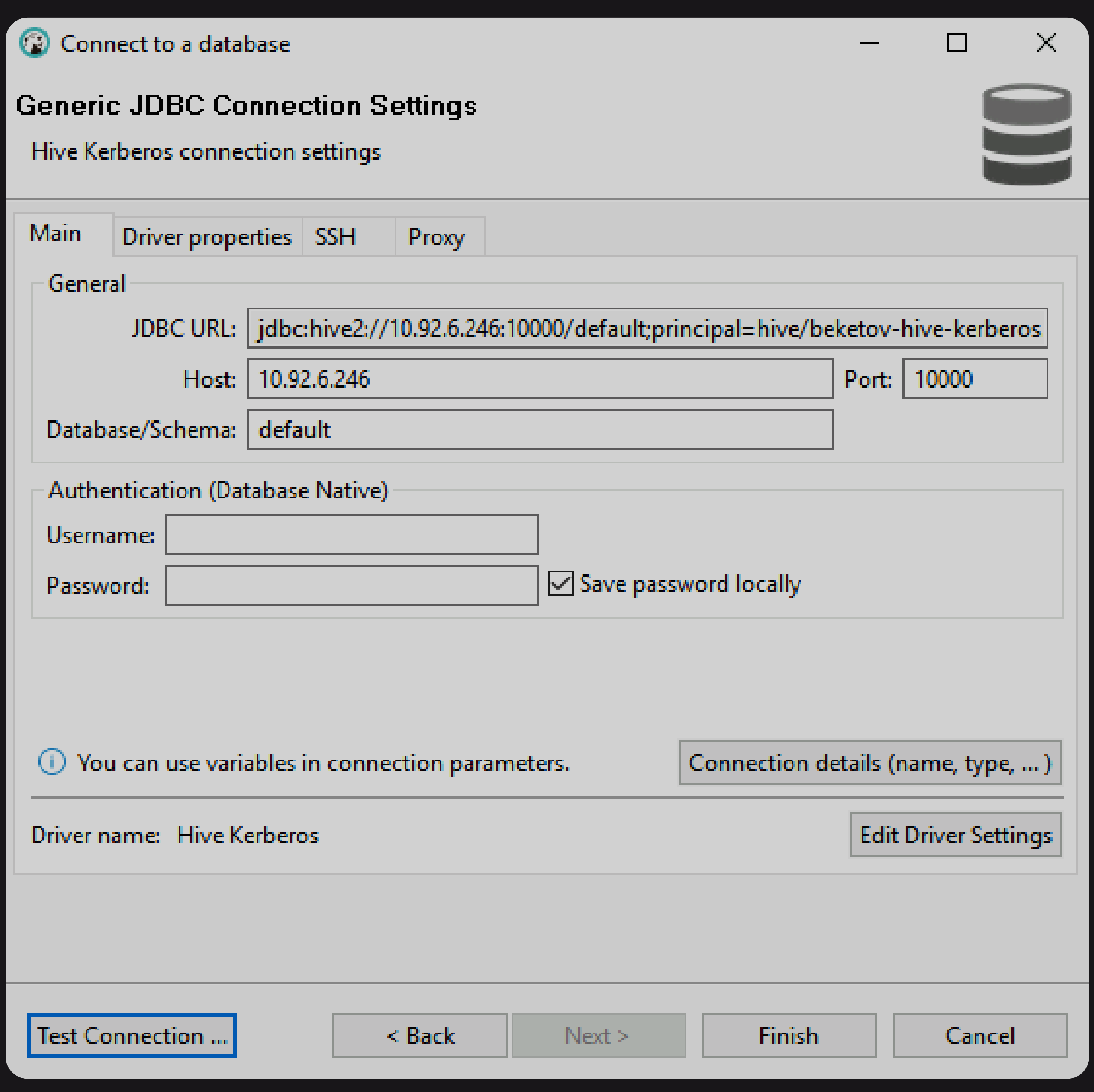
8. Скачайте JBDC-драйвер и распакуйте его в папку DBeaver.
9. Для запуска DBeaver перейдите в меню ***Database*** и выберите ***Driver Manager***.
10. Выберите ***New***.
Заполните необходимые поля:
    - ***Driver name***: Hive Kerberos;
    - ***URL Template***: jdbc:hive2://{host}:{port}/{database};principal=hive/hive-kerberos.ru-central1.internal@ADREALM.IO;
    - ***Default port***: 10000;
    - ***Default Database***: default.
11. Перейдите в ***Libraries*** и выберите файл драйвера *hive-jdbc-3.1.1-arenadata-standalone.jar*, который вы распаковали ранее. Выберите ***FindClass*** → ***org.apache.hive.jdbc.HiveDriver***. Кликните ***Ok***.


Create new driver

Driver Name: Hive Kerberos
Driver Type: Generic
Class Name: org.apache.hive.jdbc.HiveDriver
URL Template: cipal=hive/beketov-hive-kerberos.ru-central1.internal@ADREALM.IO
Default Port: 10000
Default Database: default
Default User:
☐ Embedded ☐ No authentication ☐ Allow Empty Password
☐ Use legacy JDBC instantiation

Создание драйвера

12. Создайте новое подключение. Для этого:
    1. Перейдите в ***Database*** → ***New Database Connection***.
    2. Выберите ***Hive Kerberos***.
    3. В ***Host*** укажите IP-адрес сервера Hive.


Connect to a database

Generic JDBC Connection Settings
Hive Kerberos connection settings

General
JDBC URL: jdbc:hive2://10.92.6.246:10000/default;principal=hive/beketov-hive-kerberos
Host: 10.92.6.246
Port: 10000
Database/Schema: default

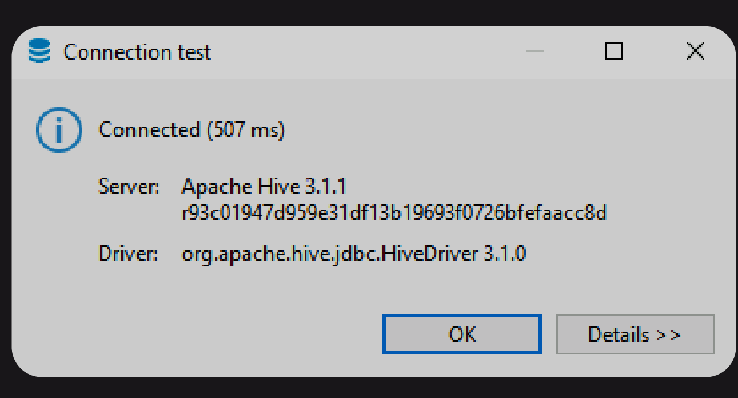
Authentication (Database Native)
Username:
Password:
☑ Save password locally

You can use variables in connection parameters.

Driver name: Hive Kerberos

Создание нового соединения

    4. Кликните ***Test Connection*** для проверки соединения.

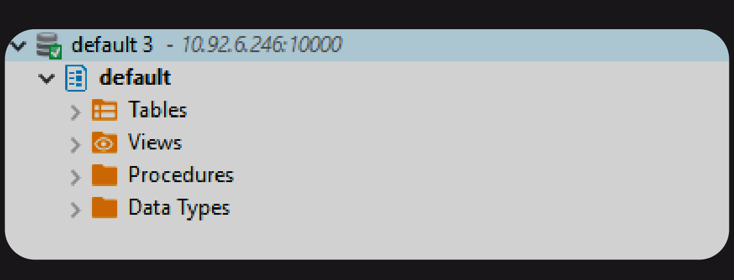

Connection test

Connected (507 ms)

Server: Apache Hive 3.1.1
r93c01947d959e31df13b19693f0726bfefaacc8d
Driver: org.apache.hive.jdbc.HiveDriver 3.1.0

Проверка соединения

13. Кликните ***Finish***.



Доступная база данных Hive

Аутентификация SPNEGO

Сергей Тихомиров

Содержание

Назад к содержанию

Настройка браузера

Поддерживаемые сервисы

SPNEGO (Simple and Protected GSS API Negotiation Mechanism) — механизм, применяемый для определения технологии защиты, которая будет использоваться при безопасном подключении. В частности он используется, когда ни сервер, ни клиент не знают о поддерживаемых протоколах безопасности друг друга. SPNEGO использует протокол для определения доступных механизмов GSS API, выбирает один из них и делегирует ему все операции, связанные с безопасностью.

Когда сервер получает запрос от браузера, он может запросить, чтобы тот использовал для аутентификации протокол SPNEGO. Этот протокол осуществляет Kerberos-аутентификацию через HTTP. Детально процесс выглядит так:

1. На запрос с заголовком `WWW-Authorization: Negotiate` сервер возвращает код `401 Unauthorized`.
2. Браузер связывается с сервисом выдачи тикетов (Ticket Granting Service, TGS) для получения тикета.
3. Полученный тикет оборачивается в закодированный в base64 токен SPNEGO и отправляется на сервер с последующим запросом аутентификации.
4. Токен разворачивается, и сервер аутентифицирует тикет.

Настройка браузера

Некоторым браузерам требуется дополнительная настройка для Kerberos-аутентификации с помощью SPNEGO. Инструкции для самых популярных браузеров представлены ниже.

Mozilla Firefox

1. В строке поиска введите команду `about:config` и запустите её.
2. Измените следующие параметры, присвоив им соответствующие значения:
    - `network.negotiate-auth.trusted-uris` — ваш домен, защищенный Kerberos (например, `.ru-central1.internal`).
    - `network.automatic-ntlm-auth.trusted-uris` — ваш домен, защищенный Kerberos (например, `.ru-central1.internal`).
    - `network.auth.use-sspi` — `false`.
3. Перезапустите браузер и проследуйте к защищенному ресурсу.

Google Chrome

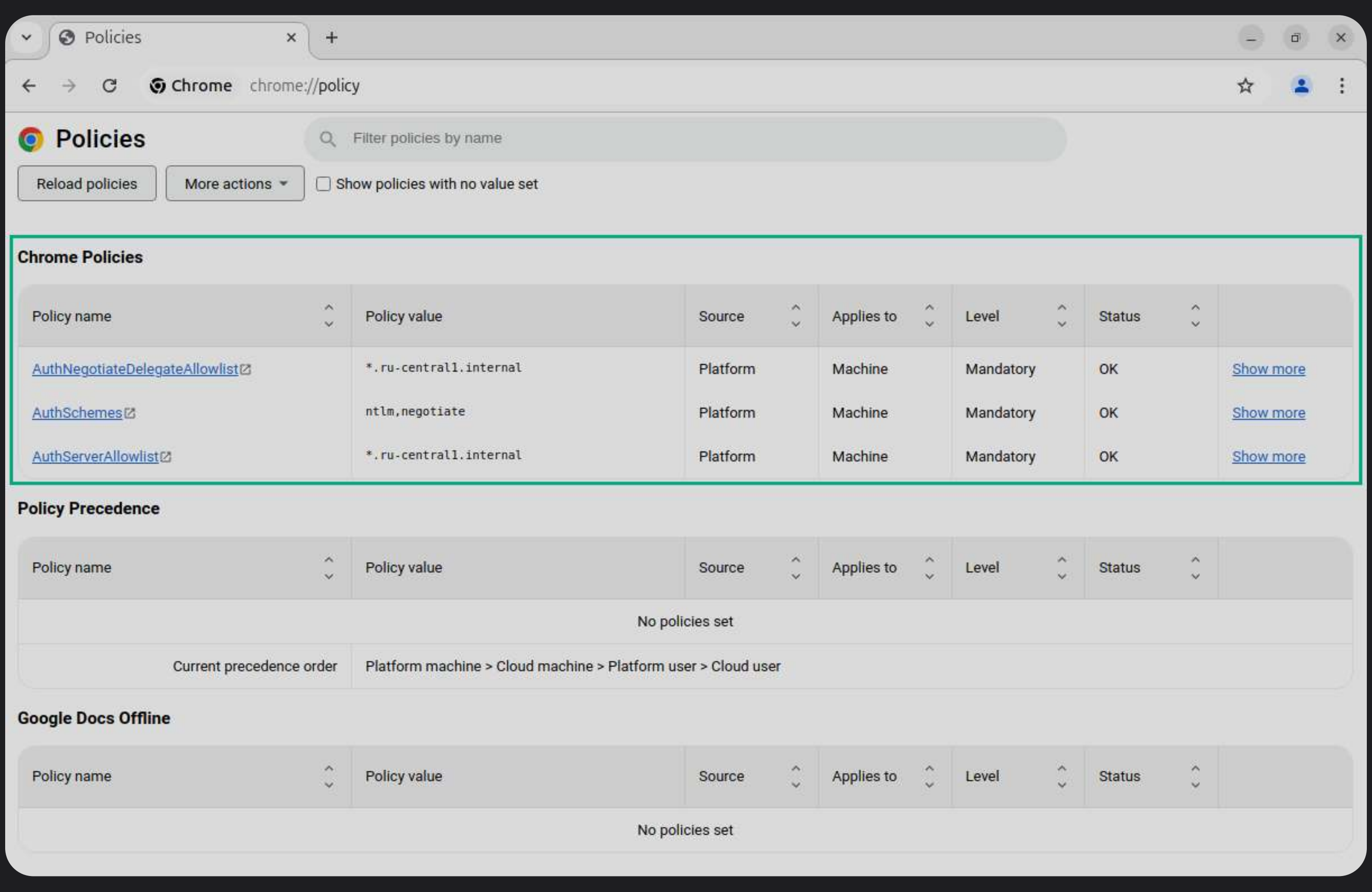
Для корректного процесса аутентификации требуется установить дополнительные политики для браузера.

- В Windows откройте редактор регистра и раскройте ***HKEY_LOCAL_MACHINE*** → ***SOFTWARE*** → ***Google*** → ***Chrome***. Для добавления новых политик нажмите правой кнопкой мыши в пустом месте, затем ***New*** → ***String value*** и введите название политики и её значение:
    - `AuthSchemes` — `digest,ntlm,negotiate`.
    - `AuthServerAllowlist` — `<domain>`. Если этот параметр не установлен, то Google Chrome проверит, есть ли сервер, запрашивающий аутентификацию в интранете. Если есть, то процесс продолжится, а в противном случае — Google Chrome будет игнорировать запросы на аутентификацию, поступающие от этого сервера.
    - `AuthNegotiateDelegateAllowlist` — `<domain>`. Если этот параметр не установлен, то Google Chrome не будет передавать учетные данные пользователей даже серверам, находящимся в интранете.
      В описаниях параметров выше `<domain>` — ваш домен, защищенный Kerberos (e.g. `.ru-central1.internal`).

      Обратите внимание, что хост с Windows обязан находиться в домене, в то время как для Linux это необязательно.
- В Linux перейдите в директорию (создайте недостающие папки) */etc/opt/chrome/policies/managed* и создайте файл политики (например, *test_policy.json*) со следующим содержимым:

```
{
  "AuthServerAllowlist": "<domain>",
  "AuthNegotiateDelegateAllowlist": "<domain>",
  "AuthSchemes": "digest,ntlm,negotiate"
}
```

Правильность создания политик можно проверить, запустив команду `chrome://policy` в поисковой строке.



Chrome Policies

Policy name	Policy value	Source	Applies to	Level	Status	
AuthNegotiateDelegateAllowlist	`*.ru-central1.internal`	Platform	Machine	Mandatory	OK	Show more
AuthSchemes	`ntlm,negotiate`	Platform	Machine	Mandatory	OK	Show more
AuthServerAllowlist	`*.ru-central1.internal`	Platform	Machine	Mandatory	OK	Show more

Policy Precedence

Policy name	Policy value	Source	Applies to	Level	Status
No policies set					
Current precedence order	Platform machine > Cloud machine > Platform user > Cloud user				

Google Docs Offline

Policy name	Policy value	Source	Applies to	Level	Status
No policies set					

Политики Google Chrome для SPNEGO

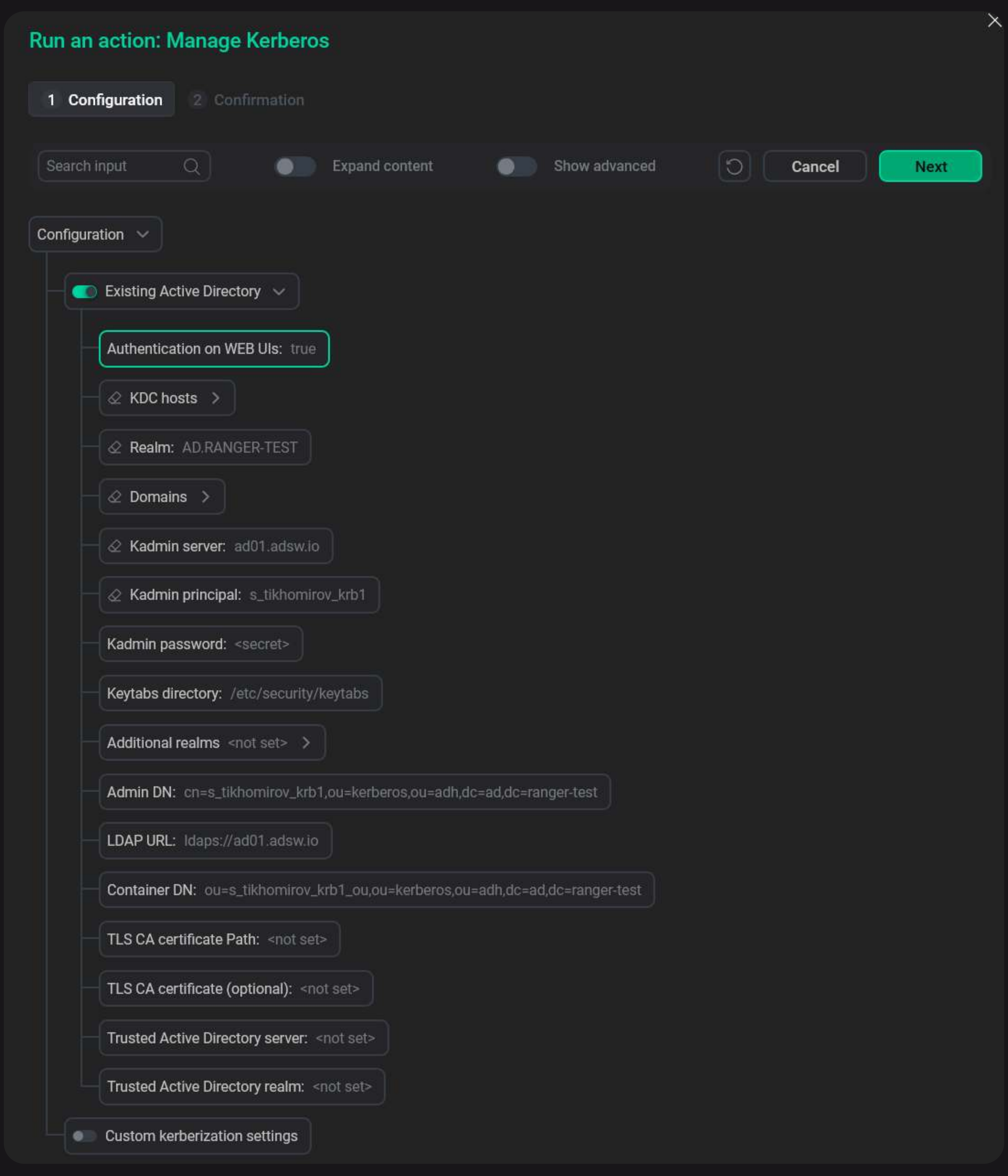
При правильной настройке вы сможете получить доступ к запрашиваемым ресурсам с помощью тикета Kerberos, выданного в вашей системе. В противном случае вы продолжите получать ошибку `401 Unauthorized`.

Поддерживаемые сервисы

Следующие сервисы ADH поддерживают аутентификацию в веб-интерфейсах с помощью SPNEGO:

- Flink
- HDFS
- Hive
- HUE
- Impala
- Kyuubi
- Spark3
- SSM
- YARN

Чтобы включить для них механизм SPNEGO (если не указано иначе в инструкции для сервиса), необходимо установить флаг ***Authentication on WEB UIs*** при активации Kerberos.



Authentication on WEB UIs: true

Realm: AD.RANGER-TEST

Kadmin server: ad01.adsw.io

Kadmin principal: s_tikhomirov_krb1

Kadmin password: <secret>

Keytabs directory: /etc/security/keytabs

Additional realms <not set>

Admin DN: cn=s_tikhomirov_krb1,ou=kerberos,ou=adh,dc=ad,dc=ranger-test

LDAP URL: ldaps://ad01.adsw.io

Container DN: ou=s_tikhomirov_krb1_ou,ou=kerberos,ou=adh,dc=ad,dc=ranger-test

TLS CA certificate Path: <not set>

TLS CA certificate (optional): <not set>

Trusted Active Directory server: <not set>

Trusted Active Directory realm: <not set>

Параметр "Authentication on WEB UIs"

Требования к SSL-сертификатам

Константин Алпашкин

Назад к содержанию

Для реализации шифрования между узлами выполните шаги, описанные ниже.

1. Создайте сертификат для каждого узла, используя алгоритм RSA. Длина ключа должна быть 2048 бит, значение CN должно содержать FQDN соответствующего узла. Не рекомендуется использовать алгоритм DSA, а также алгоритм ECDSA из-за того, что этот алгоритм не очень широко поддерживается сертификационными центрами.
   Длина ключа, равная 2048 битам, является необходимой и достаточной для реализации надежного шифрования.
2. Создайте хранилище ключей *trustore.jks*, которое будет содержать необходимые сертификаты. Путь к этому хранилищу должен быть указан для запуска SSL.

> **ПРИМЕЧАНИЕ**
>
> - Сертификаты должны быть подписаны ключом, принадлежащим одному из доверенных корневых сертификатов, включенных в хранилище сертификатов Java.
> - Самоподписанные сертификаты разрешены. В этом случае корневой сертификат должен быть добавлен в хранилище *trustore.jks*.
> - Nginx требует OpenSSL-сертификаты. Если ваш кластер использует Nginx (с Hive TezUI или Airflow), сгенерируйте ключи и сертификаты OpenSSL. OpenSSL-сертфикат для вашего хоста необходимо добавить в хранилище *ca-bundle.crt*.
> - Учетная запись, используемая для установки сертификатов, должна иметь права на запись в следующие директории:
>   - */etc/pki/tls/certs/*
>   - */etc/pki/java/*
> - В качестве alias ключа в *keystore.jks* используйте FQDN, а не короткое имя.

3. Включите трафик между узлами в соответствии с приведенной ниже таблицей.
   Приведенный ниже скрипт создает все необходимые сущности и может быть использован для справки:

```bash
#!/bin/bash

# Script for generate and import self-signet certificates to java keystore
# Edit NUMHOSTS and HOSTS as it will be suitable for your case.


declare -a NUMHOSTS
declare -a HOSTS

SSH_OPTS='-o GlobalKnownHostsFile=/dev/null -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no'
NUMHOSTS=(1 2 3)
HOSTS=$(for i in "${NUMHOSTS[@]}"; do echo "adh-$i.ru-central1.internal"; done)

echo Generate keystore.jks on each host
for HOST in $HOSTS; do
        ssh $SSH_OPTS $HOST "keytool -genkey -noprompt -keyalg RSA -alias $HOST -dname \"CN=$HOST, OU=AD, O=AD, L=MSK, S=MO,
C=RU\" -keystore /tmp/keystore.jks -storepass bigdata -keypass bigdata -validity 360 -keysize 2048";
done

echo
echo Export certificates
for HOST in $HOSTS;do
        ssh $SSH_OPTS $HOST "keytool -export -file /tmp/$HOST.crt -keystore /tmp/keystore.jks -storepass bigdata -alias $HOST -
rfc";
done

echo
echo Collect all certificates
for HOST in $HOSTS; do
        scp $SSH_OPTS $HOST:/tmp/$HOST.crt /tmp/
done

echo
echo Transfer certificates on hosts
for HOST in $HOSTS; do
        scp $SSH_OPTS /tmp/*.crt $HOST:/tmp/
done

echo
echo Import certificates on each host
for HOST in $HOSTS; do
        ssh $SSH_OPTS $HOST "for CERT in $(echo ${HOSTS[*]}); do
            keytool -import -noprompt -alias \$CERT -file /tmp/\$CERT.crt -keystore /tmp/truststore.jks -storepass bigdata;
            sudo bash -c \"cat /tmp/\$CERT.crt >> /etc/pki/tls/certs/ca-bundle.crt\";
        done";
done

echo
echo Import truststore to Java CA store
for HOST in $HOSTS; do
        ssh $SSH_OPTS $HOST "sudo keytool -importkeystore -noprompt -srckeystore /tmp/truststore.jks -destkeystore
/etc/pki/java/cacerts -deststorepass changeit -srcstorepass bigdata"
done
```

Сопоставление портов и служб HTTP/HTTPS

Сервис	Компонент/Сервер	Порт	Протокол	Описание
Airflow	Server	8080/8080	HTTP/HTTPS	Порт веб-интерфейса Airflow Server
	Server/Worker	8793/8793	HTTP/HTTPS	Порт для доступа к Celery Worker API
	Server/Flower	5555/5555	HTTP/HTTPS	Порт веб-интерфейса Flower
Flink	Job Manager	8081/8081	HTTP/HTTPS	Порт веб-интерфейса Flink Job Manager
HBase	HBase Region	16030/16030	HTTP/HTTPS	Порт веб-интерфейса HBase Region Server
	Phoenix Query Server	8765/8765	HTTP/HTTPS	Порт для доступа к API
	HBase REST Server	60080/60080	HTTP/HTTPS	Порт для доступа к API
	HBase REST Server	8085/8085	HTTP/HTTPS	Web UI Port
	HBase Master	16010/16010	HTTP/HTTPS	Порт веб-интерфейса HBase Master
HBaseThrift2Server	Trift2 Server	9095/9095	HTTP/HTTPS	Порт веб-интерфейса Thrift2 Server
HDFS	NameNode	9870/9871	HTTP/HTTPS	Порт веб-интерфейса NameNode
	DataNode	9864/9865	HTTP/HTTPS	Порт веб-интерфейса DataNode
	JournalNode	8480/8481	HTTP/HTTPS	Порт веб-интерфейса JournalNode
	HTTPfs server	14000/14000	HTTP/HTTPS	Порт для доступа к HTTPfs API
Hive	Hive Server	10002/10002	HTTP/HTTPS	Порт веб-интерфейса Hive Server
	Tez	9999/9999	HTTP/HTTPS	Порт веб-интерфейса Tez
Knox	Knox Gateway	8443	HTTPS	Порт шлюза
Ranger	Ranger Admin	6080/6182	HTTP/HTTPS	Порт для доступа к API и веб-интерфейсу Ranger Admin
	Ranger KMS	9292/9393	HTTP/HTTPS	Порт для Ranger KMS
Solr	Solr Server	8983/8985	HTTP/HTTPS	Порт для доступа к веб-интерфейсу и API Solr Server
Spark	History Server	18082/18082	HTTP/HTTPS	Порт веб-интерфейса History Server
	Thrift Server	4040/4040	HTTP/HTTPS	Порт веб-интерфейса Thrift Server
	Livy Server	8998/8998	HTTP/HTTPS	Порт веб-интерфейса Livy Server
YARN	Resource Manager	8088/8090	HTTP/HTTPS	Порт веб-интерфейса Resource Manager
	Node Manager	8042/8044	HTTP/HTTPS	Порт веб-интерфейса Node Manager
	MapReduce History Server	19888/19890	HTTP/HTTPS	Порт веб-интерфейса History Server
	Timeline Server	8188/8190	HTTP/HTTPS	Порт веб-интерфейса Timeline Server
Zeppelin	Server	8180/8180	HTTP/HTTPS	Порт веб-интерфейса Zeppelin

Установка версии TLS для кластера ADH

Константин Аллашкин

Содержание

Назад к содержанию

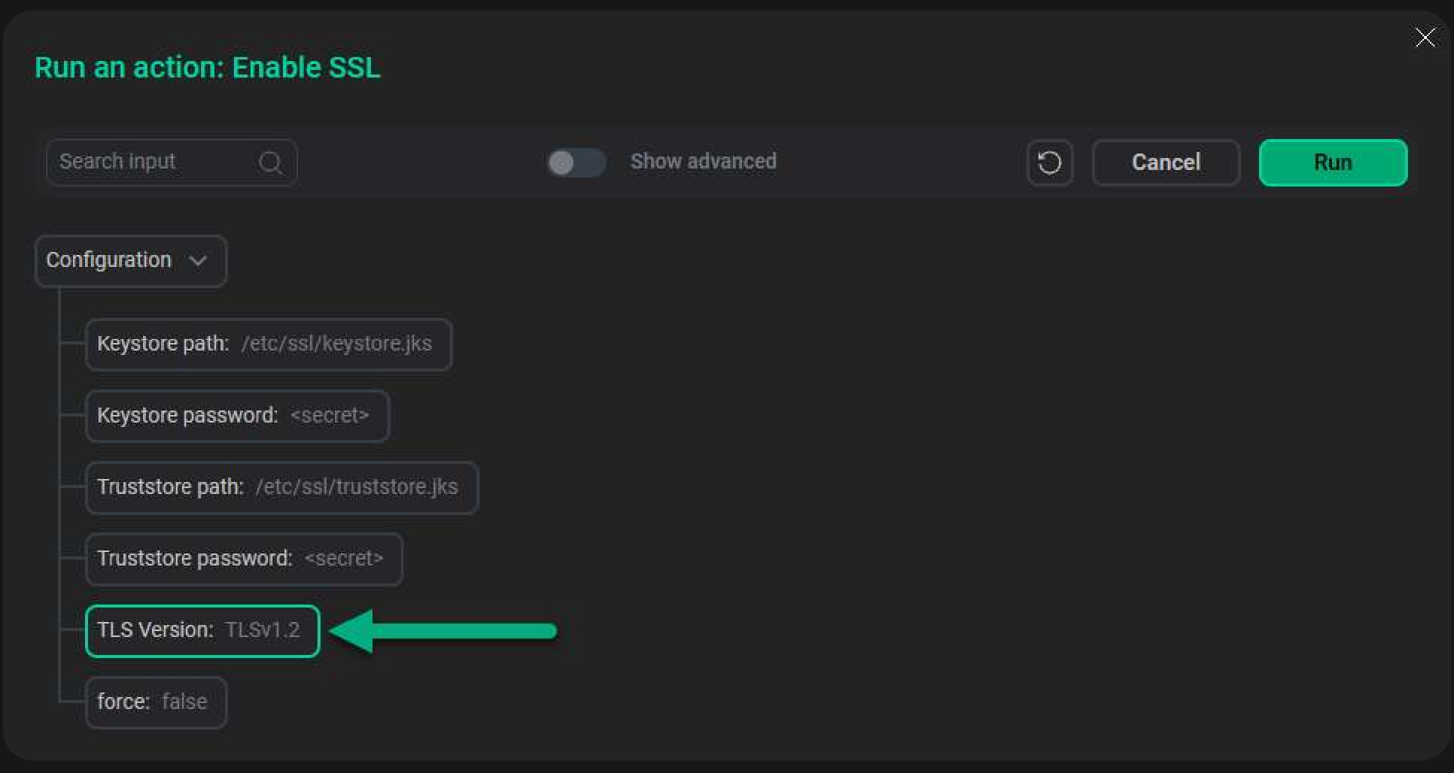
- Выбор версии TLS
- Настройки TLS для отдельных ADH сервисов
- Проверка версии TLS
  - Проверка с помощью curl
  - Проверка с помощью OpenSSL

В данной статье показано, как установить определенную версию TLS для SSL-коммуникаций в кластере ADH, а также приведены инструкции для проверки текущей версии TLS.

На момент релиза ADH 2.1.10.b1 минимально поддерживаемая версия TLS — TLSv1.2, что обусловлено использованием Java 8. TLSv1.3 поддерживается в JDK11+ и сборках JDK8 8u261 или более поздних (JRE 1.8.0_261-b12+).

Выбор версии TLS

Вы можете выбрать желаемую версию TLS для всего ADH кластера с помощью параметра ***TLS Version***, который доступен при выполнении действия Manage SSL. На момент релиза ADH 2.1.10.b1 единственной доступной версией является `TLSv1.2`.



Keystore path: /etc/ssl/keystore.jks
Keystore password: <secret>
Truststore path: /etc/ssl/truststore.jks
Truststore password: <secret>
TLS Version: TLSv1.2
force: false

Выбор версии TLS

Настройки TLS для отдельных ADH сервисов

Также вы можете указать версию TLS для отдельных сервисов ADH. Список конфигурационных файлов и параметров, отвечающих за использование TLS, представлен в таблице ниже.

Настройки TLS для сервисов ADH

Сервис	Файл конфигурации	Параметр	Значение по умолчанию	Описание
Hive	/etc/hive/conf/hive-site.xml	hive.ssl.protocol.blacklist	SSLv2Hello,SSLv2,SSLv3,TLSv1,TLSv1.1	Отключает указанные TLS-протоколы для всех Hive Servers
TezUI	/etc/nginx/nginx.conf	server.ssl_protocols	TLSv1.2	Список TLS-протоколов, разделенных запятой, которые могут использоваться для SSL-соединений
Metastore	/etc/hive/conf/hive-site.xml	metastore.ssl.protocol.blacklist	SSLv2Hello,SSLv2,SSLv3,TLSv1,TLSv1.1	Отключает указанные TLS-протоколы для Hive Metastore
HDFS, YARN	/etc/hadoop/conf/core-site.xml	hadoop.ssl.enabled.protocols	TLSv1.2	Разделенный запятыми список TLS-протоколов, используемый для HDFS DataNode
Spark3	/etc/spark3/conf/spark-defaults.conf	spark.ssl.protocol	TLSv1.2	Указывает TLS-протокол для SSL-соединений
Livy	/etc/livy-spark3/conf/livy.conf	livy.server.thrift.ssl.protocol.blacklist	SSLv2,SSLv3,TLSv1,TLSv1.1	Отключает указанные TLS-протоколы
Flink	/etc/flink/conf/flink-conf.yaml	security.ssl.protocol	TLSv1.2	Указывает определенную версию TLS. Указание нескольких значений через запятую не поддерживается
ZooKeeper 3.5.5+	/etc/zookeeper/conf/zoo.cfg	▪ ssl.protocol ▪ ssl.quorum.protocol	TLSv1.2	`ssl.protocol` устанавливает TLS-протокол для коммуникации между клиентами. `ssl.quorum.protocol` устанавливает TLS-протокол для коммуникации при выборе кворума

Проверка версии TLS

Чтобы определить версию TLS, которая используется в ADH, можно использовать curl или OpenSSL, как показано ниже.

Проверка с помощью curl

В следующем примере curl отправляет HTTPS-запрос к Hive server, используя устаревший TLSv1.1.

```
$ curl https://ka-adh-1.ru-central1.internal:10002/  --insecure --tlsv1.1 --tls-max 1.1
```

Полученный ответ информирует, что протокол TLSv1.1 не поддерживается.

```
curl: (35) Peer reports incompatible or unsupported protocol version
```

Если не указывать `--tlsv1.1 --tls-max 1.1`, будет использован TLSv1.2 (по умолчанию) и curl вернет содержимое web-страницы.

Проверка с помощью OpenSSL

Следующей командой OpenSSL открывает SSL-соединение с сервером Tez и пытается получить SSL-сертификаты, используя устаревший TLSv1.1.

```
$ openssl s_client -connect ka-adh-2.ru-central1.internal:9999 -tls1_1
```

OpenSSL не удается установить безопасное соединение из-за неподдерживаемой версии TLS.

```
no peer certificate available. No client certificate CA names send
```

Если не указывать `-tls1_1`, будет использован TLSv1.2 (по умолчанию) и OpenSSL получит SSL-сертификат.

Настройка имперсонации в ADH

Сергей Тихомиров

Содержание

Назад к содержанию

Обзор
Имперсонация для Livy
  Подготовка кластера в ADCM
  Подготовка на хосте
  Тестирование
Имперсонация для SSM
Имперсонация для Zeppelin

Обзор

В экосистеме Hadoop под имперсонацией понимается механизм, позволяющий сервису производить действия от имени другого пользователя. Имперсонация реализуется для соответствия политикам безопасности (например, Kerberos или Ranger-политика), изоляции данных и ресурсов между пользователями, а также аудита действий с фиксацией в логах имени пользователя, который осуществил запуск действия.

ADH поддерживает имперсонацию для следующих сервисов: HBase (Phoenix, Thrift), Spark (Livy), Hive, Zeppelin, Flink, Solr, Sqoop, Airflow. По умолчанию имперсонация включена для всех сервисов, но некоторым из них требуются дополнительные настройки, описанные ниже. Чтобы настроить сервис для имперсонации определенных пользователей, измените значения параметров ***hadoop.proxyuser.<service>.groups*** и ***hadoop.proxyuser.<service>.hosts*** в группе параметров ***core-site.xml*** сервиса ***Core configuration***.

Имперсонация для Livy

Пример ниже демонстрирует, как настроить имперсонацию для Livy.

Подготовка кластера в ADCM

1. В ADCM выберите ваш кластер ADH и перейдите к настройкам сервиса Spark. Раскройте группу параметров ***livy.conf*** и убедитесь, что параметру ***livy.impersonation.enabled*** присвоено значение `true`.
2. Перейдите к настройкам сервиса ***Core configuration*** и отредактируйте параметры ***hadoop.proxyuser.livy.groups*** и ***hadoop.proxyuser.livy.hosts***. По умолчанию они заполнены значением `*`, а вам нужно предоставить настоящие имена групп и адреса хостов. Сервис сможет имперсонировать пользователей из указанных групп при условии, что запрос приходит с разрешенного хоста.
3. Сохраните конфигурацию и запустите действие ***Update Core configuration***.

Подготовка на хосте

1. Создайте пользовательские директории в HDFS:

```
$ sudo -u hdfs hdfs dfs -mkdir /user/<username>
$ sudo -u hdfs hdfs dfs -chown <username>:<group> /user/<username>
```

2. Выдайте необходимые права пользователю и скопируйте JAR-примеры Spark в HDFS:

```
$ hdfs dfs -chmod -R a+rwx /user/<username>
$ hdfs dfs -chmod -R a+rwx /var/log/spark/apps
$ hdfs dfs -mkdir /user/<username>/spark_examples
$ hdfs dfs -put /usr/lib/spark/examples/jars/* /user/<username>/spark_examples
```

Тестирование

1. Создайте JSON-файл, подобный представленному:

```
{
"file": "/user/<username>/spark_examples/spark-examples_2.11-2.3.2.3.1.0.15-1.jar",
"className": "org.apache.spark.examples.SparkPi",
"name": "test-livy-1",
"args": [10],
"proxyUser": "<username>"
}
```

2. Отправьте запрос с созданным JSON в Livy:

```
$ curl --negotiate -v -u: -d @app_spark.json -H "X-Requested-By: <username>" -H 'Content-Type: application/json' -X POST
'http://kru-adh-22.ru-central1.internal:8998/batches'
```

Имперсонация для SSM

Чтобы настроить имперсонацию для SSM, выполните следующие шаги:

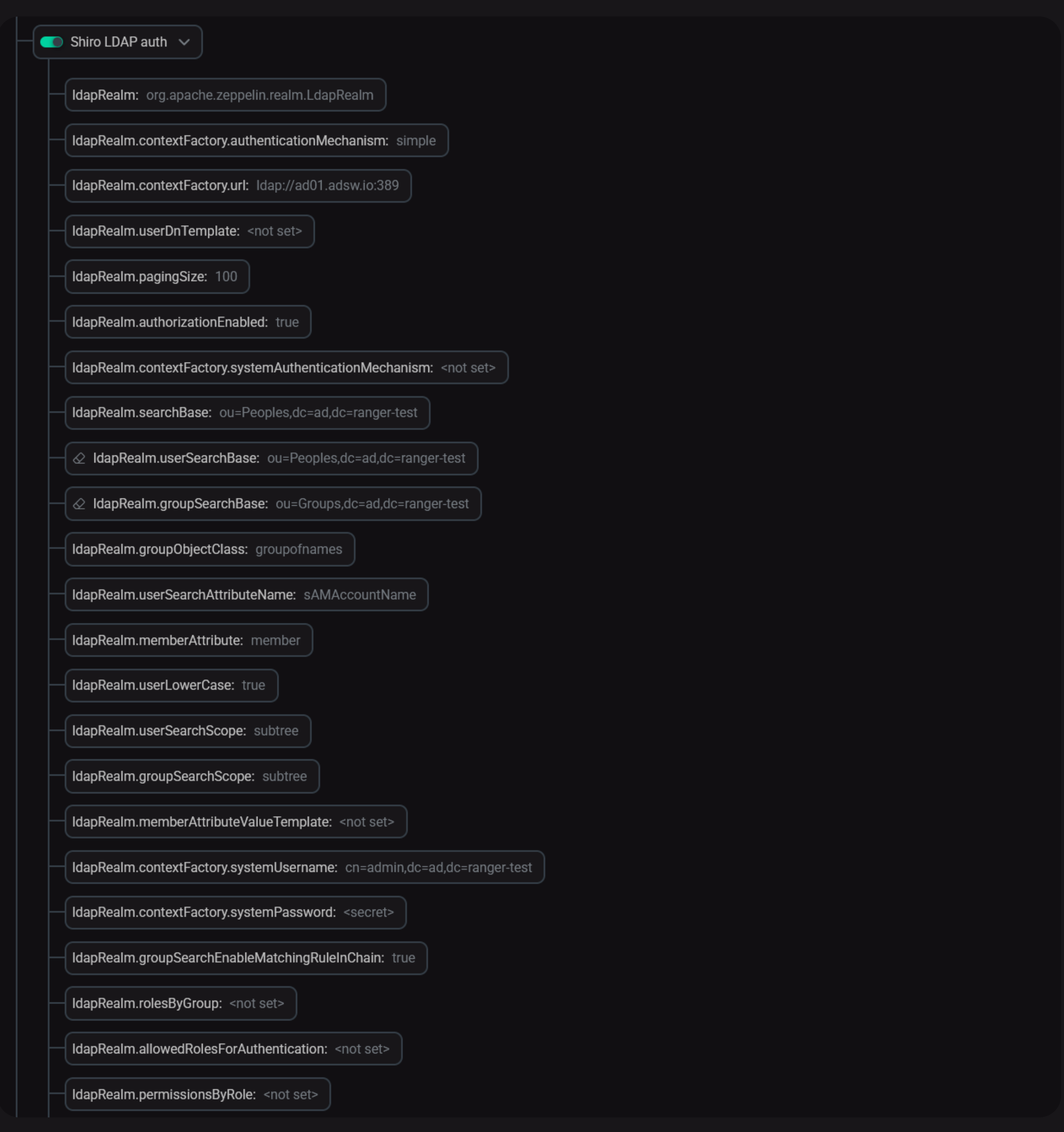
1. Перейдите в веб-интерфейс ADCM и выберите свой кластер на странице ***Clusters***.
2. Откройте вкладку ***Services*** и выберите сервис ***SSM***.
3. Переведите в активное состояние переключатель ***Show advanced*** и раскройте группу параметров ***Custom smart-site.xml***.
4. Нажмите ***Add property*** и задайте следующие значения:
   - Имя параметра — `smart.proxy.user`;
   - Значение параметра — имя пользователя для имперсонации.
5. Нажмите ***Apply***.
6. Раскройте группу параметров ***smart-site.xml*** и задайте значения следующих параметров:
   - ***smart.proxy.user.strategy*** — определяет используемую стратегию имперсонации. Выберите одно из следующих значений:
     - `NODE_SCOPE` — включить имперсонацию на уровне узла. Все действия будут выполняться от имени ранее заданного пользователя.
     - `CMDLET_SCOPE` — включить имперсонацию на уровне cmdlet. Все действия будут выполняться от имени владельца cmdlet (создателя cmdlet).
   - ***smart.proxy.users.cache.ttl*** — интервал времени с момента последнего осуществления доступа к записи пользователя в кеше механизма имперсонации, прежде чем эта запись будет удалена.
   - ***smart.proxy.users.cache.size*** — размер кеша механизма имперсонации.
7. Нажмите ***Save***, затем ***Create***.
8. Нажмите ***Actions*** и выберите ***Restart***.

Имперсонация для Zeppelin

Для настройки имперсонации для Zeppelin требуется включить Active Directory Kerberos как для кластера ADH, так и для связанного кластера ADPS. Также необходимо активировать Ranger-плагины для HDFS, YARN и Hive.

Чтобы увидеть эффект от имперсонации для Zeppelin, следуйте шагам ниже:

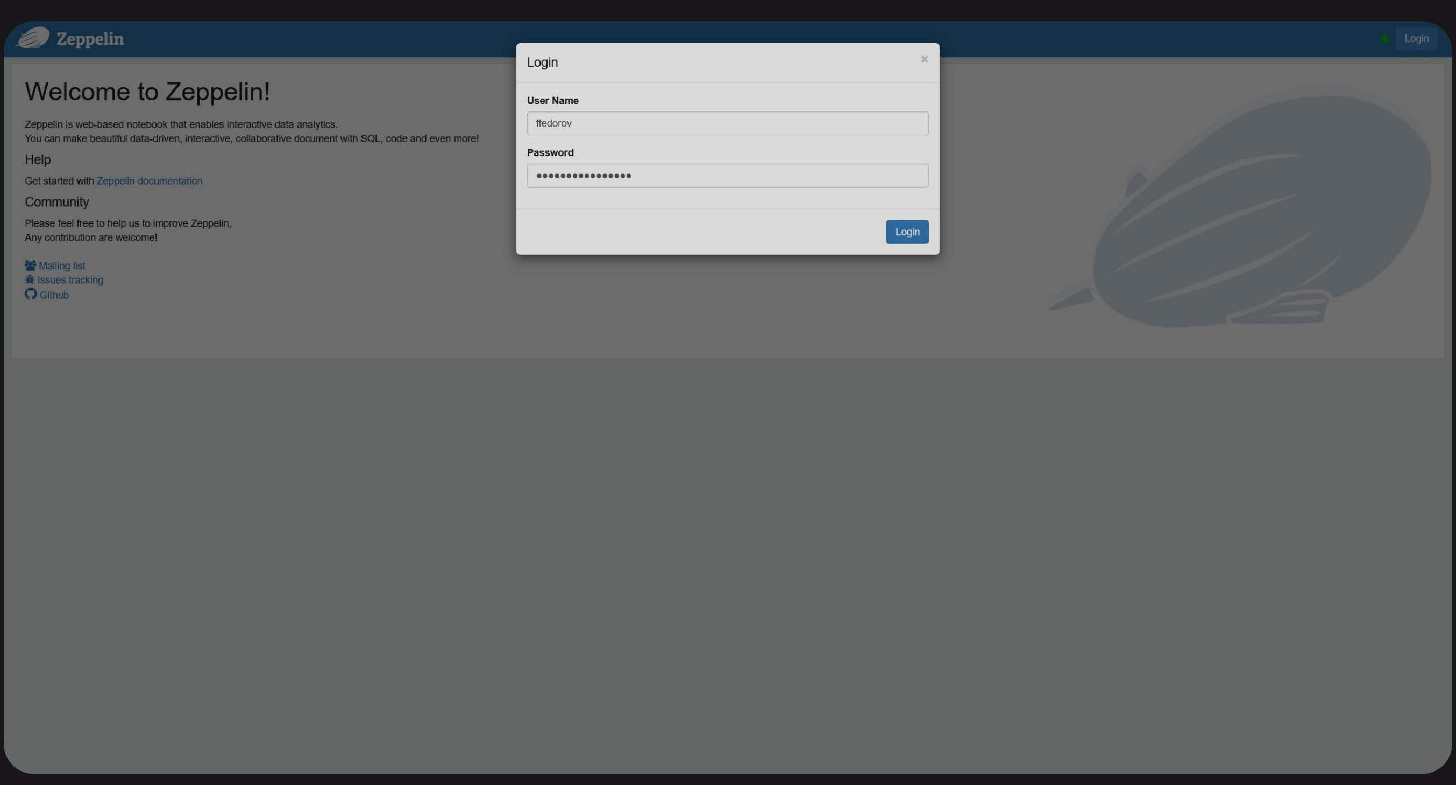
1. В ADCM выберите ваш кластер ADH и перейдите к настройкам сервиса Zeppelin. Активируйте параметр ***Shiro LDAP auth*** и заполните необходимые данные. После этого перезапустите Zeppelin.



Shiro LDAP auth
ldapRealm: org.apache.zeppelin.realm.LdapRealm
ldapRealm.contextFactory.authenticationMechanism: simple
ldapRealm.contextFactory.url: ldap://ad01.adsw.io:389
ldapRealm.userDnTemplate: <not set>
ldapRealm.pagingSize: 100
ldapRealm.authorizationEnabled: true
ldapRealm.contextFactory.systemAuthenticationMechanism: <not set>
ldapRealm.searchBase: ou=Peoples,dc=ad,dc=ranger-test
ldapRealm.userSearchBase: ou=Peoples,dc=ad,dc=ranger-test
ldapRealm.groupSearchBase: ou=Groups,dc=ad,dc=ranger-test
ldapRealm.groupObjectClass: groupofnames
ldapRealm.userSearchAttributeName: sAMAccountName
ldapRealm.memberAttribute: member
ldapRealm.userLowerCase: true
ldapRealm.userSearchScope: subtree
ldapRealm.groupSearchScope: subtree
ldapRealm.memberAttributeValueTemplate: <not set>
ldapRealm.contextFactory.systemUsername: cn=admin,dc=ad,dc=ranger-test
ldapRealm.contextFactory.systemPassword: <secret>
ldapRealm.groupSearchEnableMatchingRuleInChain: true
ldapRealm.rolesByGroup: <not set>
ldapRealm.allowedRolesForAuthentication: <not set>
ldapRealm.permissionsByRole: <not set>

Параметры LDAP-аутентификации в Zeppelin

2. Авторизуйтесь в веб-интерфейсе Zeppelin с данными пользователя Active Directory.



Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Help

Get started with Zeppelin documentation

Community

Please feel free to help us to improve Zeppelin,
Any contribution are welcome!

Mailing list
Issues tracking
Github

Login

User Name: ffedorov

Password: ••••••••••••••••

Авторизация в веб-интерфейсе Zeppelin

3. Выберите интерпретатор, создайте заметку и запустите ее на выбранном интерпретаторе.
4. В веб-интерфейсе Ranger Admin перейдите на страницу ***Audit*** → ***Access***. В логах вы увидите запись с именем пользователя Active Directory.

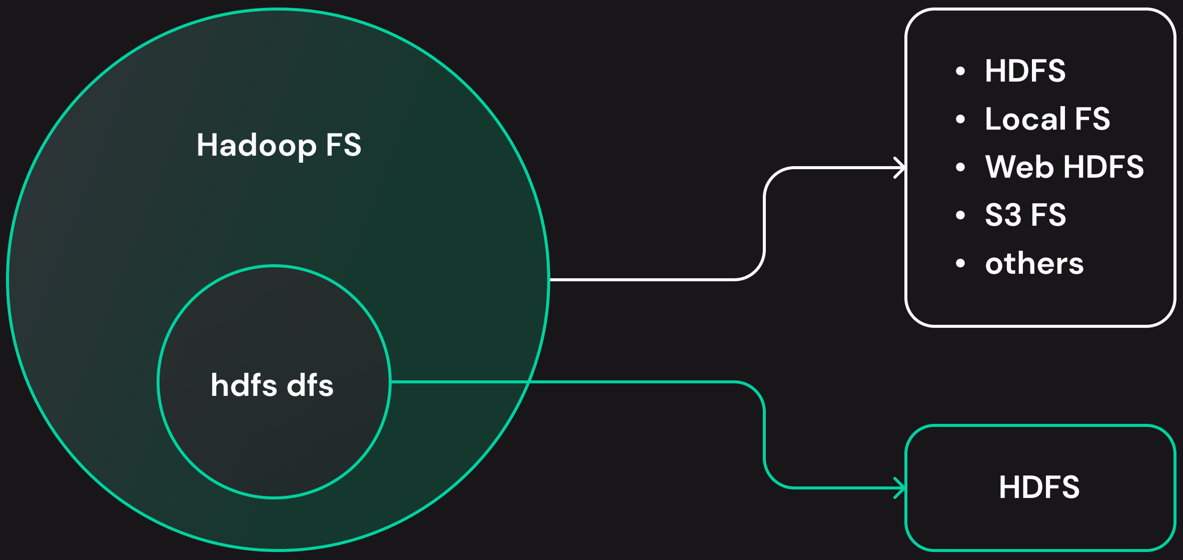
Командная строка Hadoop

Сергей Остапов

Назад к содержанию

Рассмотрим разницу между командами `hadoop fs {args}`, `hadoop dfs {args}` и `hdfs dfs {args}`.

Параметр `fs` относится к общей файловой системе, которая может указывать на любую файловую систему, например, локальную, HDFS и т.д. Таким образом, `fs` можно использовать при работе с различными файловыми системами, такими как Local FS, (S)FTP, S3 и другими.



Уровни Hadoop FS

Когда вы используете параметр `dfs`, вы напрямую указываете взаимодействие с HDFS. Так, если необходимо работать только с HDFS, используйте команду `hdfs dfs`. Если же требуется получить доступ/перенести данные между различными файловыми системами, используйте команду `hadoop fs`.

Чтобы взаимодействовать с оболочкой HDFS, пользователь должен обладать соответствующими правами на чтение/запись к директории(-ям) HDFS. Директории могут быть созданы пользователями, принадлежащими группам `hadoop` или `hdfs`. В демонстрационных целях в примере ниже используется пользователь `hdfs`.

Можно предоставить права `root` текущему пользователю, как показано ниже:

```
$ sudo -s
```

Затем нужно переключиться на пользователя `hdfs`:

```
$ su - hdfs
```

Теперь можно выполнять команды в оболочке HDFS.

Hadoop CLI

Сергей Остапов

Содержание

Назад к содержанию

Обзор

Пользовательские команды

Административные команды

Обзор

Эта статья содержит справочную документацию по оболочке Hadoop shell CLI.

ПРИМЕЧАНИЕ

В настоящее время команда `hadoop dfs` устарела.

Все команды и подпроекты Hadoop имеют одну и ту же базовую структуру:

```
$ shellcommand [SHELL_OPTIONS] [COMMAND] [GENERIC_OPTIONS] [COMMAND_OPTIONS]
```

Базовая структура оболочки Hadoop

shellcommand	Команда вызываемого проекта. Например, в Hadoop обычно используется `hadoop`, в HDFS — `hdfs`, а в YARN — `yarn`
SHELL_OPTIONS	Параметры, которые оболочка (shell) обрабатывает перед выполнением Java
COMMAND	Действия для выполнения
GENERIC_OPTIONS	Общий набор опций, поддерживаемых несколькими командами
COMMAND_OPTIONS	Различные варианты опций для подпроектов Hadoop

Все команды командного интерпретатора принимают общий набор параметров. Для некоторых команд эти параметры игнорируются. Например, передача параметра `--hostnames` команде, которая выполняется только на одном хосте, будет проигнорирована.

Shell options

--buildpaths	Позволяет использовать версии JAR для разработчиков
--config confdir	Перезаписывает каталог конфигурации по умолчанию. По умолчанию используется *$HADOOP_HOME/etc/hadoop*
--daemon mode	Если команда поддерживает демонизацию (например, `hdfs namenode`), то она выполняется в соответствующем режиме. Поддерживаются следующие режимы: `start` для запуска процесса в режиме демона, `stop` для остановки процесса и `status` для определения активного состояния процесса. Статус возвращает код результата, см. LSB-compliant. Если опция не указана, команды, поддерживающие демонизацию, будут выполняться в фоновом режиме. Для команд, которые не поддерживают демонизацию, этот параметр игнорируется
--debug	Включает отладочную информацию о конфигурации на уровне оболочки
--help	Выводит справочную информацию об использовании всех команд
--hostnames	Если используется аргумент `--workers`, то происходит переопределение файла *workers* со списком имен хостов, разделенных пробелами, для выполнения подкоманды на нескольких хостах. Если аргумент `--workers` не используется, то параметр `--hostnames` игнорируется
--hosts	Когда используется аргумент `--workers`, происходит подмена файла *workers* другим файлом, содержащим список имен хостов для выполнения мультихостовой подкоманды. Если аргумент `--workers` не используется, то параметр `-hosts` игнорируется
--loglevel loglevel	Отменяет уровень логов. Доступные уровни логов: `FATAL`, `ERROR`, `WARN`, `INFO`, `DEBUG` и `TRACE`. По умолчанию — `INFO`
--workers	Используется на всех хостах, где есть файл *workers*

Многие подкоманды используют общий набор параметров конфигурации для изменения своего поведения.

Generic options

-archives <comma separated list of archives>	Указывает список архивов, разделенных запятыми, для разархивирования на вычислительных (compute) машинах. Применяется только к задаче (job)
-conf <configuration file>	Задает файл конфигурации приложения
-D <property>=<value>	Использует значение для данного свойства
-files <comma separated list of files>	Указывает разделенные запятыми файлы для копирования в кластер MapReduce. Применяется только к задаче (job)
-fs <file:///> or <hdfs://namenode:port>	Указывает URL-адрес файловой системы по умолчанию. Переопределяет параметр `fs.defaultFS`
-jt <local> or <resourcemanager:port>	Указывает ResourceManager. Применяется только к задаче (job)
-libjars <comma separated list of jars>	Указывает разделенные запятыми JAR-файлы для включения в classpath. Применяется только к задаче (job)

Пользовательские команды

Эти команды используются пользователями кластера Hyperwave.

Команда	Описание
archive	Создает архив Hadoop
checknative	Проверяет доступность нативного кода Hadoop
CLASSNAME	Запускает Java-класс
classpath	Выводит classpath
credential	Управляет учетными данными, паролями и секретами
distch	Изменяет права собственности и разрешения на файлы
distcp	Копирует файл или каталоги рекурсивно
dtutil	Утилита для получения и управления токенами Hadoop
envvars	Отображает вычисляемые переменные окружения Hadoop
fs	Это синоним `hdfs dfs`, когда используется HDFS
gridmix	Инструмент бенчмарка для кластера Hyperwave
jar	Запускает JAR-файл
jnipath	Выводит вычисленное значение java.library.path
kerbname	Преобразует именованный принципал с помощью правил auth_to_local в имя пользователя Hadoop
kdiag	Диагностика проблем Kerberos
key	Управление ключами через KeyProvider
kms	Запускает сервер управления ключами (KMS)
trace	Просмотр и изменение настроек трассировки Hadoop
version	Выводит текущую версию Hadoop

Административные команды

Команды используются администраторами кластера Hyperwave.

Команда	Описание
daemonlog	Получает/устанавливает уровень логов

archive

Сергей Остапов

Назад к содержанию

Создает Hadoop-архив. Дополнительная информация о работе с архивами доступна на странице Hadoop Archives Guide.

Использование показано ниже:

```
$ hadoop archive -archiveName name -p <parent> [-r <replication factor>] <src>* <dest>
```

Аргументы

-archiveName	Имя создаваемого *.har* архива
-p	Относительный путь для архивации файлов. Например: `-p /foo/bar a/b/c e/f/g` где `/foo/bar` — путь к родительской директории, а `a/b/c` и `e/f/g` — относительные по отношению к родительской директории
-r	Устанавливает желаемый фактор репликации. Если этот необязательный параметр не указан, используется фактор репликации, равный .

Пример:

```
$ hadoop archive -archiveName zoo.har -p /foo/bar -r 3 /outputdir
```

checknative

Сергей Остапов

Назад к содержанию

Проверяет доступность нативного кода Hadoop. По умолчанию эта команда проверяет доступность только *libhadoop*. Подробную информацию о нативном коде Hadoop можно получить в статье Native Libraries Guide.

Использование показано ниже:

```
$ hadoop checknative [-a] [-h]
```

Аргументы

-a	Проверяет доступность всех библиотек
-h	Отображает справочную информацию

CLASSNAME

Сергей Остапов

Назад к содержанию

Запускает класс с именем `ClassName`. Класс должен принадлежать пакету.

Использование показано ниже:

```
$ hadoop classname <ClassName>
```

classpath

Сергей Остапов

Назад к содержанию

Выводит classpath, необходимый для получения файла Hadoop JAR и требуемых библиотек.
Если вызывается без аргументов, то выводит classpath, установленный командными скриптами, который, скорее всего, будет содержать подстановочные знаки в записях пути классов.

Дополнительные опции выводят classpath после расширения wildcard или записывают classpath в манифест JAR-файла.
Последний вариант полезен в средах, где подстановочные знаки не могут быть использованы, а расширенный classpath превышает максимальную поддерживаемую длину командной строки.

Использование показано ниже:

```
$ hadoop fs classpath [--glob |--jar <path> |-h |--help]
```

Аргументы

--glob	Разворачивает подстановочные знаки
--jar path	Записывает classpath как манифест в JAR с именем `path`
-h, --help	Выводит справочную информацию

conftest

Сергей Останов

Назад к содержанию

Выполняет валидацию XML-файлов конфигурации. Если параметр `-conffile` не указан, все *.xml*-файлы в каталоге *${HADOOP_CONF_DIR}* будут проверены. Если параметр задан, выполняется проверка только указанных файлов.

Можно указать либо файл, либо каталог. Если указан каталог, будут провалидированы файлы каталоге, имена которых заканчиваются на *.xml*. Параметр `-conffile` можно указать несколько раз.

Валидация довольно минимальна: XML анализируется и дублируется, проверяются пустые имена свойств. Команда не поддерживает XInclude. Если вы используете XInclude для импорта элементов конфигурации, такой XML-файл будет объявлен невалидным.

Пример использования показан ниже:

```
$ hadoop conftest [-conffile <path>]...
```

Аргументы

-conffile	Путь к файлу конфигурации, который необходимо провалидировать
-h, --help	Выводит справочную информацию

Пример:

```
$ hadoop conftest
$ hadoop conftest -conffile /etc/hadoop/conf/core-site.xml
```

credential

Сергей Остапов

Назад к содержанию

Управляет учетными данными, паролями и секретными ключами в поставщиках учетных данных (credential providers).

CredentialProvider API в Arenadata Hyperwave позволяет разделить приложения и то, как они хранят свои пароли/секреты.
Чтобы указать конкретный тип и местоположение провайдера, необходимо указать параметр конфигурации `hadoop.security.credential.provider.path` в *core-site.xml* или использовать флаг `-provider` для каждой из следующих команд.

Путь к провайдеру представляет собой список URL-адресов, разделенных запятыми, который указывает тип и расположение списка провайдеров, к которым следует обратиться.
Например, путь *user:///jceks://file/tmp/test.jceks,jceks://hdfs@nn1.example.com/my/path/test.jceks* указывает, что:

- к файлу учетных данных текущего пользователя следует обращаться через User Provider;
- локальный файл */tmp/test.jceks* является Java Keystore Provider;
- HDFS-файл *nn1.example.com/my/path/test.jceks* также является хранилищем для Java Keystore Provider.

Команда `credential` часто используется для предоставления пароля или секретного ключа определенному поставщику хранилища учетных данных.
Чтобы явно указать, какое хранилище провайдеров использовать, используйте флаг `-provider`.
Иначе при указании нескольких путей к провайдерам будет использован первый непереходный провайдер.
Это может быть как нужный провайдер, так и любой другой.

Провайдеры часто требуют ввода пароля или другого секретного ключа.
Если провайдер требует пароль и не может его найти, он использует пароль по умолчанию и выдает предупреждение о том, что используется пароль по умолчанию.
Если установлен флаг `-strict`, предупреждение становится сообщением об ошибке, и команда немедленно возвращается со статусом ошибки.

Использование показано ниже:

```
$ hadoop credential <subcommand> [options]
```

Аргументы

print *[-alias alias]* *filename [filename2 ...]*	Выводит поля в токенах, содержащихся в *filename* (и *filename2* ...). Если указан `alias`, выводит только токены, соответствующие `alias`. Иначе, выводит все токены
get URL *[-service scheme]* *[-format (java|protobuf)]* *[-alias alias]* *[-renewer renewer]* *filename | filename [filename2 ...]*	Получает токен от сервиса по URL-адресу и помещает его в файл. URL является обязательным и должен следовать сразу за запросом `get`. `URL` — это URL сервиса, например, *hdfs://localhost:9000*. `alias` будет перезаписывать поле сервиса в токене. Предназначен для хостов, имеющих внешние и внутренние имена, например, *firewall.com:14000*. `filename` должен указываться последним и является именем файла токена. Будет создан, если не существует. В противном случае токен(ы) добавляются в существующий файл. Флаг `-service` следует использовать только с URL, который начинается с http или https. Следующие варианты эквивалентны: *hdfs://localhost:9000/* и *http://localhost:9000 -service hdfs*
append *[-format (java|protobuf)]* *filename filename2 [filename3 ...]*	Добавляет содержимое первых `N` файлов к последнему файлу. Маркеры, присутствующие в последнем файле, всегда сохраняются
remove -alias alias *[-format (java|protobuf)]* *filename [filename2 ...]*	Из каждого указанного файла удаляет токены, соответствующие `alias`, и каждый файл записывается в указанном формате. Аргумент `alias` должен быть указан
cancel -alias alias *[-format (java|protobuf)]* *filename [filename2 ...]*	Аналогично `remove`, за исключением того, что токены также отменяются с помощью сервиса, указанного в объекте токена. Аргумент `alias` должен быть указан
renew -alias alias *[-format (java|protobuf)]* *filename [filename2 ...]*	Для каждого указанного файла обновляет токены, соответствующие `alias`, и выписывает каждый файл в указанном формате. Аргумент должен быть указан
import base64 *[-alias alias]* *filename*	Импортирует Base64-токен. Аргумент `alias` перезапишет служебное поле в токене

Пример:

```
$ hadoop credential list -provider jceks://file/tmp/test.jceks
```

distch

Сергей Остапов

Назад к содержанию

Изменяет права собственности и разрешения для нескольких файлов одновременно.

Использование показано ниже:

```
$ hadoop distch [-f urilist_url] [-i] [-log logdir] path:owner:group:permissions
```

Аргументы

-f	Использует список, указанный в `<urilist_uri>` в качестве списка исходных данных
-i	Игнорирует ошибки
-log	Указывает каталог для вывода логов

distcp

Сергей Остапов

Назад к содержанию

Используется для копирования данных внутри и между кластерами. Команда использует MapReduce для распространения, обработки и восстановления ошибок, а также для создания отчетов. Передает список файлов и директорий в качестве входных данных для map-задач, каждая из которых копирует файлы, указанные в списке исходных данных. Дополнительную информацию можно найти на странице Hadoop DistCp Guide.

Использование показано ниже:

```
$ hadoop distcp [OPTIONS] <src> <dst>
```

Аргументы

-append	Использует существующие данные в конечных файлах и добавляет к содержимому новые данные, если это возможно
-async	Определяет, должно ли выполнение `distcp` быть блокирующим
-atomic	Выполняет все изменения атомарно
-bandwidth <arg>	Указывает пропускную способность для map-задачи в МБ, возможно указание дробного значения
-blocksperchunk <arg>	При указании положительного значения те файлы, которые занимают больше блоков, чем указано в значении, будут разделены на части (chunks) <blocksperchunk> и будут перемещены параллельно. При достижении конечной файловой системы, части файлов будут собраны воедино. По умолчанию <blocksperchunk> равен 0, а значит файлы передается цельно, без разделения на части. Это параметр работает, если исходная файловая система реализует метод `getBlockLocations`, а конечная файловая система реализует метод `concat`
-copybuffersize <arg>	Размер буфера копирования (в байтах). По умолчанию равен 8192
-delete	Удаляет файлы с конечной файловой системы, если этих файлов нет в source
-diff <arg>	Использует diff-отчет снепшотов для определения разницы между source и target
-f <arg>	Указывает список файлов для копирования
-filters <arg>	Путь к файлу, который содержит список путей, которые необходимо исключить из операции копирования
-i	Игнорирует ошибки при копировании
-log <arg>	Указывает DFS-директорию для сохранения логов `distcp`
-m <arg>	Максимальное количество параллельных map-задач для копирования
-numListstatusThreads <arg>	Количество потоков для построения листинга файлов (40 максимум)
-overwrite	Перезаписывает файлы в target-директории, даже если файлы уже существуют
-p <arg>	Сохраняет статус (rbugpcaxt)(репликация, размер блока, пользователь, группа, разрешения, тип контрольной суммы, списки управления доступом, XATTR, временные метки). Если `-p` задан без аргументов, сохраняет репликацию, размер блока, пользователя, группу, разрешения, тип контрольной суммы и отметки времени
-rdiff <arg>	Использует diff-отчет снепшотов для определения изменений на target-системе
-skipcrccheck	Определяет, пропускать ли CRC-проверки между source/target-путями
-strategy <arg>	Определяет стратегию копирования. По умолчанию используется подход с разделением работы на основе размера файлов
-tmp <arg>	Путь к временной директории, используемой для осуществления атомарных операций копирования
-update	Копирует в target недостающие файлы и переписывает существующие, если есть совпадения
-v	Сохраняет дополнительную информацию (путь, размер) в логи SKIP/COPY
-xtrack <arg>	Сохраняет информацию о недостающих source-файлах в указанную директорию

Пример:

```
$ hadoop distcp hdfs://nn1:8020/foo/bar hdfs://nn2:8020/bar/foo
$ hadoop distcp hdfs://nn1:8020/foo/a hdfs://nn1:8020/foo/b hdfs://nn2:8020/bar/foo
```

dtutil

Сергей Остапов

Назад к содержанию

Утилита для управления токенами делегации Hadoop внутри файлов учетных данных. Утилита предназначена для замены более простой команды fetchdt. Существует несколько подкоманд, каждая со своими флагами и аргументами.

Для каждой подкоманды аргумент `-format` указывает внутренний формат для использования. `java` — это унаследованный формат, который соответствует `fetchdt`. По умолчанию используется `protobuf`.

Для каждой подкоманды, которая подключается к сервису, предоставляются флаги для указания принципала Kerberos и keytab-файла, который будет использоваться для аутентификации.

Использование показано ниже:

```
$ hadoop dtutil [-keytab keytab_file -principal principal_name ] subcommand [-format (java|protobuf)] [-alias alias ] [-renewer renewer ] filename…
```

Аргументы

print [-alias alias] filename [filename2 ...]	Выводит поля в токенах, содержащихся в *filename* (и *filename2* ...). Если указан `alias`, выводит только токены, соответствующие псевдониму. В противном случае выводит все токены
get URL [-service scheme] [-format (java|protobuf)] [-alias alias] [-renewer renewer] filename | filename [filename2 ...]	Получает токен от сервиса по URL и помещает его в файл. `URL` является обязательным и должен следовать сразу за запросом `get`. `URL` — это URL службы, например, *hdfs://localhost:9000*. Аргумент `alias` будет перезаписывать поле сервиса в токене. Предназначен для хостов, имеющих внешние и внутренние имена, например, *firewall.com:14000*. Аргумент `filename` должен указываться последним и является именем файла токена. Будет создан, если не существует. Иначе токен(ы) добавляются в существующий файл. Флаг `-service` следует использовать только с URL, который начинается с "http" или "https". Следующие варианты эквивалентны: *hdfs://localhost:9000/* и *http://localhost:9000 -service hdfs*
append [-format (java|protobuf)] filename filename2 [filename3 ...]	Добавляет содержимое первых `N` файлов к последнему файлу. Когда токены с общими служебными полями присутствуют в нескольких файлах, токены предыдущих файлов перезаписываются, то есть токены, присутствующие в последнем файле, всегда сохраняются
remove -alias alias [-format (java|protobuf)] filename [filename2 ...]	Из каждого указанного файла удаляет токены, соответствующие `alias`, и записывает каждый файл в указанном формате. Аргумент `alias` обязателен
cancel -alias alias [-format (java|protobuf)] filename [filename2 ...]	Аналогично `remove`, за исключением того, что токены также отменяются с помощью сервиса, указанного в объекте токена. Аргумент `alias` обязателен
renew -alias alias [-format (java|protobuf)] filename [filename2 ...]	Для каждого указанного файла обновляет токены, соответствующие псевдониму, и выписывает каждый файл в указанном формате. Аргумент `alias` обязателен
import base64 [-alias alias] filename	Импортирует токен из base64-токена. Аргумент `alias` перезапишет служебное поле в токене

envvars

Сергей Остапов

Назад к содержанию

Отображает вычисленные переменные окружения Hadoop.

Использование показано ниже:

```
$ hadoop envvars
```

fs

Сергей Остапов

Назад к содержанию

Эта команда позволяет напрямую управлять распределенной файловой системой Hadoop (HDFS), а также работать с другими файловыми системами, которые поддерживает Hadoop, такими как Local FS, WebHDFS, S3 FS и другими.

Использование:

```
$ hadoop fs <args>
```

Описания аргументов, которые поддерживаются командой `fs`, доступны в разделе FileSystem shell.

gridmix

Сергей Остапов

Назад к содержанию

Gridmix — это инструмент бенчмарка для кластера Hyperwave. Более подробную информацию об этом инструменте можно получить в статье Gridmix Guide.

Использование без параметров конфигурации показано ниже:

```
$ hadoop gridmix [-generate <size>] [-users <users-list>] <iopath> <trace>
```

Использование с параметрами конфигурации показано ниже:

```
$ hadoop gridmix \
  -Dgridmix.client.submit.threads=10 -Dgridmix.output.directory=foo \
  [-generate <size>] [-users <users-list>] <iopath> <trace>
```

jar

Сергей Остапов

Назад к содержанию

Запускает JAR-файл. Вместо этой команды можно использовать команду YARN jar для запуска JAR-файлов.

Использование показано ниже:

```
$ hadoop jar <jar> [mainClass] args...
```

jnipath

Сергей Остапов

Prints the computed java.library.path.

The usage is as follows:

```
$ hadoop jnipath
```

kerbname

Сергей Остапов

Назад к содержанию

Преобразует именованные принципалы с помощью правил auth_to_local в имя пользователя Arenadata Hyperwave.

Использование показано ниже:

```
$ hadoop kerbname
```

Пример:

```
$ hadoop kerbname user@EXAMPLE.COM
```

kdiag

Сергей Остапов

Назад к содержанию

Запускает диагностику проблем Kerberos.

Использование показано ниже:

```
$ hadoop kdiag
```

key

Сергей Остапов

Назад к содержанию

Данная команда позволяет управлять ключами с помощью KeyProvider. Больше информации о KeyProvider можно получить в статье Transparent Encryption in HDFS.

Провайдеры часто требуют ввода пароля или другого секретного ключа. Если провайдер требует пароль и не может его найти, он использует пароль по умолчанию и выдает предупреждение об использовании такого пароля. Если установлен флаг `strict`, предупреждение становится сообщением об ошибке, и команда немедленно возвращается со статусом ошибки.

ПРИМЕЧАНИЕ

- Некоторые KeyProviders (например, `org.apache.hadoop.crypto.key.JavaKeyStoreProvider`) не поддерживают имена ключей в верхнем регистре.
- Некоторые KeyProviders не выполняют удаление ключа напрямую (например, вместо этого выполняют мягкое удаление или откладывают удаление, чтобы предотвратить ошибку). В таких случаях могут возникать ошибки при создании/удалении ключа с тем же именем после его удаления. Для получения подробной информации обратитесь к вашему KeyProvider.

Использование показано ниже:

```
$ hadoop key <subcommand> [options]
```

Аргументы

create <keyname> [-cipher cipher] [-size size] [-description description] [-attr attribute=value] [-provider provider] [-strict] [-help]	Создает новый ключ для имени, указанного в аргументе `<keyname>`, в провайдере, указанном в аргументе `provider`. Флаг `strict` приведет к неудаче команды, если провайдер использует пароль по умолчанию. Допускается указать шифр с помощью аргумента `cipher`. В настоящее время по умолчанию используется шифр `AES/CTR/NoPadding`. По умолчанию размер ключа составляет `128`. Вы можете указать требуемую длину ключа с помощью аргумента `size`. С помощью аргумента `attr` можно указать произвольные атрибуты в стиле `attribute=value`. Аргумент `attr` может быть указан несколько раз, по одному разу для каждого атрибута
roll <keyname> [-provider provider] [-strict] [-help]	Создает новую версию для указанного ключа в рамках провайдера, указанного с помощью аргумента `provider`. Флаг `strict` приведет к ошибке, если провайдер использует пароль по умолчанию
delete <keyname> [-provider provider] [-strict] [-f] [-help]	Удаляет все версии ключа, указанного аргументом `<keyname>`, в пределах провайдера, указанного `provider`. Флаг `strict` приведет к ошибке, если провайдер использует пароль по умолчанию. Команда запрашивает подтверждение пользователя, если не указан аргумент `f`
list [-provider provider] [-strict] [-metadata] [-help]	Отображает имена ключей, содержащиеся в определенном провайдере, настроенном в *core-site.xml* или указанном с помощью аргумента `provider`. Флаг `strict` приведет к ошибке, если провайдер использует пароль по умолчанию. Аргумент `metadata` отображает метаданные
check <keyname> [-provider provider] [-strict] [-help]	Проверяет пароль имени ключа, содержащегося в определенном провайдере, настроенном в *core-site.xml* или указанном с помощью аргумента `provider`. Флаг `strict` приведет к отказу команды, если провайдер использует пароль по умолчанию
-help	Выводит помощь об использовании этой команды

kms

Сергей Остапов

Назад к содержанию

Позволяет управлять сервером KMS (Key Management Server).

Пример использования ниже:

```
$ hadoop kms [command]
```

Пример:

```
$ hadoop --daemon start kms
```

trace

Сергей Остапов

Назад к содержанию

Позволяет просмотреть и изменить настройки трассировки Hadoop.
Больше информации о настройке трассировки можно получить в статье Enabling Dapper-like Tracing in Hadoop.

Использование показано ниже:

```
$ hadoop trace
```

version

Сергей Останов

Назад к содержанию

Отображает текущую версию Hadoop.

Использование показано ниже:

```
$ hadoop fs -put [-f] [-p] [-l] [-d] [-t <thread count>] [ - | <localsrc1> .. ]. <dst>
```

daemonlog

Сергей Остапов

Назад к содержанию

Динамически возвращает/устанавливает уровень логирования для лога, идентифицированного квалифицированным именем класса в daemon. По умолчанию команда отправляет HTTP-запрос, но это можно отменить, используя аргумент `-protocol https` для отправки HTTPS-запроса.

Обратите внимание, что эта настройка не является перманентной и будет сброшена при перезапуске daemon. Эта команда работает путем отправки HTTP/HTTPS запроса на внутренний сервлет Jetty daemon, поэтому она поддерживает следующие daemon:

- Common
  - key management server
- HDFS
  - name node
  - secondary name node
  - data node
  - journal node
  - HttpFS server
- YARN
  - resource manager
  - node manager
  - Timeline server

Использование показано ниже:

```
$ hadoop daemonlog -getlevel <host:port> <classname> [-protocol (http|https)]
$ hadoop daemonlog -setlevel <host:port> <classname> <level> [-protocol (http|https)]
```

Аргументы

-getlevel host:port classname [-protocol (httphttps)]	Выводит уровень лога, идентифицированного квалифицированным именем класса, в daemon, запущенном по адресу `host:port`. Флаг `-protocol` указывает протокол для соединения
-setlevel host:port classname level [-protocol (httphttps)]	Устанавливает уровень лога, идентифицированного квалифицированным именем класса, в daemon, запущенном по адресу `host:port`. Флаг `-protocol` указывает протокол для соединения

Пример:

```
$ bin/hadoop daemonlog -setlevel 127.0.0.1:9870 org.apache.hadoop.hdfs.server.namenode.NameNode DEBUG
$ bin/hadoop daemonlog -getlevel 127.0.0.1:9871 org.apache.hadoop.hdfs.server.namenode.NameNode DEBUG -protocol https
```

HDFS CLI

Сергей Остапов

Содержание

Назад к содержанию

Обзор
Пользовательские команды
Административные команды
Команды отладки

Обзор

Этот раздел включает справочную документацию по командной строке HDFS.

Использование команд `hdfs` без аргументов выводит описание всех этих команд.

Использование показано ниже:

```
$ hdfs [SHELL_OPTIONS] COMMAND [GENERIC_OPTIONS] [COMMAND_OPTIONS]
```

В Hadoop есть механизм парсинга опций, который позволяет парсить как общие опции, так и запущенные классы.

Базовая структура командной строки Hadoop

SHELL_OPTIONS	Общий набор опций командной строки
GENERIC_OPTIONS	Общий набор опций, поддерживаемых несколькими командами
COMMAND_OPTIONS	Различные команды с их опциями описаны в следующих разделах

Пользовательские команды

Эти команды полезны для пользователей кластера Hyperwave.

Команда	Описание
classpath	Выводит путь к классу, необходимый для получения JAR Hadoop и требуемых библиотек
dfs	Выполняет команду файловой системы на файловой системе, поддерживаемой в Hadoop
envvars	Отображает вычисленные переменные окружения Hadoop
fetchdt	Получает Delegation Token от NameNode
fsck	Запускает утилиту проверки файловой системы HDFS
getconf	Получает информацию о конфигурации из каталога конфигурации после обработки
groups	Возвращает информацию о группе, заданную одним или несколькими именами пользователей
httpfs	Запускает HttpFS server, HDFS HTTP Gateway
lsSnapshottableDir	Получает список snapshottable-каталогов
jmxget	Выгружает информацию JMX из службы
oev	Hadoop offline edits viewer
oiv	Hadoop Offline Image Viewer для файлов образов
oiv_legacy	Hadoop Offline Image Viewer для устаревших версий Hadoop
snapshotDiff	Определяет разницу между моментальными снимками HDFS
version	Отображает текущую версию HDFS

Административные команды

Эти команды полезны для администраторов кластера Hyperwave.

Команда	Описание
balancer	Запускает утилиту балансировки кластера
cacheadmin	Взаимодействует с пулами кеша и директивами через подкоманду `hdfs cacheadmin`
crypto	Создает новую зону шифрования
datanode	Запускает HDFS DataNode
dfsadmin	Запускает клиент HDFS dfsadmin
dfsrouter	Запускает маршрутизатор DFS
dfsrouteradmin	Управление федерацией на основе маршрутизаторов
diskbalancer	Запускает командную строку балансировки диска
ec	Запускает командную строку ErasureCoding
haadmin	Получает доступ к некоторым дополнительным командам для администрирования кластера HA HDFS
journalnode	Запуск JournalNode для использования с HDFS HA с QJM
mover	Запускает утилиту миграции данных
namenode	Запускает NameNode
nfs3	Запускает шлюз NFS3 для использования со службой HDFS NFS3 Service
portmap	Запускает карту портов RPC для использования со службой HDFS NFS3 Service
secondarynamenode	Запускает HDFS secondary NameNode
storagepolicies	Выводит список всех/получает/устанавливает/отключает политики хранения
zkfc	Запускает процесс ZooKeeper Failover Controller для использования с HDFS HA с QJM

Команды отладки

Эти команды используются для помощи администраторам в отладке проблем с HDFS.

Команда	Описание
computeMeta	Вычисляет метаданные HDFS из блочных файлов
recoverLease	Восстанавливает аренду по указанному пути
verifyMeta	Проверяет метаданные HDFS и файлы блоков

classpath

Сергей Остапов

Назад к содержанию

Выводит classpath, необходимый для получения JAR Hadoop и требуемых библиотек. Если вызывается без аргументов, то выводит путь классов, установленный командными сценариями, который, скорее всего, будет содержать подстановочные знаки в записях пути классов.

Дополнительные опции выводят classpath после расширения wildcard или записывают classpath в манифест JAR-файла. Последний вариант полезен в средах, где подстановочные знаки не могут быть использованы, а расширенный classpath превышает максимальную поддерживаемую длину командной строки.

Использование показано ниже:

```
$ hdfs classpath [--glob |--jar <path> |-h |--help]
```

Аргументы

--glob	Раскрывает wildcards
--jar path	Записывает classpath в манифест JAR-файлов с названием `<path>`
-h, --help	Выводит справочную информацию

dfs

Сергей Остапов

Назад к содержанию

Выполняет команду файловой системы в файловой системе, поддерживаемой в Arenadata Hyperwave HDFS.

Использование показано ниже:

```
$ hdfs dfs [COMMAND [COMMAND_OPTIONS]]
```

Пример:

```
$ hdfs dfs -cat file:///file3 /user/hadoop/file4
$ hdfs dfs -count -q hdfs://nn1.example.com/file1
$ hdfs dfs -cp /user/hadoop/file1 /user/hadoop/file2
```

envvars

Сергей Остапов

Назад к содержанию

Отображает вычисленные переменные окружения Hadoop.

Использование показано ниже:

```
$ hdfs envvars
```

Пример:

```
$ hdfs envvars
JAVA_HOME='/usr/lib/jvm/jre-openjdk'
HADOOP_HDFS_HOME='/usr/lib/hadoop-hdfs'
HDFS_DIR='./'
HDFS_LIB_JARS_DIR='lib'
HADOOP_CONF_DIR='/etc/hadoop/conf'
HADOOP_TOOLS_HOME='/usr/lib/hadoop'
HADOOP_TOOLS_DIR='share/hadoop/tools'
HADOOP_TOOLS_LIB_JARS_DIR='share/hadoop/tools/lib'
```

fetchdt

Сергей Остапов

Назад к содержанию

Получает Delegation Token от NameNode.

Использование показано ниже:

```
$ hdfs fetchdt <opts> <token_file_path>
```

Аргументы

--webservice NN_Url	Отображает URL для подключения к NameNode (начинается с http или https)
--renewer name	Отображает имя delegation token renewer
--cancel	Отменяет delegation token
--renew	Обновляет delegation token. Delegation token необходимо получить с помощью аргумента `-renewer`
--print	Выводит delegation token
token_file_path	Задает путь для хранения токена

fsck

Сергей Остапов

Назад к содержанию

Запускает утилиту проверки HDFS. Эта команда нужна для сообщения о проблемах с различными файлами, например, об отсутствующих блоках для файла или о недостаточно реплицированных блоках.

В отличие от традиционной утилиты `fsck` для native файловых систем, эта команда не исправляет обнаруженные ошибки. Обычно NameNode автоматически исправляет большинство восстанавливаемых сбоев. По умолчанию `fsck` игнорирует открытые файлы, но предоставляет возможность выбрать все файлы при составлении отчета.

Использование показано ниже:

```
$ hdfs fsck <path>
  [-list-corruptfileblocks] |
  [-move | -delete | -openforwrite]
  [-files [-blocks [-locations | -racks | -replicaDetails | -upgradedomains]]]
  [-includeSnapshots][-storagepolicies] [-maintenance]
  [-blockId <blk_Id>] [-replicate]
```

Аргументы

path	Указывает путь для начала проверки
-delete	Удаляет поврежденные файлы
-files	Выводит список проверяемых файлов
-files -blocks	Выводит отчет по блоку
-files -blocks -locations	Выводит местоположения каждого блока
-files -blocks -racks	Отображает топологию сети для расположения DataNode
-files -blocks -replicaDetails	Отображает подробную информацию каждой реплики
-files -blocks -upgradedomains	Отображает обновленные домены каждого блока
-move	Перемещает поврежденные файлы в */lost+found*
-openforwrite	Отображает открытые для записи файлы
-storagepolicies	Отображает сводку политики хранения для блоков
-maintenance	Отображает информацию о статусе обслуживания ноды
-blockId	Отображает информацию о блоке
-replicate	Запускает работу по репликации, чтобы неправильно реплицированные блоки соответствовали политике размещения блоков

Example:

```
$ hdfs fsck myDir/file_test.csv -files -blocks -locations
```

getconf

Сергей Остапов

Назад к содержанию

Получает информацию о конфигурации из каталога конфигурации после обработки.

Использование показано ниже:

```
$ hdfs getconf -namenodes
$ hdfs getconf -secondaryNameNodes
$ hdfs getconf -backupNodes
$ hdfs getconf -journalNodes
$ hdfs getconf -includeFile
$ hdfs getconf -excludeFile
$ hdfs getconf -nnRpcAddresses
$ hdfs getconf -confKey [key]
```

Аргументы

-namenodes	Получает список NameNode в кластере
-secondaryNameNodes	Получает список secondary NameNode в кластере
-backupNodes	Получает список backup node в кластере
-journalNodes	Получает список journal node в кластере
-includeFile	Получает путь к включаемому файлу, определяющему DataNode, которые могут присоединиться к кластеру
-excludeFile	Получает путь к файлу исключения, определяющему DataNode, которые необходимо вывести из эксплуатации
-nnRpcAddresses	Получает RPC-адреса NameNode
-confKey [key]	Получает определенный ключ из конфигурации

groups

Сергей Остапов

Назад к содержанию

Возвращает информацию о группах при наличии одного или нескольких логинов (username).

Использование показано ниже:

```
$ hdfs groups [username ...]
```

httpfs

Сергей Остапов

Назад к содержанию

Запускает сервер HttpFS, HDFS HTTP Gateway.

Использование показано ниже:

```
$ hdfs httpfs
```

lsSnapshottableDir

Сергей Остапов

Назад к содержанию

Получает список snapshottable-каталогов. Если запускается суперпользователем, то возвращаются все snapshottable-каталоги.

Использование показано ниже:

```
$ hdfs lsSnapshottableDir [-help]
```

Аргументы

-help	Отображает справочную информацию

jmxget

Сергей Остапов

Назад к содержанию

Сбрасывает информацию JMX.

Использование показано ниже:

```
$ hdfs jmxget [-localVM ConnectorURL | -port port | -server mbeanserver | -service service]
```

Аргументы

-help	Выводит справочную информацию
-localVM ConnectorURL	Подключение к ВМ на той же машине
-port mbean server port	Указывает порт сервера mbean. При отсутствии порта будет попытка подключения к MBean серверу на той же ВМ
-server	mbean server (по умолчанию `localhost`)
-service NameNode|DataNode	Указывает службу jmx (`NameNode` по умолчанию)

oev

Сергей Остапов

Назад к содержанию

Hadoop Offline Edits Viewer. Подробную информацию об этом инструменте можно найти в статье Offline Edits Viewer Guide.

Использование показано ниже:

```
$ hdfs oev [OPTIONS] -i INPUT_FILE -o OUTPUT_FILE
```

Обязательные аргументы

-i,--inputFile arg	Редактирует файл для обработки, расширение `xml` (без учета регистра) означает формат XML, любое другое имя файла означает двоичный формат
-o,--outputFile arg	Имя выходного файла. Если указанный файл существует, он будет перезаписан. Формат файла определяется опцией `-p`

Опциональные аргументы

-f,--fix-txids	Перенумеровывает идентификаторы транзакций во входных данных, чтобы не было пробелов или недействительных идентификаторов транзакций
-h,--help	Отображает справочную информацию
-r,--recover	При чтении двоичных логов редактирования использует режим восстановления. Это дает позволяет пропускать поврежденные части логов
--p,--processor arg	Тип обработчика для применения к файлу образа. В настоящее время поддерживаются следующие типы обработчиков: ▪ `binary` (собственный двоичный формат, который использует Hadoop); ▪ `xml` (по умолчанию, формат XML); ▪ `stats` (выводит статистику о редактировании файла).
-v,--verbose	Более подробный вывод, выводит имена входных и выходных файлов для обработчиков, записывающих в файл, также выводит данные на экран. Для больших файлов образов это значительно увеличит время обработки (по умолчанию `false`)

Пример:

```
$ hdfs oev -p xml -i edits -o edits.xml
```

oiv

Сергей Остапов

Назад к содержанию

Hadoop Offline Image Viewer для файлов образов.

Использование показано ниже:

```
$ hdfs oiv [OPTIONS] -i INPUT_FILE
```

Обязательные аргументы

-i,--inputFile INPUT_FILE	Указывает входной файл fsimage (или файл XML, если используется обработчик ReverseXML) для обработки

Опциональные аргументы

-o,--outputFile output file	Имя выходного файла, если указанный обработчик вывода генерирует его. Если указанный файл уже существует, он перезаписывается. Если входной файл является XML-файлом, создается файл *<outputFile>.md5*
-p,--processor processor	Обработчик образов для применения к файлу образа. В настоящее время действительными опциями являются `Web` (по умолчанию), `XML`, `Delimited`, `FileDistribution` и `ReverseXML`
-addr address	Адрес (`host:port`) для прослушивания. Эта опция используется с Web-обработчиком
-maxSize size	Диапазон [`0, maxSize`] размеров файлов для анализа в байтах (по умолчанию `128GB`). Этот параметр используется с обработчиком FileDistribution
-step size	Гранулярность распределения в байтах (по умолчанию `2MB`). Этот параметр используется с обработчиком FileDistribution
-format	Форматирует выходной результат в удобный для чтения вид вместо числа байтов. Этот параметр используется с обработчиком FileDistribution
-delimiter arg	Разделительная строка для использования в обработчике Delimited
-t,--temp temporary dir	Использует временный каталог для кеширования промежуточных результатов при генерации делимитированного вывода. Если этот аргумент не используется, обработчик Delimited конструирует пространство имен в памяти перед выводом текста
-h,--help	Отображает справочную информацию об использовании инструмента и выходит

oiv_legacy

Сергей Остапов

Назад к содержанию

Hadoop Offline Image Viewer для устаревших версий Hadoop. Подробную информацию об Offline Image Viewer можно найти в статье oiv_legacy Command.

Использование показано ниже:

```
$ hdfs oiv_legacy [OPTIONS] -i INPUT_FILE -o OUTPUT_FILE
```

Аргументы

-i, --inputFile input file	Указывает входной файл fsimage для обработки. Обязательный параметр
-o,--outputFile output file	Указывает имя выходного файла, если указанный обработчик вывода генерирует его. Если указанный файл уже существует, он перезаписывается без подтверждения. Обязательный параметр
-p|--processor processor	Указывает обработчик образа для применения к файлу образа. Возможные варианты: `Ls` (по умолчанию), `XML`, `Delimited`, `Indented`, `FileDistribution` и `NameDistribution`
-maxSize size	Указывает диапазон [`0`, `maxSize`] размеров файлов для анализа в байтах (по умолчанию `128GB`). Эта опция используется с обработчиком FileDistribution
-step size	Указывает гранулярность распределения в байтах (по умолчанию `2MB`). Эта опция используется с обработчиком FileDistribution
-format	Форматирует результат вывода в удобном для чтения виде, а не в виде количества байт. (по умолчанию `false`). Эта опция используется с обработчиком FileDistribution
-skipBlocks	Не перечисляет отдельные блоки внутри файлов. Это может сэкономить время обработки и место в файле outfile в пространствах имен с очень большими файлами. Обработчик `Ls` считывает блоки для правильного определения размеров файлов и игнорирует эту опцию
-printToScreen	Отправляет вывод обработчика на консоль, а также в указанный файл. Для очень больших пространств имен это может увеличить время обработки на порядок
-delimiter arg	При использовании вместе с обработчиком Delimited заменяет разделитель табуляции по умолчанию на строку, указанную в аргументе `arg`
-h|--help	Отображает информацию об использовании инструмента, справочную информацию и выходит

snapshotDiff

Сергей Остапов

Назад к содержанию

Определяет разницу между снепшотами HDFS.

Использование показано ниже:

```
$ hdfs snapshotDiff <path> <fromSnapshot> <toSnapshot>
```

version

Сергей Остапов

Назад к содержанию

Отображает текущую версию HDFS.

Использование показано ниже:

```
$ hdfs version
```

balancer

Сергей Остапов

Назад к содержанию

Запускает утилиту балансировщика кластера. Чтобы остановить процесс перебалансировки, администратору достаточно нажать `Ctrl+C`.

ПРИМЕЧАНИЕ

Обратите внимание, что политика blockpool строже, чем политика DataNode.

Использование показано ниже:

```
$ hdfs balancer
    [-policy <policy>]
    [-threshold <threshold>]
    [-exclude [-f <hosts-file> | <comma-separated list of hosts>]]
    [-include [-f <hosts-file> | <comma-separated list of hosts>]]
    [-source [-f <hosts-file> | <comma-separated list of hosts>]]
    [-blockpools <comma-separated list of blockpool ids>]
    [-idleiterations <idleiterations>]
    [-runDuringUpgrade]
    [-asService]
```

Аргументы

-policy <policy>	Возможные значения: - `DataNode` (default) — кластер балансируется, если каждая DataNode отбалансирована; - `Blockpool` — кластер балансируется, если каждый блок-пул в каждой DataNode отбалансирован.
threshold <threshold> | <comma-separated list of hosts>	Процент (порог) емкости диска. Это параметр переписывает стандартное значение (порог)
-exclude -f <hosts-file> | <comma-separated list of hosts>	Исключает указанные DataNode из балансировки
-include -f <hosts-file> | <comma-separated list of hosts>	Включает только указанные DataNode в балансировку
-source -f <hosts-file> | <comma-separated list of hosts>	Выбирает только указанные DataNode как исходные ноды
-blockpools <comma-separated list of blockpool ids>	Запускает балансировщик только на block pool, включенных в этот список
-idleiterations <iterations>	Указывает максимальное число бездействующих итераций перед выходом. Переписывает стандартное значение `idleiterations`
-runDuringUpgrade	Указывает, запускать ли балансировщик во время обновления HDFS. Обычно это нежелательно, так как это не повлияет на используемое пространство на перегруженных машинах
-asService	Запускает балансировщик как долго работающую службу
-h, --help	Отображает справочную информацию

cacheadmin

Сергей Остапов

Содержание

Назад к содержанию

- Команды директив кеша
  - addDirective
  - removeDirective
  - listDirectives
- Cache pool commands
  - addPool
  - modifyPool
  - removePool
  - listPools
  - help

Командная строка позволяет взаимодействовать с пулами кеша и директивами с помощью подкоманды `hdfs cacheadmin`.

Директивы кеша идентифицируются уникальным, неповторяющимся 64-битным целочисленным идентификатором. Идентификаторы не будут повторно использоваться, даже если директива кеша будет впоследствии удалена. Пулы кеша идентифицируются уникальным строковым именем.

Команды директив кеша

addDirective

Добавляет новую директиву кеша.

```
$ hdfs cacheadmin -addDirective -path <path> -pool <pool-name> [-force] [-replication <replication>] [-ttl <time-to-live>]
```

Аргументы

<path>	Путь к кешу. Можно указать путь к каталогу или файлу
<pool-name>	Указывает пул, к которому добавится директива. Для добавления новых директив необходимо иметь права записи в пул кеша
-force	Пропускает проверку лимитов ресурсов пула кеша
<replication>	Указывает используемое значение репликации кеша
<time-to-live>	Указывает время, в течение которого директива действительна. Можно указать в минутах, часах, днях и так далее. Например, `30m`, `4h`, `2d`. Если значение не указано, то директива всегда остается действительной

removeDirective

Удаляет все директивы кеша в указанном пути.

```
$ hdfs cacheadmin -removeDirective <id>
```

Аргументы

<path>	Путь к директивам кеша для удаления. Необходимо иметь разрешение на запись в пул директивы, чтобы удалить ее

listDirectives

Показывает список директив кеша.

```
$ hdfs cacheadmin -listDirectives [-stats] [-path <path>] [-pool <pool>]
```

Аргументы

<path>	Перечисляет только директивы кеша с этим путем. Если в пуле кеша есть директива кеша для пути, к которому у вас нет прав на чтение, она не будет отображена в списке
<pool>	Показывает директивы кеша только в данном пуле
-stats	Показывает статистику директив кеша на основе путей

Cache pool commands

addPool

Добавляет новый пул кеша.

```
$ hdfs cacheadmin -addPool <name> [-owner <owner>] [-group <group>] [-mode <mode>] [-limit <limit>] [-maxTtl <maxTtl>]
```

Аргументы

<name>	Имя нового пула. Если в пуле кеша есть директива кеша для пути, к которому у вас нет прав на чтение, она не будет включена в список
<owner>	Имя пользователя владельца пула. По умолчанию указывается текущий пользователь
<group>	Группа пула. По умолчанию указывается имя основной группы текущего пользователя
<mode>	Разрешения в стиле UNIX для пула. Разрешения указываются в восьмеричном формате, например, `0755`
<limit>	Максимальное количество байт, которое может быть совокупно кешировано директивами в этом пуле. По умолчанию ограничение не задается
<maxTtl>	Максимально допустимое время жизни для директив, добавляемых в пул. Это значение может быть указано в секундах, минутах, часах и днях, например, `120s`, `30m`, `4h`, `2d`. По умолчанию максимальное значение не задается Значение `never` означает, что ограничение отсутствует

modifyPool

Изменяет метаданные существующего пула кеша

```
$ hdfs cacheadmin -modifyPool <name> [-owner <owner>] [-group <group>] [-mode <mode>] [-limit <limit>] [-maxTtl <maxTtl>]
```

Аргументы

<name>	Имя пула для изменения
<owner>	Имя пользователя владельца пула
<group>	Имя группы пула
<mode>	Разрешения для пула в стиле UNIX в восьмеричном формате
<limit>	Максимальное количество байтов, которые могут быть кешированы этим пулом
<maxTtl>	Максимальное время жизни директив, добавляемых в пул

removePool

Удаляет пул кеша. При этом также удаляются пути, связанные с пулом.

```
$ hdfs cacheadmin -removePool <name>
```

Аргументы

<name>	Имя пула кеша для удаления

listPools

Отображает информацию об одном или нескольких пулах кеша, например, имя, владелец, группа, разрешения и т.д.

```
hdfs cacheadmin -listPools [-stats] [<name>]
```

Аргументы

-stats	Отображает дополнительную статистику кеша пула
<name>	Если указано, отображает только кеш под этим именем

help

Выводит подробную справку о команде.

```
$ hdfs cacheadmin -help <command-name>
```

Аргументы

<command-name>	Указывает команду, для которой вывести отобразить справку. Если команда не указана, то выводится справка по всем командам

crypto

Сергей Остапов

Содержание

Назад к содержанию

createZone

listZones

provisionTrash

getFileEncryptionInfo

reencryptZone

listReencryptionStatus

Более подробную информацию об интерфейсе командной строки crypto можно получить в статье HDFS Transparent Encryption Documentation.

Использование показано ниже:

```
$ hdfs crypto -createZone -keyName <keyName> -path <path>
$ hdfs crypto -listZones
$ hdfs crypto -provisionTrash -path <path>
$ hdfs crypto -help <command-name>
```

createZone

Создает новую зону шифрования.

```
[-createZone -keyName <keyName> -path <path>]
```

Аргументы

path	Путь к зоне шифрования, которую необходимо создать. Это должен быть пустой каталог. По этому пути создается мусорный каталог
keyName	Имя ключа, используемого для зоны шифрования. Имена ключей в верхнем регистре не поддерживаются

listZones

Показывает список всех зон шифрования. Требуются права суперпользователя.

```
[-listZones]
```

provisionTrash

Создает мусорный каталог для зоны шифрования.

```
[-provisionTrash -path <path>]
```

Аргументы

path	Путь к корневой директории зоны шифрования

getFileEncryptionInfo

Получает информацию о шифровании из файла. С ее помощью можно узнать, зашифрован ли файл, а также имя/версию ключа, используемые для его шифрования.

```
[-getFileEncryptionInfo -path <path>]
```

Arguments

path	Путь к файлу для получения информации о шифровании

reencryptZone

Перешифровывает зону шифрования. Выполняет итерацию по зоне шифрования, вызывая KeyProvider-интерфейс `reencryptEncryptedKeys` для пакетного повторного шифрования всех файлов с помощью ключа зоны шифрования последней версии в поставщике ключей. Требуются права суперпользователя.

```
[-reencryptZone <action> -path <zone>]
```

> **ПРИМЕЧАНИЕ**
>
> Повторное шифрование не применяется к моментальным снимкам, поскольку моментальные снимки неизменяемы.

Аргументы

action	Действие повторного шифрования, которое необходимо выполнить. Можно указать или `-start`, или `-cancel`
path	Путь к корневому каталогу зоны шифрования

Повторное шифрование — это операция, полностью выполняемая NameNode, поэтому может привести к высокой нагрузке на NameNode. Следующие настройки позволяют контролировать нагрузку на NameNode, учитывая допустимую пропускную способность кластера.

dfs.namenode.reencrypt.batch.size	Количество зашифрованных data encryption keys (EDEKs) в пакете для отправки на KMS для повторного шифрования. Для обработки каждого пакета задействуется системная блокировка на чтение/запись, а регулирование (throttling) выполняется между пакетами.
dfs.namenode.reencrypt.throttle.limit.handler.ratio	Соотношение блокировок на чтение для повторного шифрования. `1.0` означает, что регулировка не выполняется (no throttling). `0.5` означает, что процесс повторного шифрования может удерживать блокировку на чтение до 50% от общего времени обработки. Отрицательные значения или `0` не принимаются
dfs.namenode.reencrypt.throttle.limit.updater.ratio	Соотношение блокировок на запись для повторного шифрования. `1.0` означает, что регулировка не выполняется (no throttling). `0.5` означает, что процесс повторного шифрования может удерживать блокировку на чтение до 50% от общего времени обработки. Отрицательные значения или `0` не принимаются

listReencryptionStatus

Выводит список информации о повторном шифровании для всех зон шифрования. Требуются права суперпользователя.

```
[-listReencryptionStatus]
```

datanode

Сергей Остапов

Назад к содержанию

Запускает DataNode в HDFS.

Использование показано ниже:

```
$ hdfs datanode [-regular | -rollback | -rollingupgrade rollback]
```

Аргументы

-regular	Запускает DataNode в обычном режиме (по умолчанию)
-rollback	Откатывает к предыдущей версии DataNode
-rollingupgrade rollback	Откатывает процесс постепенного обновления

dfsadmin

Сергей Остапов

Назад к содержанию

Запускает клиент HDFS dfsadmin.

Использование показано ниже:

```
$ hdfs dfsadmin [-report [-live] [-dead] [-decommissioning] [-enteringmaintenance] [-inmaintenance]]
$ hdfs dfsadmin [-safemode enter | leave | get | wait | forceExit]
$ hdfs dfsadmin [-saveNamespace [-beforeShutdown]]
$ hdfs dfsadmin [-rollEdits]
$ hdfs dfsadmin [-restoreFailedStorage true |false |check]
$ hdfs dfsadmin [-refreshNodes]
$ hdfs dfsadmin [-setQuota <quota> <dirname>...<dirname>]
$ hdfs dfsadmin [-clrQuota <dirname>...<dirname>]
$ hdfs dfsadmin [-setSpaceQuota <quota> [-storageType <storagetype>] <dirname>...<dirname>]
$ hdfs dfsadmin [-clrSpaceQuota [-storageType <storagetype>] <dirname>...<dirname>]
$ hdfs dfsadmin [-finalizeUpgrade]
$ hdfs dfsadmin [-rollingUpgrade [<query> |<prepare> |<finalize>]]
$ hdfs dfsadmin [-upgrade [query | finalize]]
$ hdfs dfsadmin [-refreshServiceAcl]
$ hdfs dfsadmin [-refreshUserToGroupsMappings]
$ hdfs dfsadmin [-refreshSuperUserGroupsConfiguration]
$ hdfs dfsadmin [-refreshCallQueue]
$ hdfs dfsadmin [-refresh <host:ipc_port> <key> [arg1..argn]]
$ hdfs dfsadmin [-reconfig <namenode|datanode> <host:ipc_port> <start |status |properties>]
$ hdfs dfsadmin [-printTopology]
$ hdfs dfsadmin [-refreshNamenodes datanodehost:port]
$ hdfs dfsadmin [-getVolumeReport datanodehost:port]
$ hdfs dfsadmin [-deleteBlockPool datanode-host:port blockpoolId [force]]
$ hdfs dfsadmin [-setBalancerBandwidth <bandwidth in bytes per second>]
$ hdfs dfsadmin [-getBalancerBandwidth <datanode_host:ipc_port>]
$ hdfs dfsadmin [-fetchImage <local directory>]
$ hdfs dfsadmin [-allowSnapshot <snapshotDir>]
$ hdfs dfsadmin [-disallowSnapshot <snapshotDir>]
$ hdfs dfsadmin [-shutdownDatanode <datanode_host:ipc_port> [upgrade]]
$ hdfs dfsadmin [-evictWriters <datanode_host:ipc_port>]
$ hdfs dfsadmin [-getDatanodeInfo <datanode_host:ipc_port>]
$ hdfs dfsadmin [-metasave filename]
$ hdfs dfsadmin [-triggerBlockReport [-incremental] <datanode_host:ipc_port> [-namenode <namenode_host:ipc_port>]]
$ hdfs dfsadmin [-listOpenFiles [-blockingDecommission] [-path <path>]]
$ hdfs dfsadmin [-help [cmd]]
```

Аргументы

--report [-live] [-dead] [-decommissioning] [-enteringmaintenance] [-inmaintenance]	Сообщает основную информацию о файловой системе и статистику. Использование `dfs` может отличаться от использования `du`, поскольку измеряет необработанное пространство, используемое репликацией, контрольными суммами, моментальными снимками и т.д. на всех DataNode. Дополнительные флаги могут использоваться для фильтрации списка отображаемых DataNodes
-safemode enter|leave|get|wait|forceExit	Команда обслуживания безопасного режима. Безопасный режим — это состояние узла имен, в котором он: - не принимает изменения в пространстве имен (только для чтения); - не реплицирует и не удаляет блоки. Безопасный режим вводится автоматически при запуске узла NameNode и автоматически выходит из безопасного режима, когда настроенный минимальный процент блоков удовлетворяет условию минимальной репликации. Если NameNode обнаружит какую-либо аномалию, то он будет находиться в безопасном режиме до тех пор, пока эта проблема не будет решена. Если эта аномалия является следствием преднамеренного действия, то администратор может использовать `-safemode forceExit` для выхода из безопасного режима. Случаи, когда может потребоваться принудительный выход, следующие: - метаданные NameNode не согласованы. Если NameNode обнаруживает, что метаданные были изменены вне диапазона и могут привести к потери данных, то NameNode перейдет в состояние `forceExit`. В этот момент пользователь может либо перезапустить NameNode с правильными файлами метаданных, либо принудительно выйти из системы (если потеря данных допустима); - откат приводит к замене метаданных, и в редких случаях он может вызвать состояние `forceExit` безопасного режима в NameNode. В этом случае вы можете продолжить работу, передав команду `-safemode forceExit`. В безопасный режим можно также войти вручную, но тогда и выключить его тоже можно только вручную
-saveNamespace [-beforeShutdown]	Сохранить текущее пространство имен в каталогах хранения и сбросить журнал правок. Если задана опция `beforeShutdown`, узел NameNode выполняет контрольную точку тогда и только тогда, когда ни одна контрольная точка не была выполнена в течение временного окна (настраиваемое количество периодов контрольных точек). Обычно этот параметр используется перед выключением узла NameNode для предотвращения возможного повреждения fsimage/editlog
-rollEdits	Сворачивает логи редактирования на активной NameNode
-restoreFailedStorage true|false|check	Эта опция включает/отключает автоматическую попытку восстановления отказавших реплик хранилища. Если отказавшее хранилище снова станет доступным, система попытается восстановить правки и/или fsimage во время контрольной точки
-refreshNodes	Перечитывает файлы hosts и exclude, чтобы обновить набор DataNodes, которым разрешено подключаться к NameNode, и те, которые должны быть выведены из эксплуатации или вновь введены в эксплуатацию
-setQuota <quota> <dirname>…<dirname>	Устанавливает квоту для каждого каталога
-clrQuota <dirname>…<dirname>	Удаляет любую квоту имен для каждого каталога. Лучше всего для каждого каталога, с сообщением об ошибках, если каталог не существует или является файлом. Не является ошибкой, если каталог не имеет квоты
-setSpaceQuota <quota> [-storageType <storagetype>] <dirname>…<dirname>	Устанавливает квоту типа хранения на байты типа хранения, указанного для каждого каталога
-clrSpaceQuota [-storageType <storagetype>] <dirname>…<dirname>	Удаляет квоты типа хранилища, указанной для каждого каталога. Лучше всего для каждого каталога, с сообщением об ошибках, если каталог не существует или является файлом. Это не является ошибкой, если для каталога не установлена квота для указанного типа хранения Квота для конкретного типа хранения удаляется при указании опции `-storageType`
-finalizeUpgrade	Завершает обновления HDFS. Узлы данных удаляют свои рабочие каталоги предыдущей версии, затем то же самое делает узел имени. Это завершает процесс обновления
-rollingUpgrade [<query>|<prepare>|<finalize>]	Выполняет действия по скользящему обновлению: - `query` — запрос текущего состояния скользящего обновления; - `prepare` — подготовить новое скользящее обновление; - `finalize` — завершить текущее скользящее обновление.
-upgrade query|finalize	Выполняет запрос текущего статуса обновления. Завершить обновление HDFS (эквивалентно `-finalizeUpgrade`)
-refreshServiceAcl	Перезагружает файл политики авторизации на уровне сервиса
-refreshUserToGroupsMappings	Обновляет соответствий пользователей с группами
-refreshSuperUserGroupsConfiguration	Обновляет соответствия прокси групп суперпользователя
-refreshCallQueue	Перезагружает очередь вызовов из конфигурации
-refresh <host:ipc_port> <key> [arg1..argn]	Запускает runtime-refresh ресурса, указанного `key`, на `host:ipc_port`. Все остальные последующие аргументы отправляются на хост
-reconfig <datanode |namenode> <host:ipc_port> <start|status|properties>	Запускает реконфигурацию или получает статус текущей реконфигурации, или получает список реконфигурируемых свойств. Второй параметр задает тип ноды
-printTopology	Выводит дерево стоек и их узлов, о которых сообщает NameNode
-refreshNameNodes datanodehost:port	Для данной DataNode перезагружает файлы конфигурации, прекращает обслуживание удаленных блок-пулов и начинает обслуживать новые блок-пулы
-getVolumeReport datanodehost:port	Получить отчет о томе для данной DataNode
-deleteBlockPool datanode-host:port blockpoolId [force]	Если передано значение `force`, каталог блок-пула для данного идентификатора блок-пула на данной DataNode удаляется вместе с его содержимым, иначе каталог удаляется, только если он пуст. Команда завершится с ошибкой, если DataNode все еще обслуживает пул блоков. Используйте команду `-refreshNameNodes` для отключения службы данных блок-пулов на DataNode
-setBalancerBandwidth <bandwidth in bytes per second>	Изменяет пропускную способность сети, используемую каждой DataNode во время балансировки блоков HDFS. `<bandwidth>` — это максимальное количество байт в секунду, которое будет использоваться каждым DataNode. Это значение переопределяет параметр `fs.datanode.balance.bandwidthPerSec`. Новое значение не сохраняется на DataNode
-getBalancerBandwidth <datanode_host:ipc_port>	Получает пропускную способность сети (в байтах в секунду) для данной DataNode. Это максимальная пропускная способность сети, используемая узлом данных во время балансировки блоков HDFS
-fetchImage <local directory>	Загружает последний fsimage с узла NameNode и сохраняет его в указанном локальном каталоге
-allowSnapshot <snapshotDir>	Разрешение создания моментальных снимков каталога. Если операция завершается успешно, каталог становится таблицей моментальных снимков
-disallowSnapshot <snapshotDir>	Запрет на создание моментальных снимков каталога Все моментальные снимки каталога должны быть удалены перед запрещением создания моментальных снимков
-shutdownDatanode <datanode_host:ipc_port> [upgrade]	Отправляет запрос на отключение для данной DataNode
-evictWriters <datanode_host:ipc_port>	Удаляет все клиенты, которые записывают блок в DataNode. Это полезно, если вывод из эксплуатации завис из-за медленной записи
-getDatanodeInfo <datanode_host:ipc_port>	Получает информацию о данной DataNode
*-metasave filename_*	Сохраняет первичные структуры данных NameNode в файл в директории, указанной свойством `hadoop.log.dir`. Имя файла будет перезаписано, если оно существует. Имя файла будет содержать по одной строке для каждого из следующих параметров: - DataNodes heart beating с NameNode; - Блоки, ожидающие репликации; - Блоки, реплицируемые в настоящее время; - Блоки, ожидающие удаления.
-triggerBlockReport [-incremental] <datanode_host:ipc_port> [-namenode <namenode_host:ipc_port>]	Запускает блока отчета для данной DataNode Если указано `-incremental`, то это будет полный блочный отчет. Если указано `-namenode <namenode_host:ipc_port>`, отправляет блочный отчет только на указанный NameNode
-listOpenFiles [-blockingDecommission] [-path <path>]	Показывает список всех открытых файлов, которыми в настоящее время управляет NameNode вместе с именем клиента и клиентской машиной, осуществляющей доступ к ним. Список открытых файлов будет отфильтрован по заданному типу и пути. Параметр `-blockingDecommission`, позволяет перечислить только те открытые файлы, которые блокируют вывод узла данных из эксплуатации
-help [cmd]	Отображает справку для данной команды или для всех команд, если никакая из команд не указана

dfsrouter

Сергей Останов

Назад к содержанию

Запускает маршрутизатор DFS. Больше информации доступно на странице HDFS Router-based Federation.

Использование показано ниже:

```
$ hdfs dfsrouter
```

dfsrouteradmin

Сергей Остапов

Назад к содержанию

Данные команды управляют федерацией на основе маршрутизаторов.

Использование показано ниже:

```
$ hdfs dfsrouteradmin
    [-add <source> <nameservice1, nameservice2, ...> <destination> [-readonly] [-faulttolerant] [-order HASH|LOCAL|RANDOM|HASH_ALL]
-owner <owner> -group <group> -mode <mode>]
    [-update <source> [<nameservice1, nameservice2, ...> <destination>] [-readonly true|false] [-faulttolerant true|false] [-order
HASH|LOCAL|RANDOM|HASH_ALL] -owner <owner> -group <group> -mode <mode>]
    [-rm <source>]
    [-ls [-d] <path>]
    [-getDestination <path>]
    [-setQuota <path> -nsQuota <nsQuota> -ssQuota <quota in bytes or quota size string>]
    [-setStorageTypeQuota <path> -storageType <storage type> <quota in bytes or quota size string>]
    [-clrQuota <path>]
    [-clrStorageTypeQuota <path>]
    [-safemode enter | leave | get]
    [-nameservice disable | enable <nameservice>]
    [-getDisabledNameservices]
    [-refresh]
    [-refreshRouterArgs <host:ipc_port> <key> [arg1..argn]]
    [-refreshSuperUserGroupsConfiguration]
```

Аргументы

-add source nameservices destination	Добавляет запись в таблицу монтирования или обновить ее, если она существует
-update source nameservices destination	Обновляет атрибуты записи таблицы монтирования
-rm source	Удаляет точку монтирования указанного пути
--ls [-d] path	Список точек монтирования по указанному пути. Укажите параметр `-d` для получения подробного списка
-getDestination path	Получает подкластер, в котором находится или должен быть создан файл
-setQuota path -nsQuota nsQuota -ssQuota ssQuota	Устанавливает квоту для указанного пути
-setStorageTypeQuota path -storageType storageType stQuota	Очищает квоту данной точки монтирования
-clrQuota path	Устанавливает квоту типа хранения для указанного пути
-clrStorageTypeQuota path	Очищает квоту типа хранения для данной точки монтирования
-safemode enter leave get	Ручная настройка входа или выхода маршрутизатора из безопасного режима. Опция `get` будет использоваться для проверки того, находится ли маршрутизатор в безопасном режиме
-nameservice disable enable nameservice	Отключает/включает службы имен из федерации. Если отключить, запросы не будут направляться в эту службу имен
-getDisabledNameservices	Получает службы имен, отключенных в федерации
-refresh	Обновляет кеш таблицы монтирования подключенного маршрутизатора
-refreshRouterArgs <host:ipc_port> <key> [arg1..argn]	Запускает обновление ресурса, указанного `key`, на `host:ipc_port`. Например, чтобы включить проверку белого списка, достаточно послать команду `refresh`, а не перезапускать сервер маршрутизатора
-refreshSuperUserGroupsConfiguration	Обновляет сопоставления прокси-групп суперпользователей на маршрутизаторе

diskbalancer

Сергей Остапов

Назад к содержанию

Запускает инструмент diskbalancer.

Использование показано ниже:

```
$ hdfs diskbalancer
     [-plan <datanode> -fs <namenodeURI>]
     [-execute <planfile>]
     [-query <datanode>]
     [-cancel <planfile>]
     [-cancel <planID> -node <datanode>]
     [-report -node <file://> | [<DataNodeID|IP|Hostname>,...]]
     [-report -node -top <topnum>]
```

Аргументы

-plan	Создает план diskbalancer
-execute	Выполняет созданный план на DataNode
-query	Получает текущий статус diskbalancer от DataNode
-cancel	Отменяет запущенный план
-report	Сообщает информацию об объеме от DataNode

ec

Сергей Останов

Назад к содержанию

Запускает CLI ErasureCoding.

Использование показано ниже:

```
$ hdfs ec [generic options]
     [-setPolicy -policy <policyName> -path <path>]
     [-getPolicy -path <path>]
     [-unsetPolicy -path <path>]
     [-listPolicies]
     [-addPolicies -policyFile <file>]
     [-listCodecs]
     [-enablePolicy -policy <policyName>]
     [-disablePolicy -policy <policyName>]
     [-removePolicy -policy <policyName>]
     [-verifyClusterSetup -policy <policyName>...<policyName>]
     [-help [cmd ...]]
```

Аргументы

-setPolicy	Задает определенную политику ErasureCoding для директории
-getPolicy	Получает информацию о политике ErasureCoding для указанного пути
-unsetPolicy	Отменяет политику ErasureCoding, установленную предыдущей командой `setPolicy`
-listPolicies	Отображает все доступные политики ErasureCoding
-addPolicies	Добавляет политики ErasureCoding
-listCodecs	Получает список поддерживаемых кодеков ErasureCoding и кодировщиков в системе
-enablePolicy	Включает политику ErasureCoding в системе
-disablePolicy	Отключает политику ErasureCoding в системе
-removePolicy	Удаляет политику ErasureCoding из системы
-verifyClusterSetup	Проверяет, что конфигурация кластера поддерживает список политик ErasureCoding

haadmin

Сергей Остапов

Назад к содержанию

Запускает управление режимом высокой доступности (High Availability, HA) для HDFS-кластера.

Использование показано ниже:

```
$ hdfs haadmin -transitionToActive <serviceId> [--forceactive]
$ hdfs haadmin -transitionToStandby <serviceId>
$ hdfs haadmin -transitionToObserver <serviceId>
$ hdfs haadmin -failover [--forcefence] [--forceactive] <serviceId> <serviceId>
$ hdfs haadmin -getServiceState <serviceId>
$ hdfs haadmin -getAllServiceState
$ hdfs haadmin -checkHealth <serviceId>
$ hdfs haadmin -help <command>
```

Аргументы

-checkHealth	Проверяет состояние выбранной NameNode
-failover	Запускает обход отказа между двумя NameNode
-getServiceState	Определяет, является ли выбранная NameNode Active или Standby
-getAllServiceState	Возвращает статус всех NameNode
-transitionToActive	Переводит выбранную NameNode в статус Active (ограждение не производится)
-transitionToStandby	Переводит выбранную NameNode в статус Standby (ограждение не производится)
-transitionToObserver	Переводит выбранную NameNode в статус Observer (ограждение не производится)
-help [cmd]	Отображает справочную информацию по указанной команде или по всем командам, если конкретная не указана

journalnode

Сергей Остапов

Назад к содержанию

Запускает JournalNode для работы с HDFS HA с QJM (Quorum Journal Manager).

Использование показано ниже:

```
$ hdfs journalnode
```

mover

Сергей Остапов

Назад к содержанию

Запускает утилиту миграции данных, она похожа на Balancer. Эта утилита периодически сканирует файлы в HDFS, чтобы проверить, соответствует ли размещение блоков политике хранения. Для блоков, нарушающих политику хранения, она перемещает реплики на другой тип хранилища, чтобы выполнить требования политики хранения.

Эта утилита перемещает реплики блоков в пределах одной node, когда это возможно. Если это невозможно (например, если node не имеет целевого типа хранилища), то она будет копировать блочные реплики на другую node по сети.

ПРИМЕЧАНИЕ

Если опции `-p` и `-f` не указаны, путь по умолчанию — это корневой каталог.

Использование показано ниже:

```
$ hdfs mover [-p <files/dirs> | -f <local file name>]
```

Аргументы

-f <local file>	Локальный файл, содержащий список файлов/директорий HDFS для миграции
-p <files/dirs>	Разделенный пробелами список файлов/директорий HDFS для миграции

namenode

Сергей Остапов

Назад к содержанию

Запускает NameNode.

Использование показано ниже:

```
$ hdfs namenode [-backup] |
          [-checkpoint] |
          [-format [-clusterid cid ] [-force] [-nonInteractive] ] |
          [-upgrade [-clusterid cid] [-renameReserved<k-v pairs>] ] |
          [-upgradeOnly [-clusterid cid] [-renameReserved<k-v pairs>] ] |
          [-rollback] |
          [-rollingUpgrade <rollback |started> ] |
          [-importCheckpoint] |
          [-initializeSharedEdits] |
          [-bootstrapStandby [-force] [-nonInteractive] [-skipSharedEditsCheck] ] |
          [-recover [-force] ] |
          [-metadataVersion ]
```

Аргументы

-backup	Запускает Backup node
-checkpoint	Запускает Checkpoint node
-format [-clusterid cid]	Форматирует указанную NameNode. Запускает NameNode, форматирует ее и затем выключает. Возвращает NameNodeFormatException, если директория уже существует и если переформатирование отключено для кластера
-upgrade [-clusterid cid] [-renameReserved <k-v pairs>]	NameNode должна запускаться с опцией `upgrade` после выхода новой версии Arenadata Hyperwave
-upgradeOnly [-clusterid cid] [-renameReserved <k-v pairs>]	Обновляет указанную NameNode и затем выключить ее
-rollback	Откатывает NameNode к предыдущей версии Arenadata Hyperwave. Это можно выполнять после остановки кластера и выхода
-rollingUpgrade <rollback|started>	При выполнении скользящего обновления, параметр запуска NameNode `-rollingUpgrade` используется для указания различных опций скользящего обновления: ▪ `rollback`: возвращает NameNode и пользовательские данные к состоянию, предшествующему обновлению; ▪ `started`: указывает, что скользящее обновление уже началось, чтобы NameNode разрешила каталоги изображений с разными версиями компоновки во время запуска.
-importCheckpoint	Загружает образ из каталога контрольной точки и сохраняет его в текущем каталоге. Каталог контрольной точки считывается из свойства `dfs.namenode.checkpoint.dir`
-initializeSharedEdits	Форматирует новый общий каталог редактирования и копирует в него достаточное количество сегментов логов редактирования, чтобы standby NameNode могла запуститься
-bootstrapStandby [-force] [-nonInteractive] [-skipSharedEditsCheck]	Позволяет каталогам хранения standby NameNode загружаться путем копирования последнего снимка пространства имен с активной NameNode. Этот параметр используется при первой настройке кластера HA. Параметр `-force` или `-nonInteractive` имеет то же значение, что и параметр, описанный в команде namenode -format
-recover [-force]	Восстанавливает утраченные метаданные в поврежденной файловой системе
-metadataVersion	Проверяет существование сконфигурированных каталогов, затем выводит версии метаданных программного обеспечения и образа

nfs3

Сергей Остапов

Назад к содержанию

Запускает NFS3 gateway для использования со службой HDFS NFS3.

Использование показано ниже:

```
$ hdfs nfs3
```

portmap

Сергей Остапов

Назад к содержанию

Запускает RPC portmap для использования со службой HDFS NFS3.

Использование показано ниже:

```
$ hdfs portmap
```

secondarynamenode

Сергей Остапов

Назад к содержанию

Запускает secondary NameNode HDFS.

Использование показано ниже:

```
$ hdfs secondarynamenode [-checkpoint [force]] | [-format] | [-geteditsize]
```

Аргументы

-checkpoint [force]	Устанавливает контрольную точку secondary NameNode, если размер EditLog >= `fs.checkpoint.size`. Если используется `-force`, то контрольная точка не зависит от размера EditLog
-format	Форматирует локальное хранилище при запуске
-geteditsize	Выводит количество транзакций на NameNode с неустановленными контрольными точками

storagepolicies

Сергей Остапов

Назад к содержанию

Возвращает/устанавливает/отменяет политики хранения.

Использование показано ниже:

```
$ hdfs storagepolicies
      [-listPolicies]
      [-setStoragePolicy -path <path> -policy <policy>]
      [-getStoragePolicy -path <path>]
      [-unsetStoragePolicy -path <path>]
      [-satisfyStoragePolicy -path <path>]
      [-isSatisfierRunning]
      [-help <command-name>]
```

Аргументы

-listPolicies	Выводит список всех политик хранения
-setStoragePolicy -path <path> -policy <policy>	Устанавливает политику хранения для файла или каталога: ▪ `<path>` — путь, указывающий либо на каталог, либо на файл; ▪ `<policy>` — имя политики хранения.
-getStoragePolicy -path <path>	Получает политику хранения файла или каталога. `<path>` — путь, ссылающийся либо на каталог, либо на файл
-unsetStoragePolicy -path <path>	Отменяет политику хранения для файла или каталога. `<path>` — путь, указывающий либо на каталог, либо на файл. После команды unset будет применена политика хранения ближайшего предка, а если ни у одного предка нет политики, то будет применена политика хранения по умолчанию
-satisfyStoragePolicy -path <path>	Устанавливает расписание для перемещения блоков на основе текущей политики хранения файла/каталога. `<path>` — путь, указывающий либо на каталог, либо на файл

zkfc

Сергей Остапов

Назад к содержанию

Запускает процесс ZooKeeper Failover Controller для использования с HDFS HA с QJM (Quorum Journal Manager).

Использование показано ниже:

```
$ hdfs zkfc [-formatZK [-force] [-nonInteractive]]
```

Аргументы

-formatZK	Форматирует экземпляр ZooKeeper: ■ `-force` — форматирует Znode, если Znode существует; ■ `noninteractive` — форматирует Znode.
-h	Отображает справочную информацию об использовании команды

computeMeta

Сергей Остапов

Назад к содержанию

Вычисляет метаданные HDFS из блочных файлов. Если указан блочный файл, то вычисляются контрольные суммы из блочного файла. Они сохраняются в указанном выходном файле метаданных.

ВНИМАНИЕ

Будьте осторожны при использовании этой команды. Если блочный файл поврежден и вы перезаписали его метафайл, он будет отображаться как "хороший" в HDFS, но без возможности прочитать данные. Используйте только в случае крайней необходимости и когда вы на 100% уверены, что блочный файл не поврежден.

Использование показано ниже:

```
$ hdfs debug computeMeta -block <block-file> -out <output-metadata-file>
```

Аргументы

-block block-file	Указывает абсолютный путь к блочному файлу в локальной файловой системы DataNode
-out output-metadata-file	Указывает абсолютный путь к выходному файлу метаданных для хранения вычисленных контрольных сумм блочного файла

recoverLease

Сергей Остапов

Назад к содержанию

Восстанавливает файл блокировки (lease) по указанному пути. Путь должен находиться в файловой системе HDFS. По умолчанию количество повторных попыток равно `1`.

Использование показано ниже:

```
$ hdfs debug recoverLease -path <path> [-retries <num-retries>]
```

Аргументы

-path <path>	Путь HDFS для восстановления блокировки
[-retries <num-retries>]	Число попыток вызова `recoverLease`. Число попыток по умолчанию: `1`

verifyMeta

Сергей Остапов

Назад к содержанию

Проверяет метаданные и блочных файлов HDFS. Если указан блочный файл, то проверяется, что контрольные суммы в файле метаданных соответствуют блочному файлу.

Использование показано ниже:

```
$ hdfs debug verifyMeta -meta <metadata-file> [-block <block-file>]
```

Аргументы

-block block-file	Необязательный параметр для указания абсолютного пути к блочному файлу в локальной файловой системе DataNode
-meta metadata-file	Абсолютный путь к файлу метаданных в локальной файловой системе DataNode

FileSystem shell

Сергей Остапов

Содержание

Назад к содержанию

Обзор

Команды

Обзор

Команды файловой системы (File System, FS) — это различные shell-подобные команды, которые напрямую взаимодействуют с Hadoop Distributed File System (HDFS), а также с другими файловыми системами, которые поддерживает Hadoop, такими как локальная файловая система, WebHDFS, S3 FS и другими.

Использование FS shell показано ниже:

```
$ bin/hadoop fs <args>
```

Все команды оболочки FS принимают в качестве аргументов URI пути Формат URI: *scheme://authority/path*. Схема для HDFS — `hdfs`, а схема для локальной FS — `file`. Указание схемы и прав доступа необязательно. Если они не указаны, используется схема по умолчанию, заданная в конфигурации. Файл или каталог HDFS, например */parent/child*, может быть указан как *hdfs://namenodehost/parent/child* или просто как */parent/child* (если в конфигурации задан адрес *hdfs://namenodehost*).

Большинство команд FS shell ведут себя подобно соответствующим командам Unix. Различия описаны для каждой из команд. Информация об ошибках отправляется в stderr, а вывод — в stdout.

Если используется HDFS, то команда `hdfs dfs` будет синонимична.

Можно использовать относительные пути. Для HDFS текущим рабочим каталогом является домашний каталог HDFS */user/<username>*, который часто приходится создавать вручную. К домашнему каталогу HDFS также можно обращаться неявно, например, при использовании папки HDFS trash — каталог *.Trash* в домашнем каталоге.

Команды

Команда	Описание
appendToFile	Добавляет данные из источника на локальной файловой системе в конечный файл
cat	Выводит содержимое файла в stdout
checksum	Возвращает информацию о контрольной сумме файла
chgrp	Изменяет группу файлов
chmod	Изменяет права доступа к файлам
chown	Изменяет владельца файлов
copyFromLocal	Идентична команде put
copyToLocal	Идентична команде get
count	Возвращает количество каталогов и файлов
cp	Копирует файлы из источника в место назначения
createSnapshot	Создаёт моментальный снимок snapshottable-каталога
deleteSnapshot	Удаляет моментальный снимок из snapshottable-каталога
df	Отображает свободное пространство
du	Отображает размеры файлов и каталогов
dus	Отображает информацию о размере файлов
expunge	Постоянное удаление файлов в контрольных точках старше порога сохранения из каталога корзины и создание новой контрольной точки
find	Находит все файлы, соответствующие указанному выражению, и применяет к ним выбранные действия
get	Копирует файлы в локальную файловую систему
getfacl	Отображает ACL файлов и каталогов
getfattr	Отображает имена и значения расширенных атрибутов (если они есть) для файла или каталога
getmerge	Принимает исходный каталог и конечный файл в качестве входных данных и объединяет файлы в локальный целевой файл
head	Выводит первый килобайт файла в stdout
help	Выводит справочную информацию об использовании всех команд
ls	Возвращает статистику по файлу или список дочерних элементов для каталога
lsr	Рекурсивная версия ls
mkdir	Принимает URI-пути в качестве аргумента и создает каталоги
moveFromLocal	Идентична команде put
moveToLocal	Перемещает файл/каталог из HDFS в локальную файловую систему
mv	Перемещает файлы из источника в пункт назначения
put	Копирует исходные файлы из локальной в конечную файловую систему
renameSnapshot	Переименовывает моментальный снимок
rm	Удаляет файлы, указанные в качестве аргументов
rmdir	Удаляет каталог
rmr	Рекурсивная версия команды rm
setfacl	Устанавливает ACL для файлов и каталогов
setfattr	Устанавливает имя и значение расширенного атрибута для файла или каталога
setrep	Изменяет коэффициент репликации файла
stat	Выводит статистику о файле/каталоге по заданному адресу
tail	Выводит последний килобайт файла в stdout
test	Запускает тестирование
text	Принимает исходный файл и выводит его в текстовом формате
touch	Обновляет время доступа и модификации файла, указанного URI, до текущего времени
touchz	Создаёт пустой файл
truncate	Уменьшает (truncate) все файлы, которые соответствуют указанному шаблону файла, до указанной длины
usage	Выводит справочную информацию для отдельной команды

appendToFile

Сергей Остапов

Назад к содержанию

Добавляет один или несколько источников из локальной файловой системы в целевую файловую систему. Также считывает ввод из stdin и добавляет его в целевую файловую систему. Возвращает `0` при успешной попытке и `-1` при ошибке.

Целевой файл создается, если он не существовал ранее.

Использование показано ниже:

```
$ hadoop fs -appendToFile <localsrc> ... <dst>
```

Пример:

```
$ hadoop fs -appendToFile localfile /user/hadoop/hadoopfile
$ hadoop fs -appendToFile localfile1 localfile2 /user/hadoop/hadoopfile
$ hadoop fs -appendToFile localfile hdfs://nn.example.com/hadoop/hadoopfile
$ hadoop fs -appendToFile - hdfs://nn.example.com/hadoop/hadoopfile
```

cat

Сергей Остапов

Назад к содержанию

Выводит содержимое HDFS-файла в stdout.

Использование показано ниже:

```
$ hadoop fs -cat <path/to/file>
```

Аргументы

-ignoreCrc	Отключает проверку контрольной суммы

Пример:

```
$ hadoop fs -cat /user/admin/testload/file_test.csv
$ hadoop fs -checksum file:///etc/hosts
```

checksum

Сергей Остапов

Назад к содержанию

Возвращает информацию о контрольной сумме (checksum) файла.

Использование показано ниже:

```
$ hadoop fs -checksum [-v] URI
```

Аргументы

-v	Отображает размер блоков для файла

Пример:

```
$ hadoop fs -checksum hdfs://nn1.example.com/file1
$ hadoop fs -checksum file:///etc/hosts
```

chgrp

Сергей Остапов

Назад к содержанию

Изменяет группу заданных файлов.
Пользователь должен быть владельцем файлов или суперпользователем.

Использование показано ниже:

```
$ hadoop fs -chgrp [-R] GROUP URI [URI ...]
```

Аргументы

-R	Вносит изменения рекурсивно для текущего каталога и его содержимого

chmod

Сергей Остапов

Назад к содержанию

Изменяет права доступа к файлам.
Пользователь должен быть владельцем файла или суперпользователем.

Использование показано ниже:

```
$ hadoop fs -chmod [-R] <MODE[,MODE]... | OCTALMODE> URI [URI ...]
```

Аргументы

-R	Вносит изменения рекурсивно для текущего каталога и его содержимого

chown

Сергей Остапов

Назад к содержанию

Изменяет владельца файлов.
Пользователь должен быть суперпользователем.

Использование показано ниже:

```
$ hadoop fs -chown [-R] [OWNER][:[GROUP]] URI [URI ]
```

Аргументы

-R	Вносит изменения рекурсивно для текущего каталога и его содержимого

copyFromLocal

Сергей Остапов

Назад к содержанию

Команда работает схожим образом с put, а исходными данными являются файлы на локальной файловой системе.

Аргументы

-d	Пропускает создание временных файлов с суффиксом *.COPYING*
-f	Перезаписывает конечный файл, если такой уже существует
-l	Позволяет DataNode "лениво" сохранять файлы на диск, принудительно использует фактор репликации 1. Данный параметр приводит к снижению устойчивости, используйте его с осторожностью
-p	Сохраняет время доступа и изменения, владельца файла и разрешения

copyToLocal

Сергей Остапов

Назад к содержанию

Команда работает схожим образом с get, а конечной файловой системой для копирования должна быть локальная файловая система.

Использование показано ниже:

```
$ hadoop fs -copyToLocal [-ignorecrc] [-crc] URI <localdst>
```

count

Сергей Остапов

Назад к содержанию

Подсчитывает количество каталогов, файлов и байтов по путям, которые соответствуют заданному шаблону файла.
Получает квоту и использование.
Выходными колонками с `-count` являются: DIR_COUNT, FILE_COUNT, CONTENT_SIZE, PATHNAME. Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -count [-q] [-h] [-v] [-x] [-t [<storage type>]] [-u] [-e] <paths>
```

Аргументы

-u	Управляет тем, какие столбцы содержит вывод: QUOTA, REMAINING_QUOTA, SPACE_QUOTA, REMAINING_SPACE_QUOTA, PATHNAME
-q	Управляет тем, какие столбцы содержит вывод: QUOTA, REMAINING_QUOTA, SPACE_QUOTA, REMAINING_SPACE_QUOTA, DIR_COUNT, FILE_COUNT, CONTENT_SIZE, PATHNAME
-t	Список возможных параметров, которые могут быть использованы: `all`, `ram_disk`, `ssd`, `disk`, `archive`
-h	Показывает размер в удобном для чтения формате
-v	Отображает строку заголовка
-x	Исключает моментальные снимки из расчета результата. Без опции `-x` (по умолчанию) результат всегда вычисляется по всем INodes, включая все снимки под заданным путем. Параметр `-x` игнорируется, если задан параметр `-u` или `-q`
-e	Показывает политику erasure coding для каждого файла
-e	Управляет тем, какие столбцы содержит вывод: DIR_COUNT, FILE_COUNT, CONTENT_SIZE, ERASURECODING_POLICY, PATHNAME
-ERASURECODING_POLICY	Имя политики для файла. Если для этого файла установлена политика erasure coding, то будет возвращено имя политики. Если политика erasure coding не установлена, будет возвращено значение `Replicated`, что означает, что используется стратегия хранения с репликацией

Пример:

```
$ hadoop fs -count hdfs://nn1.example.com/file1 hdfs://nn2.example.com/file2
$ hadoop fs -count -q hdfs://nn1.example.com/file1
$ hadoop fs -count -q -h hdfs://nn1.example.com/file1
$ hadoop fs -count -q -h -v hdfs://nn1.example.com/file1
$ hadoop fs -count -u hdfs://nn1.example.com/file1
$ hadoop fs -count -u -h hdfs://nn1.example.com/file1
$ hadoop fs -count -u -h -v hdfs://nn1.example.com/file1
$ hadoop fs -count -e hdfs://nn1.example.com/file1
```

cp

Сергей Остапов

Назад к содержанию

Копирует файлы из источника в место назначения.
Команда позволяет использовать несколько источников, в этом случае местом назначения должен быть каталог.

Расширенные атрибуты пространства имен raw.* сохраняются, если файловые системы источника и назначения поддерживают их (только HDFS), и все пути к источнику и месту назначения находятся в иерархии */.reserved/raw*.
Определение того, сохраняются ли `xattrs` пространства имен raw.*, не зависит от флага `-p` (preserve).
Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -cp [-f] [-p | -p[topax]] URI [URI ...] <dest>
```

Аргументы

-f	Перезаписывает место назначения, если оно уже существует
-p	Сохраняет атрибуты файлов [topx] (временные метки, право собственности, разрешение, ACL, XAttr). Если `-p` передается без аргументов, то сохраняются метки времени, права собственности, разрешения. Если передается `-pa`, то сохраняется и разрешение, поскольку ACL является супермножеством разрешения. Определение того, сохраняются ли расширенные атрибуты необработанных пространств имен, не зависит от флага `-p`

Пример:

```
$ hadoop fs -cp /user/hadoop/file1 /user/hadoop/file2
$ hadoop fs -cp /user/hadoop/file1 /user/hadoop/file2 /user/hadoop/dir
```

createSnapshot

Сергей Остапов

Назад к содержанию

Создаёт моментальный снимок snapshottable-каталога.
Пользователь должен быть владельцем snapshottable-каталога. Более подробную информацию о работе с моментальными снимками можно получить в статье Снепшоты в HDFS.

Использование показано ниже:

```
$ hdfs dfs -createSnapshot <path> [<snapshotName>]
```

Аргументы

<path>	Указывает путь к snapshottable-каталогу
[<snapshotName>]	Указывает имя моментального снимка. Является необязательным аргументом. Если это значение не указано, то имя по умолчанию генерируется с использованием временной метки в формате `'s’yyyMMdd-HHmmss.SSS`, например, `s20130412-151029.033`

deleteSnapshot

Сергей Остапов

Назад к содержанию

Удаляет снимок из snapshottable-каталога.
Пользователь должен быть владельцем snapshottable-каталога. Более подробную информацию о работе с моментальными снимками можно получить в статье Снепшоты в HDFS.

Использование показано ниже:

```
$ hdfs dfs -deleteSnapshot <path> <snapshotName>
```

Аргументы

<path>	Путь к snapshottable-каталогу
<snapshotName>	Имя моментального снимка

df

Сергей Остапов

Назад к содержанию

Отображает свободное пространство.

Использование показано ниже:

```
$ hadoop fs -df [-h] URI [URI ...]
```

Аргументы

-h	Форматирует размеры файлов в удобный для чтения вид (например, `64.0m` вместо `67108864`)

Пример:

```
$ hadoop fs -df /user/hadoop/dir1
```

du

Сергей Остапов

Назад к содержанию

Отображает размеры файлов и каталогов, содержащихся в данном каталоге, или длину файла, если это просто файл.

Команда `du` возвращает три столбца со следующим форматом:

```
size disk_space_consumed_with_all_replicas full_path_name
```

Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -du [-s] [-h] [-v] [-x] URI [URI ...]
```

Аргументы

-s	Возвращает суммарную длину файлов, а не отдельных файлов. Без флага `-s` расчет производится путем перехода на 1 уровень вглубь от заданного пути
-h	Форматирует размеры файлов в удобный для чтения вид (например, `64.0m` вместо `67108864`)
-v	Отображает названия столбцов в качестве строки заголовка
-x	Исключает моментальные снимки из расчета результата. Без опции `-x` (по умолчанию) результат всегда вычисляется по всем INodes, включая все снепшоты под заданным путем

Пример:

```
$ hadoop fs -du /user/hadoop/dir1 /user/hadoop/file1 hdfs://nn.example.com/user/hadoop/dir1
```

dus

Сергей Остапов

Назад к содержанию

Отображает сводку длины файлов.

Использование показано ниже:

```
$ hadoop fs -dus <args>
```

ПРИМЕЧАНИЕ

Эта команда устарела. Вместо нее используется команда `hadoop fs -du -s`.

expunge

Сергей Остапов

Назад к содержанию

Перманентно удаляет файлы контрольных точек старше порога хранения из каталога корзины и создает новую контрольную точку.

Когда создается контрольная точка, недавно удаленные файлы в корзине перемещаются в контрольную точку.
Файлы в контрольных точках старше `fs.trash.interval` будут удалены навсегда при следующем вызове команды `-expunge`.

Если файловая система поддерживает эту функцию, пользователи могут настроить периодическое создание и удаление контрольных точек с помощью параметра `fs.trash.checkpoint.interval` (в файле *core-site.xml*).
Это значение должно быть меньше или равно `fs.trash.interval`.

Если передан аргумент `-immediate`, все файлы в корзине для текущего пользователя удаляются немедленно, игнорируя настройку `fs.trash.interval`.

Если передан аргумент `-fs`, будет удалена файловая система, а не файловая система по умолчанию, и будет создана контрольная точка.

Использование показано ниже:

```
$ hadoop fs -expunge [-immediate] [-fs <path>]
```

Пример:

```
$ hadoop fs -expunge --immediate -fs s3a://landsat-pds/
```

find

Сергей Остапов

Назад к содержанию

Находит все файлы, соответствующие заданному выражению, и применяет к ним указанные действия.
Если путь не указан, то по умолчанию используется текущая рабочая директория.
Если выражение не указано, то по умолчанию используется `-print`.

Принимаются следующие первичные выражения:

- -name <pattern>;
- -iname <pattern>.

Возвращает `true`, если имя файла соответствует шаблону с использованием стандартной глобализации (globbing) файловой системы.
Если используется `-iname`, то сравнение не чувствительно к регистру.

Выражения `-print` и `-print0` всегда возвращают `true` и предназначены для вывода текущего пути в стандартный вывод.
Если используется выражение `-print0`, то добавляется ASCII-символ `NULL`.

Принимаются следующие операторы:

- expression -a expression;
- expression -and expression;
- expression expression.

Логический оператор `AND` используется для объединения двух выражений.
Возвращает `true`, если оба дочерних выражения возвращают `true`.
Используется при соединении двух выражений и поэтому не требует явного указания.
Второе выражение не будет применено, если первое не удалось вычислить.

Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -find <path> ... <expression> ...
```

Пример:

```
$ hadoop fs -find / -name test -print
```

get

Сергей Остапов

Назад к содержанию

Копирует файлы в локальную файловую систему. Файлы, не прошедшие проверку CRC, могут быть скопированы с помощью аргумента `-ignorecrc`. Файлы и CRC можно копировать с помощью аргумента `-crc`. Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -get [-ignorecrc] [-crc] [-p] [-f] <src> <localdst>
```

Аргументы

-p	Сохраняет время доступа и модификации, права собственности и сеансы (при условии, что разрешения могут быть распространены по файловой системе)
-f	Перезаписывает место назначения, если оно уже существует
-ignorecrc	Пропускает CRC-проверку для загружаемого файла (файлов)
-crc	Записывает контрольные суммы CRC для загруженных файлов

Пример:

```
$ hadoop fs -get /user/hadoop/file localfile
$ hadoop fs -get hdfs://nn.example.com/user/hadoop/file localfile
```

getfacl

Сергей Остапов

Назад к содержанию

Отображает списки контроля доступа (ACL) файлов и каталогов.
Если каталог имеет ACL по умолчанию, то `getfacl` также отображает ACL по умолчанию. Возвращает `0` при успешной попытке и ненулевое значение при ошибке.

Использование показано ниже:

```
$ hadoop fs -getfacl [-R] <path>
```

Аргументы

-R	Рекурсивно перечисляет ACL всех файлов и каталогов
<path>	Файл или каталог для вывода списка

Пример:

```
$ hadoop fs -getfacl /file
$ hadoop fs -getfacl -R /dir
```

getfattr

Сергей Остапов

Назад к содержанию

Отображает имена и значения расширенных атрибутов (если они есть) для файла или каталога.

Возвращает `0` при успешной попытке и ненулевое значение при ошибке.

Использование показано ниже:

```
$ hadoop fs -getfattr [-R] -n name | -d [-e en] <path>
```

Пример:

```
$ hadoop fs -getfattr -d /file
$ hadoop fs -getfattr -R -n user.myAttr /dir
```

Аргументы

-R	Отображает список атрибутов рекурсивно для всех файлов и каталогов
-n	Выгружает значения именованного расширенного атрибута
-d	Выгружает все расширенные значения атрибутов, связанных с именем пути
-e encoding	Кодирует значения после их получения. Возможные кодировки: `text`, `hex` и `base64`. Значения, закодированные как текстовые строки, заключаются в двойные кавычки ("), а значения, закодированные как шестнадцатеричные и `base64`, имеют префиксы `0x` и `0s`, соответственно
path	Путь к файлу или каталогу

getmerge

Сергей Остапов

Назад к содержанию

Принимает исходный каталог и целевой файл в качестве входных данных и объединяет файлы в локальный целевой файл. Опционально можно установить флаг `-nl`, чтобы включить добавление символа новой строки (LF) в конце каждого файла. Флаг `-skip-empty-file` можно использовать, чтобы избежать ненужных символов новой строки в случае пустых файлов. Возвращает `0` при успешной попытке и ненулевое значение при ошибке.

Использование показано ниже:

```
$ hadoop fs -getmerge [-nl] <src> <localdst>
```

Пример:

```
$ hadoop fs -getmerge -nl /src /opt/output.txt
$ hadoop fs -getmerge -nl /src/file1.txt /src/file2.txt /output.txt
```

head

Сергей Остапов

Назад к содержанию

Выводит первый килобайт файла в stdout. Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -head URI
```

Пример:

```
$ hadoop fs -head pathname
```

help

Сергей Остапов

Назад к содержанию

Возвращает справочную информацию.

Использование показано ниже:

```
$ hadoop fs -help
```

Аргументы

-help	Отобразить справочную информацию

ls

Сергей Остапов

Назад к содержанию

Для файлов `ls` возвращает статистику по файлу в следующем формате:

```
permissions number_of_replicas userid groupid filesize modification_date modification_time filename
```

Для каталогов `ls` возвращает список прямых дочерних элементов, как в Unix.
Каталог перечисляется как:

```
permissions userid groupid modification_date modification_time dirname
```

По умолчанию файлы в каталоге упорядочиваются по имени файла.
Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -ls [-C] [-d] [-h] [-q] [-R] [-t] [-S] [-r] [-u] [-e] <args>
```

Аргументы

-C	Отображает только пути к файлам и каталогам
-d	Перечисляет каталоги как обычные файлы
-h	Форматирует размеры файлов в удобный для чтения вид (например, `64.0m` вместо `67108864`)
-q	Выводит `?` вместо непечатаемых символов
-R	Рекурсивно перечисляет найденные подкаталоги
-t	Сортирует вывод по времени модификации (сначала самые последние)
-S	Сортирует вывод по размеру файла
-r	Изменяет порядок сортировки
-u	Использует время доступа для отображения и сортировки, а не время модификации
-e	Отображает политику erasure coding только для файлов и каталогов

Пример:

```
$ hadoop fs -ls /user/hadoop/file1
$ hadoop fs -ls -e /ecdir
```

lsr

Сергей Остапов

Назад к содержанию

Рекурсивная версия команды ls.

Использование показно ниже:

```
$ hadoop fs -lsr <args>
```

ПРИМЕЧАНИЕ

Эта команда устарела. Вместо нее используйте команду `hadoop fs -ls -R`.

mkdir

Сергей Остапов

Назад к содержанию

Создает директории по указанному пути. Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -put [-f] [-p] [-l] [-d] [-t <thread count>] [ - | <localsrc1> .. ]. <dst>
```

Аргументы

-p	Создает необходимые родительские директории подобно Unix-команде `mkdir -p`

Пример:

```
$ hadoop fs -mkdir /user/hadoop/dir1 /user/hadoop/dir2
$ hadoop fs -mkdir hdfs://nn1.example.com/user/hadoop/dir hdfs://nn2.example.com/user/hadoop/dir
```

moveFromLocal

Сергей Остапов

Назад к содержанию

Идентична команде put, за исключением того, что источник `localsrc` удаляется после копирования.

Использование показано ниже:

```
$ hadoop fs -moveFromLocal <localsrc> <dst>
```

moveToLocal

Сергей Остапов

Назад к содержанию

ПРИМЕЧАНИЕ

Это команда пока не реализована.

Перемещает файл или директорию из HDFS в локальную файловую систему.

mv

Сергей Остапов

Назад к содержанию

Перемещает файлы из источника в место назначения в файловой системе HDFS. Позволяет указать несколько источников, в этом случае местом назначения должен быть каталог. Перемещение файлов между разными файловыми системами не допускается.

Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -mv URI [URI ...] <dest>
```

Пример:

```
$ hadoop fs -mv /user/hadoop/file1 /user/hadoop/file2
$ hadoop fs -mv hdfs://nn.example.com/file1 hdfs://nn.example.com/file2 hdfs://nn.example.com/file3 hdfs://nn.example.com/dir1
```

put

Сергей Остапов

Назад к содержанию

Копирует один файл или несколько файлов из локальной файловой системы в файловую систему назначения.
Также читает входные данные из stdin и записывает в конечную файловую систему, если в качестве источника указано `-`.

Копирование завершится неудачно, если файл уже существует или если не указан флаг `-f`.
Возвращает `0` при успешной попытке и `-1` при ошибке.
Использование показано ниже:

```
$ hadoop fs -put [-f] [-p] [-l] [-d] [-t <thread count>] [ - | <localsrc1> .. ]. <dst>
```

Аргументы

-p	Сохраняет время доступа и модификации, право собственности и разрешения (при условии, что разрешения могут быть распространены по файловой системе)
-f	Перезаписывает место назначения, если оно уже существует
-l	Разрешает DataNode «лениво» сохранять файл на диск; заставляет использовать коэффициент репликации `1`. Этот флаг приведет к снижению долговечности. Будьте осторожны при использовании этого аргумента
-d	Пропускает создание временного файла с суффиксом `.COPYING`

Пример:

```
$ hadoop fs -put localfile /user/hadoop/hadoopfile
$ hadoop fs -put -f localfile1 localfile2 /user/hadoop/hadoopdir
$ hadoop fs -put -d localfile hdfs://nn.example.com/hadoop/hadoopfile
$ hadoop fs -put - hdfs://nn.example.com/hadoop/hadoopfile
```

renameSnapshot

Сергей Остапов

Назад к содержанию

Переименовывает моментальный снимок.
Пользователь должен быть владельцем snapshottable-каталога. Более подробную информацию о работе с моментальными снимками можно получить в статье HDFS Snapshots.

Использование показано ниже:

```
$ hdfs dfs -renameSnapshot <path> <oldName> <newName>
```

Аргументы

<path>	Указывает путь к snapshottable-каталогу
<oldName>	Указывает исходное имя моментального снимка
<newName>	Указывает новое имя моментального снимка

rm

Сергей Остапов

Назад к содержанию

Удаляет файлы, указанные в качестве аргументов.

Если функция trash включена, файловая система вместо этого перемещает файл в каталог *.Trash* (задается getTrashRoot).

Использование показано ниже:

```
$ hadoop fs -rm [-f] [-r
|-R] [-skipTrash] [-safely] URI [URI ...]
```

Аргументы

-f	Не выводит диагностическое сообщение и не изменяет статус выхода, чтобы отразить ошибку, если файл не существует
-R	Рекурсивно удаляет каталог и все содержимое в нём
-r	Эквивалент аргумента `-R`
-skipTrash	Игнорирует функцию trash (если включена) и немедленно удаляет указанный файл(ы). Это может быть полезно, когда необходимо удалить файлы из каталога с превышенной квотой
-safely	Требует подтверждения безопасности перед удалением каталога с общим количеством файлов, превышающим `hadoop.shell.delete.limit.num.files` (в *core-site.xml*, по умолчанию: `100`). Может использоваться вместе с `-skipTrash` для предотвращения случайного удаления больших каталогов. Возможна задержка при рекурсивном обходе большого каталога для подсчета количества удаляемых файлов перед подтверждением

Пример:

```
$ hadoop fs -rm hdfs://nn.example.com/file /user/hadoop/emptydir
```

rmdir

Сергей Остапов

Назад к содержанию

Удаляет каталог.

Использование показано ниже:

```
$ hadoop fs -rmdir [--ignore-fail-on-non-empty] URI [URI ...]
```

Аргументы

--ignore-fail-on-non-empty	При использовании подстановочных знаков игнорирует ошибки, если каталог все еще содержит файлы

Пример:

```
$ hadoop fs -rmdir /user/hadoop/emptydir
```

rmr

Сергей Остапов

Назад к содержанию

Рекурсивная версия команды rm.

Использование показано ниже:

```
$ hadoop fs -rmr [-skipTrash] URI [URI ...]
```

ПРИМЕЧАНИЕ

Данная команда устарела. Вместо нее можно использовать `hadoop fs -rm -r`.

setfacl

Сергей Остапов

Назад к содержанию

Устанавливает списки контроля доступа (ACL) к файлам и каталогам. Возвращает `0` при успешной попытке и ненулевое значение при ошибке.

Использование показано ниже:

```
$ hadoop fs -setfacl [-R] [-b |-k -m |-x <acl_spec> <path>] |[--set <acl_spec> <path>]
```

Аргументы

-b	Удаляет все, кроме базовых записей ACL. Записи для пользователя, группы и других сохраняются для совместимости с битами разрешений
-k	Удаляет стандартный ACL
-R	Применяет операции ко всем файлам и каталогам рекурсивно
-m	Изменяет ACL. В ACL добавляются новые записи, а существующие записи сохраняются
-x	Удаляет указанные записи ACL. Другие записи ACL сохраняются
--set	Полностью заменяет ACL, отбрасывая все существующие записи. Для совместимости с битами разрешений `acl_spec` должен включать записи для пользователя, группы и других. Если `acl_spec` содержит только записи доступа, то существующие записи по умолчанию сохраняются. Если `acl_spec` содержит только записи по умолчанию, то существующие записи доступа сохраняются. Если `acl_spec` содержит и записи доступа, и записи по умолчанию, то заменяются обе записи
acl_spec	Список записей ACL, разделенных запятыми
path	Файл или каталог для изменения

Пример:

```
$ hadoop fs -setfacl -m user:hadoop:rw- /file
$ hadoop fs -setfacl -x user:hadoop /file
$ hadoop fs -setfacl -b /file
$ hadoop fs -setfacl -k /dir
$ hadoop fs -setfacl --set user::rw-,user:hadoop:rw-,group::r--,other::r-- /file
$ hadoop fs -setfacl -R -m user:hadoop:r-x /dir
$ hadoop fs -setfacl -m default:user:hadoop:r-x /di
```

setfattr

Сергей Остапов

Назад к содержанию

Устанавливает имя и значение расширенного атрибута для файла или каталога. Возвращает `0` при успешной попытке и ненулевое значение при ошибке.

Использование показано ниже:

```
$ hadoop fs -setfattr -n name [-v value] | -x name <path>
```

Аргументы

-n	Указывает имя расширенного атрибута
-v	Задаёт значение расширенного атрибута. Существует три различных способа кодирования значения. Если аргумент заключен в двойные кавычки, то значением будет строка внутри кавычек. Если аргумент имеет префикс `0x` или `0X`, то он воспринимается как шестнадцатеричное число. Если аргумент начинается с `0s` или `0S`, то он принимается в кодировке base64
-x	Удаляет расширенный атрибут
<path>	Указывает файл или каталог

Пример:

```
$ hadoop fs -setfattr -n user.myAttr -v myValue /file
$ hadoop fs -setfattr -n user.noValue /file
$ hadoop fs -setfattr -x user.myAttr /file
```

setrep

Сергей Остапов

Назад к содержанию

Изменяет коэффициент репликации файла. Если путь указывает на каталог, то команда рекурсивно изменяет коэффициент репликации всех файлов в данном каталоге. Файлы EC будут проигнорированы при выполнении этой команды. Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -setrep [-R] [-w] <numReplicas> <path>
```

Аргументы

-w	Активирует ожидание завершения репликации. Потенциально это может занять длительное время
-R	Используется для обратной совместимости. Установка флага не имеет эффекта

Пример:

```
$ hadoop fs -setrep -w 3 /user/hadoop/dir1
```

stat

Сергей Остапов

Назад к содержанию

Выводит статистику о файле/каталоге по адресу `<path>` в указанном формате.
Формат принимает разрешения в восьмеричном (`%a`) и символьном (`%A`) виде, размер файла в байтах (`%b`), тип (`%F`), имя группы владельца (`%g`), имя (`%n`), размер блока (`%o`), репликацию (`%r`), имя пользователя владельца (`%u`), дату доступа (`%x`, `%X`) и дату модификации (`%y`, `%Y`).
`%x` и `%y` показывают даты UTC в формате `yyyy-MM-dd HH:mm:ss`, а `%X` и `%Y` показывают миллисекунды с 1 января 1970 UTC. Если формат не указан, по умолчанию используется `%y`.

Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -stat [format] <path> ...
```

Пример:

```
$ hadoop fs -stat "type:%F perm:%a %u:%g size:%b mtime:%y atime:%x name:%n" /file
```

tail

Сергей Остапов

Назад к содержанию

Выводит последний килобайт файла в stdout. При успешной попытке выводит значение `0`, а при ошибке — значение `-1`.

Использование показано ниже:

```
$ hadoop fs -tail [-f] URI
```

Аргументы

-f	Выводит добавленные данные по мере увеличения файла, так же, как и в Unix

Пример:

```
$ hadoop fs -tail pathname
```

test

Сергей Остапов

Назад к содержанию

Запускает тестирование файлов.

Использование показано ниже:

```
$ hadoop fs -test -[defswrz] URI
```

Аргументы

-d	Если путь является каталогом, возвращает `0`
-e	Если путь существует, возвращает `0`
-f	Если путь является файлом, возвращает `0`
-s	Если путь не пуст, возвращает `0`
-s	Если путь существует и разрешение на запись дано, возвращает `0`
-r	Если путь существует и разрешение на чтение предоставлено, возвращает `0`
-z	Если файл имеет нулевую длину, возвращает `0`

Пример:

```
$ hadoop fs -test -e filename
```

text

Сергей Остапов

Назад к содержанию

Принимает исходный файл и выводит его в текстовом формате. Допустимыми форматами являются ZIP и TextRecordInputStream.

Использование показано ниже:

```
$ hadoop fs -text <src>
```

touch

Сергей Остапов

Назад к содержанию

Обновляет время доступа и модификации файла с указанным URI до текущего времени.
Если файла не существует, то создается пустой файл по URI с текущим временем в качестве временной метки этого URI.

Формат временной метки следующий: `yyyyMMddHHmmss`. Где:

- `yyyy` — четырехзначный год (например, `2018`);
- `MM` — двухзначный месяц (например, `08` для августа);
- `dd` — двухзначный день (например, `25`);
- `HH` — двухзначный час (например, `12`);
- `mm` — двухзначная минута часа (например, `30`);
- `ss` — двухзначная секунда (например, `00`).

Таким образом, `20180825:123000` означает 25 августа 2018 года 12:30:00.

Возвращает `0` при успешной попытке и `-1` при ошибке.

Использование показано ниже:

```
$ hadoop fs -touch [-a] [-m] [-t TIMESTAMP] [-c] URI [URI ...]
```

Аргументы

-a	Изменяет только время доступа
-m	Изменяет только время модификации
-t	Указывает временную метку (в формате `yyyyMMdd:HHmmss`) вместо текущего времени
-c	Не создаёт файл, если он не существует

Пример:

```
$ hadoop fs -touch -m -t 20180809:230000 pathname
$ hadoop fs -touch -t 20180809:230000 pathname
$ hadoop fs -touch -a pathname
```

touchz

Сергей Остапов

Назад к содержанию

Создаёт файл нулевой длины.
Возвращает `0` при успешной попытке и `-1`, если файл ненулевой длины уже существует.

Использование показано ниже:

```
$ hadoop fs -touchz URI [URI ...]
```

Пример:

```
$ hadoop fs -touchz pathname
```

truncate

Сергей Остапов

Назад к содержанию

Обрезает все файлы, соответствующие указанному шаблону, до указанной длины.

Использование показано ниже:

```
$ hadoop fs -truncate [-w] <length> <paths>
```

Аргументы

-w	Активирует ожидание завершения восстановления блока. Без флага `-w` файл может оставаться незакрытым в течение некоторого времени, пока идет восстановление. В течение этого времени файл не может быть снова открыт для записи

Пример:

```
$ hadoop fs -truncate 55 /user/hadoop/file1 /user/hadoop/file2
$ hadoop fs -truncate -w 127 hdfs://nn1.example.com/user/hadoop/file1
```

usage

Сергей Остапов

Назад к содержанию

Выводит справочную информацию для отдельной команды.

Использование показано ниже:

```
$ hadoop fs -usage command
```

HDFS command cheatsheet

Сергей Остапов

Содержание

Назад к содержанию

Команды вывода списков

Команды чтения и записи

Команды скачивания и загрузки

Команды управления файлами

Команды прав владения и проверки

Команды файловой системы

Административные команды

Команды вывода списков

hdfs dfs -ls /	Список всех файлов/каталогов в корневом каталоге HDFS
hdfs dfs -ls -d /tmp	Выводит каталоги как обычные файлы. В данном случае, эта команда перечислит подробную информацию папки _tmp_
hdfs dfs -ls -h /hdfsDirectory	Форматирует размеры файлов в директории HDFS в удобный для чтения вид
hdfs dfs -ls -R /hdfsDirectory	Рекурсивно перечисляет все файлы и директории в директории HDFS
hdfs dfs -ls /hdfsDirectory/hdfsFile*	Выводит все файлы, соответствующие шаблону. В этом случае выводит все файлы в HDFS-каталоге _hdfsDirectory_, которые начинаются с _hdfsFile_

Команды чтения и записи

hdfs dfs -text /hdfsDirectory/hdfsFile	Эта команда принимает исходный файл и выводит файл в текстовом формате в stdout. Допустимыми форматами являются ZIP и TextRecordInputStream
hdfs dfs -cat /hdfsDirectory/hdfsFile	Выводит содержимое файла HDFS в stdout
hdfs dfs -appendToFile localfsFile /tmp/hdfsFile	Добавляет содержимое локального файла в файл HDFS

Команды скачивания и загрузки

hdfs dfs -put localfsFile /hdfsDirectory	Копирует файл из локальной файловой системы в каталог HDFS
hdfs dfs -put -f localfsFile /hdfsDirectory	Копирует файл из локальной файловой системы в каталог HDFS, а в случае, если локальный файл уже существует в указанном пути назначения, использование опции `-f` с командой `-put` перезапишет его
hdfs dfs -put -l localfsFile /hdfsDirectory	Копирует файл из локальной файловой системы в каталог HDFS. Разрешает DataNode «лениво» сохранить файл на диск
hdfs dfs -put -p localfsFile /hdfsDirectory	Копирует файл из локальной файловой системы в каталог HDFS. Передача аргумента `-p` сохраняет доступ, модификации, право собственности и режим
hdfs dfs -get hdfsFile /localfsDirectory	Копирует файл из каталога HDFS в каталог локальной файловой системы
hdfs dfs -get -p hdfsFile /localfsDirectory	Копирует файл из каталога HDFS в каталог локальной файловой системы. Передача аргумента `-p` сохраняет доступ, модификации, право собственности и режим
hdfs dfs -get /localfsDirectory/*.* /hdfsDirectory	Копирует все файлы, соответствующие шаблону, из каталога локальной файловой системы в каталог HDFS
hdfs dfs -copyFromLocal localfsFile /hdfsDirectory	Работает аналогично команде put, за исключением того, что исходными данными для копирования должен быть локальный файл
hdfs dfs -copyToLocal hdfsFile /localfsDirectory	Работает аналогично команде put, за исключением того, что конечными данными для копирования должен быть локальный файл
hdfs dfs -moveFromLocal localfsFile /hdfsDirectory	Работает аналогично команде put, за исключением того, что исходные данные удаляются после копирования

Команды управления файлами

hdfs dfs -cp /hdfsDirectory1/hdfsFile1 /hdfsDirectory2	Копирует файл из одного каталога HDFS в другой каталог HDFS
hdfs dfs -cp -p /hdfsDirectory1/hdfsFile1 /hdfsDirectory2	Копирует файл из одного каталога HDFS в другой каталог HDFS. Передача аргумента `-p` сохраняет доступ, модификации, право собственности и режим
hdfs dfs -cp -f /hdfsDirectory1/hdfsFile1 /hdfsDirectory2	Копирует файл из одного каталога HDFS в другой каталог HDFS. Передача `-f` перезаписывает файл назначения, если он уже существует
hdfs dfs -mv /hdfsDirectory1/hdfsFile1 /hdfsDirectory2	Перемещает файл из одного каталога HDFS в другой каталог HDFS
hdfs dfs -rm /hdfsDirectory/hdfsFile _hdfs dfs -rm -r /hdfsDirectory_ _hdfs dfs -rm -R /hdfsDirectory_ _hdfs dfs -rmr /hdfsDirectory_	Рекурсивно удаляет все подкаталоги и файлы из каталога HDFS
hdfs dfs -rm -skipTrash /hdfsDirectory	Параметр `-skipTrash` удаляет один или несколько указанных файлов из каталога HDFS, минуя trash
hdfs dfs -rm -f hdfsFile	Не выводит предупреждение при выполнении команды, если файл не существует
hdfs dfs -rmdir /hdfsDirectory	Удаляет каталог HDFS
hdfs dfs -mkdir /hdfsDirectory	Создает каталог в указанном месте HDFS
hdfs dfs -mkdir -f /hdfsDirectory1/hdfsDirectory2	Создает каталог в указанной директории HDFS. Команда не вернет ошибку, даже если каталог уже существует
hdfs dfs -touchz /hdfsDirectory/hdfsFile	Создает файл нулевой длины по указанному пути с текущим временем в качестве временной метки этого пути

Команды прав владения и проверки

hdfs dfs -checksum /hdfsDirectory/hdfsFile	Выводит информацию о контрольной сумме файла в stdout
hdfs dfs -chmod 755 /hdfsDirectory/hdfsFile	Изменяет права доступа к файлу
hdfs dfs -chmod -R 755 /hdfsDirectory	Рекурсивно изменяет права доступа к файлам
hdfs dfs -chown owner:group /hdfsDirectory	Изменяет владельца файла
hdfs dfs -chown -R owner:group /hdfsDirectory	Рекурсивно изменяет владельца файлов в каталоге
hdfs dfs -chgrp group /hdfsDirectory	Изменяет группы файлов
hdfs dfs -chgrp -R group /hdfsDirectory	Рекурсивно изменяет группы файлов

Команды файловой системы

hdfs dfs -df /	Показывает общий объем, свободное и используемое пространство файловой системы
hdfs dfs -df -h /	Показывает общий объем, свободное и используемое пространство файловой системы. Ключ `-h` форматирует размеры файлов в удобный для чтения вид
hdfs dfs -du /hdfsDirectory/hdfsFile	Показывает объем пространства (в байтах), используемый файлами, которые соответствуют указанному шаблону файла
hdfs dfs -du -s /hdfsDirectory/hdfsFile	Вместо размера каждого файла, соответствующего шаблону, отображает общий (суммарный) размер
hdfs dfs -du -h /hdfsDirectory/hdfsFile	Показывает объем пространства (в байтах), используемый файлами, которые соответствуют указанному шаблону файла. Форматирует размеры файлов в удобный для чтения вид

Административные команды

hdfs balancer -threshold 30	Запускает утилиту балансировки кластера. Пороговое значение по умолчанию перезаписывается
hadoop version	Выводит текущую версию Hadoop
hdfs fsck /	Проверяет состояние HDFS
hdfs dfsadmin -safemode leave	Отключает безопасный режим для NameNode
hdfs dfsadmin -refreshNodes	Заново зачитывает файлы _hosts_ и _exclude_, чтобы обновить список DataNodes, которым разрешено подключаться к NameNode, и тех, которые должны быть выведены из эксплуатации или вновь введены в эксплуатацию
hdfs namenode -format	Форматирует NameNode

Команды HBase shell

Константин Алпашкин

Содержание

Назад к содержанию

- General commands
- Data definition commands (DDL)
- Namespace commands
- Data manipulation commands (DML)
- Tools commands
- Replication commands
- Snapshots commands
- Configuration commands
- Quotas commands
- Security commands
- Procedures commands
- Visibility labels commands
- Rsgroup commands

В данной статье описаны основные команды, доступные в HBase shell.

General commands

Команды общего назначения, включающие отображение информации о текущем кластере и пользователе, получение справки и так далее.

Команда	Описание
status	Отображает статус HBase-кластера
table_help	Выводит справочную информацию для таких операций, как scan, put, get, disable, drop и так далее
whoami	Отображает подробную информацию о текущем пользователе
version	Отображает версию HBase
processlist	Отображает задачи Region Servers

Data definition commands (DDL)

Команды, которые используются для создания схем таблиц и работы с ними.

Команда	Описание
create	Создает новую таблицу
describe	Отображает схему таблицы
list	Отображает таблицы, соответствующие заданному регулярному выражению
exists	Проверяет существование таблицы
locate_region	Находит регион по заданному имени таблицы и ключу строки
list_regions	Возвращает все регионы для конкретной таблицы
alter	Позволяет добавлять/изменять/удалять семейства столбцов и изменять конфигурацию таблицы
alter_async	Выполняет то же, что и команда alter, но не ждет, пока все регионы получат изменения схемы
alter_status	Получает статус команды alter
disable	Делает таблицу недоступной
disable_all	Делает недоступными таблицы, соответствующие заданному регулярному выражению
is_disabled	Проверяет, недоступна ли таблица
enable	Делает доступной таблицу, которая в данный момент может быть отключена
enable_all	Делает доступными таблицы, соответствующие заданному регулярному выражению
is_enabled	Проверяет, доступна ли таблица
get_table	Возвращает данную таблицу как объект, которым может манипулировать пользователь
drop	Удаляет таблицу навсегда
drop_all	Удаляет все таблицы, соответствующие данному регулярному выражению
show_filters	Выводит все фильтры в HBase

Namespace commands

Команды, позволяющие работать с пространствами имен — логическими группами таблиц.

Команда	Описание
create_namespace	Создает новое пространство имен
describe_namespace	Возвращает описание пространства имен
list_namespace	Возвращает список пространств имен — все или соответствующие заданному регулярному выражению
list_namespace_tables	Возвращает таблицы, включенные в пространство имен
alter_namespace	Изменяет пространство имен
drop_namespace	Удаляет пространство имен

Data manipulation commands (DML)

Команды, которые полезны для работы с данными, хранящимися в таблицах HBase: чтение, запись, удаление и так далее.

Команда	Описание
put	Вставляет значение ячейки для указанных таблицы/строки/столбца и, при необходимости, временной метки
append	Добавляет постфикс к значению ячейки для указанных таблицы/строки/столбца
incr	Увеличивает значение ячейки для указанных таблицы/строки/столбца
get_counter	Возвращает значение счетчика ячейки для указанных таблицы/строки/столбца
get	Извлекает содержимое строки/ячейки
scan	Сканирует таблицу и возвращает все строки, соответствующие заданным спецификациям
count	Возвращает количество строк в таблице
delete	Удаляет значение ячейки для указанных таблицы/строки/столбца и, при необходимости, временной метки
deleteall	Удаляет все ячейки в указанной строке
truncate	Отключает, удаляет и воссоздает указанную таблицу
truncate_preserve	Отключает, удаляет и воссоздает указанную таблицу, сохраняя прежние границы региона
get_splits	Возвращает точки разделения для данной таблицы

Tools commands

Команды, предназначенные для экспертов. Они помогают выполнять операции с регионами, такие как: разделение (split), слияние, сжатие (compaction) и так далее. Используйте их осторожно.

Команда	Описание
assign	Назначает регион
unassign	Отменяет назначение региона
close_region	Команда устарела. Вместо этого используйте unassign
compact	Выполняет compaction-процедуру для определенных таблицы/региона/семейства столбцов
compact_rs	Сжимает все регионы в пределах определенного Region Server
major_compact	Выполняет major compaction для определенных таблицы/региона/семейства столбцов
compaction_state	Получает статус сжатия таблицы
splitormerge_enabled	Проверяет, разрешены ли операции split или merge
splitormerge_switch	Включает/отключает операции split или merge
merge_region	Объединяет два региона
split	Разделяет всю таблицу или указанный регион
move	Перемещает регион
flush	Сбрасывает все регионы для таблицы/Region Server или все регионы с указанным именем
wal_roll	Обновляет WAL-writer
list_deadservers	Возвращает список всех отключенных (dead) Region Servers
clear_deadservers	Очищает все/указанные отключенные Region Servers
clear_block_cache	Удаляет блоки указанной таблицы из BlockCache
clear_compaction_queues	Очищает очереди сжатия в Region Server
balance_switch	Включает/отключает балансировщик кластера
balancer_enabled	Проверяет состояние балансировщика
balancer	Запускает балансировщик кластера
catalogjanitor_switch	Включает/отключает CatalogJanitor
catalogjanitor_enabled	Проверяет состояние CatalogJanitor
catalogjanitor_run	Запускает CatalogJanitor
cleaner_chore_switch	Включает/отключает работу Cleaner chore
cleaner_chore_enabled	Проверяет состояние Cleaner chore
cleaner_chore_run	Запускает работу Cleaner chore
normalizer_switch	Включает/отключает Region Normalizer
normalizer_enabled	Проверяет состояние Region Normalizer
normalize	Запускает Region Normalizer
is_in_maintenance_mode	Проверяет, находится ли Мастер в режиме обслуживания
zk_dump	Выводит состояние дампа кластера HBase, как его видит ZooKeeper
trace	Запускает/останавливает/проверяет трассировку через HTrace

Replication commands

Команды, используемые для настройки репликации.

Команда	Описание
add_peer	Добавляет отношение репликации (peer)
append_peer_namespaces	Добавляет реплицируемые пространства имен к отношению репликации
append_peer_tableCFs	Добавляет реплицируемые таблицы/семейства столбцов к отношению репликации
set_peer_namespaces	Устанавливает все реплицируемые пространства имен для отношения репликации
set_peer_tableCFs	Устанавливает все реплицируемые таблицы/семейства столбцов для отношения репликации
set_peer_bandwidth	Устанавливает исходную пропускную способность для отношения (peer) репликации (для Region Server)
set_peer_replicate_all	Устанавливает флаг `REPLICATE_ALL` для отношения репликации
set_peer_exclude_namespaces	Устанавливает нереплицируемые пространства имен для отношения репликации
set_peer_exclude_tableCFs	Устанавливает нереплицируемые таблицы/семейства столбцов для отношения репликации
disable_table_replication	Отключает репликацию для всех семейств столбцов таблицы
enable_table_replication	Включает репликацию для всех семейств столбцов таблицы
disable_peer	Делает недоступным отношение репликации (peer)
enable_peer	Делает доступным ранее отключенное отношение репликации (peer)
get_peer_config	Перечисляет параметры конфигурации отношения репликации
update_peer_config	Обновляет параметры конфигурации отношения репликации
list_peer_configs	Перечисляет параметры конфигурации всех отношений репликации
list_peers	Возвращает список всех отношений репликации (peers)
list_replicated_tables	Возвращает список всех таблиц и семейств столбцов, реплицированных из кластера
show_peer_tableCFs	Выводит реплицируемые таблицы/семейства столбцов для отношения репликации
remove_peer_namespaces	Удаляет пространства имен из конфигурации отношения репликации
remove_peer_tableCFs	Удаляет таблицы/семейства столбцов из конфигурации отношения репликации
remove_peer	Отключает и удаляет отношение репликации

Snapshots commands

Команды, позволяющие работать со снепшотами таблицы.

Команда	Описание
snapshot	Делает снепшот указанной таблицы
list_snapshots	Возвращает список снепшотов — всех или соответствующих заданному регулярному выражению
list_table_snapshots	Возвращает снепшоты указанной таблицы
restore_snapshot	Восстанавливает указанный снепшот
clone_snapshot	Создает новую таблицу путем клонирования указанного снепшота
delete_snapshot	Удаляет снепшот с указанным именем
delete_all_snapshot	Удаляет все снепшоты с именем, соответствующим заданному регулярному выражению
delete_table_snapshots	Удаляет снепшоты указанной таблицы

Configuration commands

Команды, которые используются для перезагрузки параметров конфигурации без перезапуска кластера. Набор параметров конфигурации, которые в данный момент могут быть изменены командами этой группы, ограничен. Более детальная информация доступна в документации HBase.

Команда	Описание
update_all_config	Перезагружает подмножество параметров конфигурации на всех серверах кластера
update_config	Перезагружает подмножество параметров конфигурации на определенных серверах кластера

Quotas commands

Команды, которые используются для работы с квотами (quotas) — ресурсными и другими ограничениями, применяемыми к объектам базы данных.
Прежде чем использовать команды из этой группы, добавьте следующее свойство в *hbase-site.xml* и перезапустите свой кластер.

Конфигурация

```
<property>
   <name>hbase.quota.enabled</name>
   <value>true</value>
</property>
```

Команда	Описание
set_quota	Устанавливает новую квоту для определенного пользователя/таблицы/пространства имен
list_quotas	Возвращает квоты, добавленные в HBase — все или соответствующие заданному регулярному выражению
list_quota_table_sizes	Возвращает вычисленный размер каждой таблицы в кластере
list_quota_snapshots	Возвращает информацию о таблицах с определенными квотами по размеру
list_snapshot_sizes	Возвращает размер снепшотов, сделанных для таблиц с заданными квотами по размеру

Security commands

Команды, полезные для настройки безопасности.
Прежде чем использовать команды из этой группы, добавьте следующие свойства в *hbase-site.xml* и перезапустите свой кластер. Если свойства `hbase.coprocessor.master.classes` или `hbase.coprocessor.region.classes` уже имеют другие значения, используйте списки, разделенные запятыми, чтобы сохранить для них несколько значений.

Конфигурация

```
<property>
   <name>hbase.coprocessor.master.classes</name>
   <value>org.apache.hadoop.hbase.security.access.AccessController</value>
</property>
<property>
   <name>hbase.coprocessor.region.classes</name>
   <value>org.apache.hadoop.hbase.security.access.AccessController</value>
</property>
<property>
   <name>hbase.rpc.engine</name>
   <value>org.apache.hadoop.hbase.ipc.SecureRpcEngine</value>
</property>
<property>
   <name>hbase.security.authorization</name>
   <value>true</value>
</property>
```

Команда	Описание
grant	Предоставляет пользователю определенные права
list_security_capabilities	Возвращает поддерживаемые возможности безопасности
revoke	Отзывает права доступа пользователя
user_permission	Возвращает разрешения для указанного пользователя

Procedures commands

Команды, которые используются для работы с процедурами HBase.

Команда	Описание
list_procedures	Возвращает все процедуры в HBase
abort_procedure	Прерывает выполнение процедуры с указанным ID
list_locks	Возвращает все блокировки (locks) в HBase

Visibility labels commands

Команды, используемые для управления метками видимости (visibility labels), которые позволяют пользователям, связанным с данной меткой, читать или получать доступ к ячейкам с этой меткой.
Прежде чем использовать команды из этой группы, добавьте следующие свойства в *hbase-site.xml* и перезапустите свой кластер. Если свойства `hbase.coprocessor.master.classes` или `hbase.coprocessor.region.classes` уже имеют другие значения, используйте списки, разделенные запятыми, чтобы сохранить для них несколько значений.

Конфигурация

```
<property>
   <name>hbase.coprocessor.master.classes</name>
   <value>org.apache.hadoop.hbase.security.visibility.VisibilityController</value>
</property>
<property>
   <name>hbase.coprocessor.region.classes</name>
   <value>org.apache.hadoop.hbase.security.visibility.VisibilityController</value>
</property>
<property>
   <name>hbase.security.authorization</name>
   <value>true</value>
</property>
<property>
   <name>hfile.format.version</name>
   <value>3</value>
</property>
```

Команда	Описание
add_labels	Добавляет набор меток видимости
list_labels	Возвращает список доступных меток видимости
set_auths	Назначает набор меток видимости определенному пользователю или группе
get_auths	Получает метки видимости, назначенные определенному пользователю или группе
clear_auths	Очищает метки видимости, назначенные определенному пользователю или группе
set_visibility	Определяет выражение видимости для одной или нескольких существующих ячеек таблицы

Rsgroup commands

Команды, позволяющие управлять Region Server-группами — логическими объединениями Region Server. Каждая группа при необходимости может монтировать разные таблицы. Таким образом, каждая группа эквивалентна логическому подкластеру. Это помогает добиться изоляции ресурсов и снизить затраты на управление. Группа Region Server по умолчанию называется `default`.
Прежде чем использовать команды из этой группы, добавьте следующие свойства в *hbase-site.xml* и перезапустите свой кластер. Если свойства `hbase.coprocessor.master.classes` или `hbase.coprocessor.region.classes` уже имеют другие значения, используйте списки, разделенные запятыми, чтобы сохранить для них несколько значений.

Конфигурация

```
<property>
   <name>hbase.coprocessor.master.classes</name>
   <value>org.apache.hadoop.hbase.rsgroup.RSGroupAdminEndpoint</value>
</property>
<property>
   <name>hbase.master.loadbalancer.class</name>
   <value>org.apache.hadoop.hbase.rsgroup.RSGroupBasedLoadBalancer</value>
</property>
```

Команда	Описание
add_rsgroup	Создает новую группу Region Server
list_rsgroups	Возвращает все группы Region Server
move_servers_rsgroup	Назначает указанные серверы в указанную группу Region Server
move_tables_rsgroup	Назначает таблицы в указанную группу Region Server
move_servers_tables_rsgroup	Назначает указанные серверы и таблицы в указанную группу Region Server
move_namespaces_rsgroup	Назначает заданные пространства имен в указанную группу Region Server
move_servers_namespaces_rsgroup	Назначает указанные серверы и пространства имен в группу Region Server
balance_rsgroup	Выполняет балансировку группы Region Server
get_rsgroup	Возвращает информацию о группе Region Server
get_server_rsgroup	Получает имя группы, членом которой является данный Region Server
get_table_rsgroup	Получает имя группы, членом которой является данная таблица
remove_servers_rsgroup	Удаляет выведенные из эксплуатации серверы из группы Region Server
remove_rsgroup	Удаляет Region Server-группу

processlist

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Выводит активные задачи Region Servers. Может отображаться в разных режимах просмотра, в зависимости от входного параметра.

Использование

```
processlist ['<view_mode>']
```

Аргументы

Параметр	Описание
view_mode	Возможные значения: - `all` — возвращает все типы задач (описаны ниже); - `general` — возвращает не связанные с RPC задачи, такие как `compact`, `flush` и так далее; - `handler` — возвращает задачи обработчика RPC; - `rpc` — возвращает задачи обработчика RPC в состоянии `RUNNING`; - `operation` — возвращает задачи обработчика RPC в состоянии `RUNNING` и от клиента. Режим просмотра по умолчанию: `general`

Примеры

```
hbase(main):001:0> processlist 'general'
0 tasks as of: 2021-11-17 14:26:33
No general tasks currently running.
Took 0.5531 seconds
hbase(main):002:0> processlist 'rpc'
0 tasks as of: 2021-11-17 14:26:42
No rpc tasks currently running.
Took 0.0331 seconds
```

status

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
- Вызов без аргументов
- simple
- detailed
- replication

Определение

Выводит текущее состояние кластера. Может отображаться в разных режимах просмотра, в зависимости от входного параметра.

Использование

```
status ['<view_mode>']
```

Аргументы

Параметр	Описание
view_mode	Возможные значения: ▪ `summary` — возвращает информацию об активных и резервных master-серверах, Region Servers и средней загрузке кластера в виде одной строки; ▪ `simple` — возвращает дополнительную информацию о Region Servers, такую как: количество регионов, количество и размеры файлов хранилища и так далее; ▪ `detailed` — добавляет к выводу предыдущего режима просмотра записи системного журнала и информацию о переходах регионов; ▪ `replication` — отображает статус репликации. Режим просмотра по умолчанию: `summary`

Примеры

Вызов без аргументов

```
hbase(main):001:0> status
1 active master, 0 backup masters, 3 servers, 0 dead, 13.0000 average load
Took 0.3476 seconds
```

simple

```
hbase(main):003:0> status 'simple'
active master:  bds-adh-2.ru-central1.internal:16000 1637127966912
0 backup masters
3 live servers
    bds-adh-1.ru-central1.internal:16020 1637127969828
        requestsPerSecond=0.0, numberOfOnlineRegions=13, usedHeapMB=99, maxHeapMB=3905, numberOfStores=13, numberOfStorefiles=13,
storefileUncompressedSizeMB=0, storefileSizeMB=0, memstoreSizeMB=0, storefileIndexSizeKB=0, readRequestsCount=0,
filteredReadRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0, totalStaticBloomSizeKB=0,
totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, coprocessors=[GroupedAggregateRegionObserver, Indexer,
ScanRegionObserver, SequenceRegionObserver, ServerCachingEndpointImpl, UngroupedAggregateRegionObserver]
    bds-adh-2.ru-central1.internal:16020 1637127970177
        requestsPerSecond=0.0, numberOfOnlineRegions=14, usedHeapMB=54, maxHeapMB=3905, numberOfStores=15, numberOfStorefiles=15,
storefileUncompressedSizeMB=0, storefileSizeMB=0, memstoreSizeMB=0, storefileIndexSizeKB=2, readRequestsCount=3891,
filteredReadRequestsCount=85, writeRequestsCount=76, rootIndexSizeKB=2, totalStaticIndexSizeKB=1, totalStaticBloomSizeKB=0,
totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, coprocessors=[GroupedAggregateRegionObserver, Indexer,
MetaDataEndpointImpl, MultiRowMutationEndpoint, ScanRegionObserver, ServerCachingEndpointImpl, UngroupedAggregateRegionObserver]
    bds-adh-3.ru-central1.internal:16020 1637127969753
        requestsPerSecond=0.0, numberOfOnlineRegions=12, usedHeapMB=89, maxHeapMB=3905, numberOfStores=12, numberOfStorefiles=12,
storefileUncompressedSizeMB=0, storefileSizeMB=0, memstoreSizeMB=0, storefileIndexSizeKB=0, readRequestsCount=1,
filteredReadRequestsCount=59080, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0, totalStaticBloomSizeKB=0,
totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, coprocessors=[GroupedAggregateRegionObserver, Indexer,
MetaDataEndpointImpl, MetaDataRegionObserver, MultiRowMutationEndpoint, ScanRegionObserver, ServerCachingEndpointImpl,
UngroupedAggregateRegionObserver]
0 dead servers
Aggregate load: 0, regions: 39
Took 0.0264 seconds
```

detailed

```
hbase(main):004:0> status 'detailed'
version 2.0.2
0 regionsInTransition
active master:  bds-adh-2.ru-central1.internal:16000 1637127966912
0 backup masters
master coprocessors: []
3 live servers
    bds-adh-1.ru-central1.internal:16020 1637127969828
        requestsPerSecond=0.0, numberOfOnlineRegions=13, usedHeapMB=101, maxHeapMB=3905, numberOfStores=13,
numberOfStorefiles=13, storefileUncompressedSizeMB=0, storefileSizeMB=0, memstoreSizeMB=0, storefileIndexSizeKB=0,
readRequestsCount=0, filteredReadRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, coprocessors=
[GroupedAggregateRegionObserver, Indexer, ScanRegionObserver, SequenceRegionObserver, ServerCachingEndpointImpl,
UngroupedAggregateRegionObserver]
        "SYSTEM.LOG,\\x01\\x00\\x00\\x00,1630426768774.96683724537040b683d0a24eb1f3bef6."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x02\\x00\\x00\\x00,1630426768774.346859a56a3dd85348d1d283ef830fc5."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x05\\x00\\x00\\x00,1630426768774.d11d2b44a4501c82e9cf2acdd50aaaf7."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x0A\\x00\\x00\\x00,1630426768774.287246b94f0727e6995dec5e7a5f6693."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x0C\\x00\\x00\\x00,1630426768774.7682bbcec9a40fc7bca560169eeff228."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x0E\\x00\\x00\\x00,1630426768774.759b0d6c372b17a9efdbbcbd2319e3a8."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x16\\x00\\x00\\x00,1630426768774.c98eb78752cf0073077f0e32277800dd."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x18\\x00\\x00\\x00,1630426768774.6d8cf50500196ce5b88b78c4b89da8a5."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x19\\x00\\x00\\x00,1630426768774.4ac5876e153fc8bc2e3eed202d935f33."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x1A\\x00\\x00\\x00,1630426768774.3f972f73d5d6f996c656083ae391075c."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x1B\\x00\\x00\\x00,1630426768774.93488f87483790344655d02ad3e085b3."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.MUTEX,,1630426773323.c28426b0346e789ba03bf338d649ec3a."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=1637054058546,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=1.0
        "SYSTEM.SEQUENCE,,1630426764347.7addb995515e262ebae6c4a21be37082."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
    bds-adh-2.ru-central1.internal:16020 1637127970177
        requestsPerSecond=0.0, numberOfOnlineRegions=14, usedHeapMB=56, maxHeapMB=3905, numberOfStores=15, numberOfStorefiles=15,
storefileUncompressedSizeMB=0, storefileSizeMB=0, memstoreSizeMB=0, storefileIndexSizeKB=2, readRequestsCount=3891,
filteredReadRequestsCount=85, writeRequestsCount=76, rootIndexSizeKB=2, totalStaticIndexSizeKB=1, totalStaticBloomSizeKB=0,
totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, coprocessors=[GroupedAggregateRegionObserver, Indexer,
MetaDataEndpointImpl, MultiRowMutationEndpoint, ScanRegionObserver, ServerCachingEndpointImpl, UngroupedAggregateRegionObserver]
        "SYSTEM.FUNCTION,,1630426767460.49cd07b4f3c549a9f1dbba531e8135ad."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x03\\x00\\x00\\x00,1630426768774.67b1c68668d4c37f2322d471665503fc."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x04\\x00\\x00\\x00,1630426768774.2b75793a1d9cf7d9ce3255539c4beca7."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x0B\\x00\\x00\\x00,1630426768774.811964f87fb0a84a7234d7a8d972eece."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x0D\\x00\\x00\\x00,1630426768774.14d9edf8f4ce99c1597595e133893c37."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x11\\x00\\x00\\x00,1630426768774.0a16d0cde0d9b68e9cfe82042c585770."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x14\\x00\\x00\\x00,1630426768774.78a990188fd99794c9553208429591e5."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x15\\x00\\x00\\x00,1630426768774.620fb8312d0ada2eaf1c84a1823ae001."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x17\\x00\\x00\\x00,1630426768774.c4c90bde3c271f7c1066999dd7e7d0c1."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x1C\\x00\\x00\\x00,1630426768774.f9fb782c652503128e76bb53f95cdf35."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,,1630426768774.664c1b894adf58a3c8b0961936da6e2b."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.STATS,,1630426765916.50e9d7b7b2f75034edffedc1ab3d4be7."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "hbase:meta,,1"
            numberOfStores=2, numberOfStorefiles=2, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=1637047655761,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=3887, writeRequestsCount=76, rootIndexSizeKB=2, totalStaticIndexSizeKB=1,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=1426,
dataLocality=1.0
        "hbase:namespace,,1630426726207.d4a22db7f8447b04e8be8ab0d6a1e8b3."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=4, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=1.0
    bds-adh-3.ru-central1.internal:16020 1637127969753
        requestsPerSecond=0.0, numberOfOnlineRegions=12, usedHeapMB=91, maxHeapMB=3905, numberOfStores=12, numberOfStorefiles=12,
storefileUncompressedSizeMB=0, storefileSizeMB=0, memstoreSizeMB=0, storefileIndexSizeKB=0, readRequestsCount=1,
filteredReadRequestsCount=59220, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0, totalStaticBloomSizeKB=0,
totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, coprocessors=[GroupedAggregateRegionObserver, Indexer,
MetaDataEndpointImpl, MetaDataRegionObserver, MultiRowMutationEndpoint, ScanRegionObserver, ServerCachingEndpointImpl,
UngroupedAggregateRegionObserver]
        "SYSTEM.CATALOG,,1630426759159.de2be860631b95097d3c710319e15053."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=1635260340542,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=1, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=1.0
        "SYSTEM.LOG,\\x06\\x00\\x00\\x00,1630426768774.99209a63b8aad46633c5bc197653874e."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x07\\x00\\x00\\x00,1630426768774.103e18787444a6d5d8b8760f457986f7."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x08\\x00\\x00\\x00,1630426768774.870b23dd7bf02f5581110acfe063f4f7."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x09\\x00\\x00\\x00,1630426768774.c4384d2f8fe4c35546e4ae770fddb476."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x0F\\x00\\x00\\x00,1630426768774.36f4b9e518d3f6d8b4aec209170ff6d3."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x10\\x00\\x00\\x00,1630426768774.68e7e7c2e5e430b4c779d7ae335f0cee."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x12\\x00\\x00\\x00,1630426768774.be32639f50a14ed50b7f9b4b779482f7."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x13\\x00\\x00\\x00,1630426768774.eb5cebd439b59fda2d6217b33d1942b9."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x1D\\x00\\x00\\x00,1630426768774.f86eb1f1018a991ebf95a8857f6c1032."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x1E\\x00\\x00\\x00,1630426768774.5754c2c290bc5dbc93b653cc891a2f4c."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
        "SYSTEM.LOG,\\x1F\\x00\\x00\\x00,1630426768774.a3574b91b1ca17bf18dd17e90e62d198."
            numberOfStores=1, numberOfStorefiles=1, storefileUncompressedSizeMB=0, lastMajorCompactionTimestamp=0,
storefileSizeMB=0, memstoreSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1,
dataLocality=0.0
0 dead servers
Took 0.0255 seconds
=> #<Java::JavaUtil::Collections::UnmodifiableRandomAccessList:0x3de88f64>
```

replication

```
hbase(main):005:0> status 'replication'
version 2.0.2
3 live servers
    bds-adh-1.ru-central1.internal:
       SOURCE:
       SINK  : AgeOfLastAppliedOp=0, TimeStampsOfLastAppliedOp=Wed Nov 17 05:46:12 UTC 2021
    bds-adh-2.ru-central1.internal:
       SOURCE:
       SINK  : AgeOfLastAppliedOp=0, TimeStampsOfLastAppliedOp=Wed Nov 17 05:46:12 UTC 2021
    bds-adh-3.ru-central1.internal:
       SOURCE:
       SINK  : AgeOfLastAppliedOp=0, TimeStampsOfLastAppliedOp=Wed Nov 17 05:46:12 UTC 2021
Took 0.0139 seconds
=> #<Java::JavaUtil::Collections::UnmodifiableSet:0x290c266c>
```

table_help

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит подсказку для команд, связанных с таблицами: create, put, get, scan, etc.

Использование

```
table_help
```

Примеры

```
hbase(main):014:0> table_help
Help for table-reference commands.

You can either create a table via 'create' and then manipulate the table via commands like 'put', 'get', etc.
See the standard help information for how to use each of these commands.

However, as of 0.96, you can also get a reference to a table, on which you can invoke commands.
For instance, you can get create a table and keep around a reference to it via:

   hbase> t = create 't', 'cf'

Or, if you have already created the table, you can get a reference to it:

   hbase> t = get_table 't'

You can do things like call 'put' on the table:

  hbase> t.put 'r', 'cf:q', 'v'

which puts a row 'r' with column family 'cf', qualifier 'q' and value 'v' into table t.

To read the data out, you can scan the table:

  hbase> t.scan

which will read all the rows in table 't'.

Essentially, any command that takes a table name can also be done via table reference.
Other commands include things like: get, delete, deleteall,
get_all_columns, get_counter, count, incr. These functions, along with
the standard JRuby object methods are also available via tab completion.

For more information on how to use each of these commands, you can also just type:

   hbase> t.help 'scan'

which will output more information on how to use that command.

You can also do general admin actions directly on a table; things like enable, disable,
flush and drop just by typing:

   hbase> t.enable
   hbase> t.flush
   hbase> t.disable
   hbase> t.drop

Note that after dropping a table, your reference to it becomes useless and further usage
is undefined (and not recommended).
Took 0.0002 seconds
```

version

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит текущую версию экземпляра HBase.

Использование

```
version
```

Примеры

```
hbase(main):019:0> version
2.0.2, rUnknown, Thu Apr 15 20:20:26 UTC 2021
Took 0.0002 seconds
```

whoami

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Выводит текущего пользователя HBase.

Использование

```
whoami
```

Примеры

```
hbase(main):021:0> whoami
dasha (auth:SIMPLE)
    groups: dasha
Took 0.0090 seconds
```

alter

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Изменение семейства столбцов
  - Добавление нового семейства столбцов
  - Удаление семейства столбцов
  - Установка параметра на уровне таблицы
  - Сброс параметра на уровне таблицы
  - Настройка параметра конфигурации
  - Сброс параметра конфигурации

Определение

Позволяет добавлять, изменять или удалять семейства столбцов и изменять параметры конфигурации таблицы. Семейства столбцов определяются так же, как и в случае команды create. Каждая спецификация семейства столбцов может быть либо простой строкой, либо словарем с обязательным атрибутом `NAME`.

Использование

- Добавление или изменение семейства столбцов:

```
alter '[<namespace_name>:]<table_name>',
      '<column_family>' | {NAME => '<column_family>'[, <cf_attribute_name> => <cf_attribute_value>,...]}[,...]
```

- Удаление семейства столбцов:
  - Способ 1:

```
alter '[<namespace_name>:]<table_name>',
      NAME => '<column_family>',
      METHOD => 'delete'
```

  - Способ 2:

```
alter '[<namespace_name>:]<table_name>',
      'delete' => '<column_family>'
```

- Установка параметров на уровне таблицы:

```
alter '[<namespace_name>:]<table_name>',
      <table_attribute_name> => <table_attribute_value>[,...]
```

- Сброс параметров на уровне таблицы:

```
alter '[<namespace_name>:]<table_name>',
      METHOD => 'table_att_unset',
      NAME => '<table_attribute_name>'
```

- Настройка параметров конфигурации:

```
alter '[<namespace_name>:]<table_name>',
      CONFIGURATION => {'<conf_attribute_name>' => <conf_attribute_value>[,...]}
```

- Сброс параметров конфигурации:

```
alter '[<namespace_name>:]<table_name>',
      METHOD => 'table_conf_unset',
      NAME => '<conf_attribute_name>'
```

РЕКОМЕНДАЦИЯ

Одна команда может содержать несколько alter-выражений.

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_family	Имя семейства столбцов
cf_attribute_name	Имя атрибута семейства столбцов
cf_attribute_value	Значение указанного атрибута семейства столбцов
table_attribute_name	Имя атрибута на уровне таблицы
table_attribute_value	Значение указанного атрибута на уровне таблицы
conf_attribute_name	Имя атрибута конфигурации
conf_attribute_value	Значение указанного атрибута конфигурации

Примеры

Изменение семейства столбцов

```
hbase(main):001:0> describe 't1'
Table t1 is ENABLED
t1
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE
=> '0', BLOOMFILTER => 'ROW', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false',
PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
{NAME => 'cf2', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE
=> '0', BLOOMFILTER => 'ROW', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false',
PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
2 row(s)
Took 0.5144 seconds
hbase(main):002:0> alter 't1', {NAME => 'cf1', VERSIONS => 10}
Updating all regions with the new schema...
1/1 regions updated.
Done.
Took 2.5811 seconds
hbase(main):003:0> describe 't1'
Table t1 is ENABLED
t1
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', VERSIONS => '10', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE
=> '0', BLOOMFILTER => 'ROW', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false',
PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
{NAME => 'cf2', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE
=> '0', BLOOMFILTER => 'ROW', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false',
PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
2 row(s)
Took 0.0206 seconds
```

Добавление нового семейства столбцов

```
hbase(main):004:0> alter 't1', 'cf3'
Updating all regions with the new schema...
1/1 regions updated.
Done.
Took 2.1848 seconds
hbase(main):005:0> describe 't1'
Table t1 is ENABLED
t1
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', VERSIONS => '10', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE
=> '0', BLOOMFILTER => 'ROW', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false',
PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
{NAME => 'cf2', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536'}
{NAME => 'cf3', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE
=> '0', BLOOMFILTER => 'ROW', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false',
PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
3 row(s)
Took 0.0183 seconds
```

Удаление семейства столбцов

```
hbase(main):007:0> alter 't1', NAME => 'cf3', METHOD => 'delete'
Updating all regions with the new schema...
1/1 regions updated.
Done.
Took 2.5314 seconds
hbase(main):008:0> describe 't1'
Table t1 is ENABLED
t1
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', VERSIONS => '10', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE
=> '0', BLOOMFILTER => 'ROW', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false',
PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
{NAME => 'cf2', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE
=> '0', BLOOMFILTER => 'ROW', CACHE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false',
PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
2 row(s)
Took 0.0175 seconds
```

Установка параметра на уровне таблицы

```
hbase(main):001:0> alter 't1', MAX_FILESIZE => '134217728'
Updating all regions with the new schema...
1/1 regions updated.
Done.
Took 2.9316 seconds
```

Сброс параметра на уровне таблицы

```
hbase(main):002:0> alter 't1', METHOD => 'table_att_unset', NAME => 'MAX_FILESIZE'
Updating all regions with the new schema...
1/1 regions updated.
Done.
Took 2.3027 seconds
```

Настройка параметра конфигурации

```
hbase(main):003:0> alter 't1', CONFIGURATION => {'hbase.hregion.scan.loadColumnFamiliesOnDemand' => 'true'}
Updating all regions with the new schema...
1/1 regions updated.
Done.
Took 2.3146 seconds
```

Сброс параметра конфигурации

```
hbase(main):004:0> alter 't1', METHOD => 'table_conf_unset', NAME => 'hbase.hregion.scan.loadColumnFamiliesOnDemand'
Updating all regions with the new schema...
1/1 regions updated.
Done.
Took 2.4628 seconds
```

alter_async

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Работает схожим образом с командой alter, но не ждет, пока все регионы получат изменения схемы. Таким образом, в отличие от alter, вывод команды не содержит сообщение "Updating all regions with the new schema...".

Использование

- Добавление или изменение семейства столбцов:

```
alter_async '[<namespace_name>:]<table_name>',
            '<column_family>' | {NAME => '<column_family>'[, <cf_attribute_name> => <cf_attribute_value>,...]}[,...]
```

- Удаление семейства столбцов:
  - Способ 1:

```
alter_async '[<namespace_name>:]<table_name>',
            NAME => '<column_family>',
            METHOD => 'delete'
```

  - Способ 2:

```
alter_async '[<namespace_name>:]<table_name>',
            'delete' => '<column_family>'
```

- Установка параметров на уровне таблицы:

```
alter_async '[<namespace_name>:]<table_name>',
            <table_attribute_name> => <table_attribute_value>[,...]
```

- Сброс параметров на уровне таблицы:

```
alter_async '[<namespace_name>:]<table_name>',
            METHOD => 'table_att_unset',
            NAME => '<table_attribute_name>'
```

- Настройка параметров конфигурации:

```
alter_async '[<namespace_name>:]<table_name>',
            CONFIGURATION => {'<conf_attribute_name>' => <conf_attribute_value>[,...]}
```

- Сброс параметров конфигурации:

```
alter_async '[<namespace_name>:]<table_name>',
            METHOD => 'table_conf_unset',
            NAME => '<conf_attribute_name>'
```

РЕКОМЕНДАЦИЯ

Одна команда может содержать несколько alter-выражений.

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_family	Имя семейства столбцов
cf_attribute_name	Имя атрибута семейства столбцов
cf_attribute_value	Значение указанного атрибута семейства столбцов
table_attribute_name	Имя атрибута на уровне таблицы
table_attribute_value	Значение указанного атрибута на уровне таблицы
conf_attribute_name	Имя атрибута конфигурации
conf_attribute_value	Значение указанного атрибута конфигурации

Примеры

```
hbase(main):006:0> alter_async 't1', 'cf3'
Took 1.2380 seconds
hbase(main):007:0> describe 't1'
Table t1 is ENABLED
t1
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', VERSIONS => '10', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'f
alse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
 IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536
'}
{NAME => 'cf2', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536'
}
{NAME => 'cf3', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536'
}
3 row(s)
Took 0.0181 seconds
```

alter_status

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Возвращает статус команды alter, указывающий количество регионов таблицы, получивших обновленную схему.

Использование

```
alter_status '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):013:0> alter_status 't1'
1/1 regions updated.
Done.
Took 1.0178 seconds
```

create

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Создание таблицы в пространстве имен по умолчанию без словарей
  - Использование нового пространства имен и словарей для семейств столбцов
  - Определение параметров на уровне таблицы после спецификаций семейств столбцов
  - Определение параметров конфигурации после спецификаций семейств столбцов

Определение

Создает новую таблицу. Требует имя таблицы и хотя бы одну спецификацию семейства столбцов. При желании вы можете определить параметры на уровне таблицы и параметры конфигурации. Каждая спецификация столбца может быть простой строкой, содержащей имя семейства столбцов, или словарем с использованием синтаксиса ключ/значение с фигурными скобками. Во втором случае указание атрибута `NAME` обязательно.

Использование

```
create '[<namespace_name>:]<table_name>',
       '<column_family>' | {NAME => '<column_family>'[,
                           <cf_attribute_name> => <cf_attribute_value>,...]
                           }[,...][,
       {<table_attribute_name> => <table_attribute_value>,[...]}][,
       CONFIGURATION => {'<conf_attribute_name>' => <conf_attribute_value>[,...]}]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_family	Имя семейства столбцов
cf_attribute_name	Имя атрибута семейства столбцов
cf_attribute_value	Значение указанного атрибута семейства столбцов
table_attribute_name	Имя атрибута на уровне таблицы
table_attribute_value	Значение указанного атрибута на уровне таблицы
conf_attribute_name	Имя атрибута конфигурации
conf_attribute_value	Значение указанного атрибута конфигурации

Примеры

Создание таблицы в пространстве имен по умолчанию без словарей

```
hbase(main):005:0> create 't1', 'cf1', 'cf2'
Created table t1
Took 1.2482 seconds
=> Hbase::Table - t1
```

Использование нового пространства имен и словарей для семейств столбцов

```
hbase(main):001:0> create 'ns1:t2', {NAME => 'cf1', TTL => 2592000}, {NAME => 'cf2', VERSIONS => 5}
Created table ns1:t2
Took 1.6133 seconds
=> Hbase::Table - ns1:t2
```

Определение параметров на уровне таблицы после спецификаций семейств столбцов

```
hbase(main):019:0> create 't3', 'cf1', {NUMREGIONS => 15, SPLITALGO => 'HexStringSplit'}
Created table t3
Took 1.2313 seconds
=> Hbase::Table - t3
```

Определение параметров конфигурации после спецификаций семейств столбцов

```
hbase(main):016:0> create 't4', 'cf1', CONFIGURATION => {'hbase.hregion.scan.loadColumnFamiliesOnDemand' => 'true'}
Created table t4
Took 1.3492 seconds
=> Hbase::Table - t4
```

describe

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Описание таблицы из пространства имен по умолчанию
  - Описание таблицы из другого пространства имен

Определение

Отображает схему заданной таблицы. В качестве альтернативы вы можете использовать сокращенную команду `desc`.

Использование

```
describe '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

Описание таблицы из пространства имен по умолчанию

```
hbase(main):001:0> describe 't1'
Table t1 is ENABLED
t1
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536'
}
{NAME => 'cf2', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536'
}
2 row(s)
Took 0.4877 seconds
```

Описание таблицы из другого пространства имен

```
hbase(main):002:0> describe 'ns1:t2'
Table ns1:t2 is ENABLED
ns1:t2
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => '2592000 SECONDS (30 DAYS)', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0',
BLOOMFILTER => 'ROW', CACHE_INDEX_ON_
WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION =>
'NONE', BLOCKCACHE => 'true', BL
OCKSIZE => '65536'}
{NAME => 'cf2', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536'
}
2 row(s)
Took 0.0163 seconds
```

disable

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Делает таблицу недоступной. В недоступных (disabled) таблицах нельзя выполнять операции с данными.

ВАЖНО

Выполнение команды `disable` обязательно перед выполнением команд drop, drop_all, restore_snapshot.

Использование

```
disable '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):002:0> disable 't1'
Took 1.1979 seconds
hbase(main):003:0> disable 'ns1:t2'
Took 0.7729 seconds
```

disable_all

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Делает недоступными все таблицы, соответствующие заданному регулярному выражению. Сначала находятся таблицы, соответствующие регулярному выражению, запрашивается подтверждение операции, а затем таблицы становятся недоступны (disabled).

Использование

```
disable_all '<table_regex>'
```

Аргументы

Параметр	Описание
table_regex	Регулярное выражение для поиска таблиц

Примеры

```
hbase(main):005:0> disable_all 't.'
t1
t3

Disable the above 2 tables (y/n)?
y
2 tables successfully disabled
Took 7.2381 seconds
```

drop

Константин Алпашкин

Содержание

Назад к содержанию

[Определение](#)

[Использование](#)

[Примеры](#)

Определение

Удаляет данную таблицу навсегда. Может применяться только к таблицам типа disabled.

Использование

```
drop '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):010:0> drop "ns1:t2"

ERROR: Table ns1:t2 is enabled. Disable it first.

Drop the named table. Table must first be disabled:
  hbase> drop 't1'
  hbase> drop 'ns1:t1'

Took 0.0351 seconds
hbase(main):011:0> disable "ns1:t2"
Took 0.8114 seconds
hbase(main):012:0> drop "ns1:t2"
Took 0.4729 seconds
```

drop_all

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Удаляет таблицы, соответствующие заданному регулярному выражению. Может применяться только к таблицам типа disabled. Сначала находятся таблицы, соответствующие регулярному выражению, запрашивается подтверждение операции, а затем таблицы удаляются безвозвратно.

Использование

```
drop_all '<table_regex>'
```

Аргументы

Параметр	Описание
table_regex	Регулярное выражение для поиска таблиц

Примеры

```
hbase(main):013:0> drop_all 't.'
t1
t3

Drop the above 2 tables (y/n)?
y
0 tables successfully dropped
2 tables not dropped due to an exception: t1,t3
Took 4.2576 seconds
hbase(main):014:0> disable_all 't.'
t1
t3

Disable the above 2 tables (y/n)?
y
2 tables successfully disabled
Took 4.4517 seconds
hbase(main):015:0> drop_all 't.'
t1
t3

Drop the above 2 tables (y/n)?
y
2 tables successfully dropped
Took 3.0036 seconds
```

enable

Константин Алпашкин

Содержание

Назад к содержанию

Использование

Примеры

Делает доступной таблицу. Операции с данными можно выполнять только с доступными (enabled) таблицами.

ВАЖНО

После выполнения команд drop, drop_all, restore_snapshot необходимо выполнить команду `enable`.

Использование

```
enable '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):010:0> enable 'ns1:t2'
Took 0.7863 seconds
```

enable_all

Константин Алпашкин

Содержание

Назад к содержанию

Использование

Примеры

Делает доступными все таблицы, соответствующие заданному регулярному выражению. Сначала находятся таблицы, соответствующие регулярному выражению, запрашивается подтверждение операции, а затем таблицы становятся доступными (enabled).

Использование

```
enable_all '<table_regex>'
```

Аргументы

Параметр	Описание
table_regex	Регулярное выражение для поиска таблиц

Примеры

```
hbase(main):012:0> enable_all 't.'
t1
t3

Enable the above 2 tables (y/n)?
y
2 tables successfully enabled
Took 10.9382 seconds
```

exists

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Проверка существующей таблицы
  - Проверка несуществующей таблицы

Определение

Проверяет, существует ли таблица.

Использование

```
exists '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

Проверка существующей таблицы

```
hbase(main):017:0> exists 't1'
Table t1 does exist
Took 0.0077 seconds
=> true
```

Проверка несуществующей таблицы

```
hbase(main):018:0> exists 'ns1:t1'
Table ns1:t1 does not exist
Took 0.0072 seconds
=> false
```

get_table

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Находит таблицу по заданному имени и возвращает ее как объект, доступный для манипуляций пользователя. Все последующие операции можно применять к этому объекту, не указывая имя таблицы.

Использование

```
<object_name> = get_table '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
object_name	Имя переменной, которая будет ссылаться на таблицу и позволит вызывать ее без указания ее имени
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):025:0> t1 = get_table 't1'
Took 0.0003 seconds
=> Hbase::Table - t1
hbase(main):027:0> t1.describe
Table t1 is ENABLED
t1
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536'
}
{NAME => 'cf2', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'fa
lse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536'
}
2 row(s)
Took 0.0223 seconds
hbase(main):030:0> t1.put 'r1', 'cf1:c1', 'v2'
Took 0.0051 seconds
```

is_enabled

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает `true`, если таблица в настоящее время доступна, в противном случае возвращает `false`.

Использование

```
is_enabled '[namespace_name:]table_name'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):013:0> is_enabled 't3'
true
Took 0.0146 seconds
=> true
```

is_disabled

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает `true`, если таблица в данный момент недоступна, иначе возвращает `false`.

Использование

```
is_disabled '[namespace_name:]table_name'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):007:0> is_disabled 't3'
true
Took 0.0102 seconds
=> 1
```

list

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
- Получение списка всех таблиц
- Получение списка таблиц, соответствующих регулярному выражению

Определение

Возвращает все таблицы в HBase. Необязательный параметр регулярного выражения можно использовать для фильтрации вывода.

Использование

```
list ['[<namespace_name>:]<table_name>' | 'table_regex']
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
table_regex	Регулярное выражение для поиска таблиц

Примеры

Получение списка всех таблиц

```
hbase(main):010:0> list
TABLE
SYSTEM.CATALOG
SYSTEM.FUNCTION
SYSTEM.LOG
SYSTEM.MUTEX
SYSTEM.SEQUENCE
SYSTEM.STATS
ns1:t2
t1
t3
9 row(s)
Took 0.0044 seconds
=> ["SYSTEM.CATALOG", "SYSTEM.FUNCTION", "SYSTEM.LOG", "SYSTEM.MUTEX", "SYSTEM.SEQUENCE", "SYSTEM.STATS", "ns1:t2", "t1", "t3"]
```

Получение списка таблиц, соответствующих регулярному выражению

```
hbase(main):015:0> list 't.'
TABLE
t1
t3
2 row(s)
Took 0.0034 seconds
=> ["t1", "t3"]
```

list_regions

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Получение списка регионов без фильтров
  - Получение списка регионов для определенного Region Server
  - Ограничение выходных полей

Определение

Отображает все регионы для указанной таблицы.

Использование

```
list_regions '[namespace_name:]<table_name>'[,
             {<attr_name> => '<attr_value>'[,...]}[,
             <output_fields[]>]]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
attr_name	Опциональный параметр, используемый для фильтрации результатов. Возможные значения: `'SERVER_NAME'` (имя Region Server), `'LOCALITY_THRESHOLD'` (расположение региона)
attr_value	Желаемое значение параметра `attr_name`
output_fields	Необязательный массив, содержащий одно или несколько полей, которые должны отображаться в выходной информации. Возможные значения: `'SERVER_NAME'`, `'REGION_NAME'`, `'START_KEY'`, `'END_KEY'`, `'SIZE'`, `'REQ'`, `'LOCALITY'`. Возможно использование нескольких значений. Значения не чувствительны к регистру. По умолчанию отображаются все поля (описание доступно ниже)

Выходные параметры

Параметр	Описание
SERVER_NAME	Имя Region Server
REGION_NAME	Полное имя региона
START_KEY	Начальный ключ строки региона
END_KEY	Конечный ключ строки региона
SIZE	Размер региона (в МБ)
REQ	Количество запросов к региону
LOCALITY	Расположение региона (`1` в случае сброса данных в HFiles)

Примеры

Получение списка регионов без фильтров

```
hbase(main):007:0> list_regions 't1'
                                        SERVER_NAME |                                         REGION_NAME |  START_KEY |
END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | --------------------------------------------------- | ---------- | --------
-- | ----- | ----- | ---------- |
 bds-adh-2.ru-central1.internal,16020,1637214343541 | t1,,1637234844003.562aabcac6ff0688b31260c44c9863c8. |            |
|     0 |     0 |        1.0 |
 1 rows
Took 0.5142 seconds
```

Получение списка регионов для определенного Region Server

```
hbase(main):010:0> list_regions 't1', {SERVER_NAME => 'bds-adh-2.ru-central1.internal,16020,1637214343541'}
                                        SERVER_NAME |                                         REGION_NAME |  START_KEY |
END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | --------------------------------------------------- | ---------- | --------
-- | ----- | ----- | ---------- |
 bds-adh-2.ru-central1.internal,16020,1637214343541 | t1,,1637234844003.562aabcac6ff0688b31260c44c9863c8. |            |
|     0 |     0 |        1.0 |
 1 rows
Took 0.0180 seconds
```

Ограничение выходных полей

```
hbase(main):011:0> list_regions 't1', {}, ['SERVER_NAME','REGION_NAME']
                                        SERVER_NAME |                                         REGION_NAME |
 -------------------------------------------------- | --------------------------------------------------- |
 bds-adh-2.ru-central1.internal,16020,1637214343541 | t1,,1637234844003.562aabcac6ff0688b31260c44c9863c8. |
 1 rows
Took 0.0181 seconds
```

locate_region

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Определяет регион по заданному имени таблицы и ключу строки.

Использование

```
locate_region '[<namespace_name>:]<table_name>', '<row_key>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
row_key	Ключ строки

Примеры

```
hbase(main):021:0> put 't1', 'r1', 'cf1:c1', 'v1'
Took 0.1000 seconds
hbase(main):022:0> locate_region 't1', 'r1'
HOST                                     REGION
 bds-adh-2.ru-central1.internal:16020    {ENCODED => 562aabcac6ff0688b31260c44c9863c8, NAME => 't1,,1637234844003.
562aabcac6ff0688b31260c44c9863c8.', STARTKEY => '', ENDKEY => ''}
1 row(s)
Took 0.0070 seconds
=> #<Java::OrgApacheHadoopHbase::HRegionLocation:0x6814262d>
```

show_filters

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Выводит все фильтры, доступные в HBase.

Использование

```
show_filters
```

Примеры

```
hbase(main):010:0> show_filters
DependentColumnFilter
KeyOnlyFilter
ColumnCountGetFilter
SingleColumnValueFilter
PrefixFilter
SingleColumnValueExcludeFilter
FirstKeyOnlyFilter
ColumnRangeFilter
ColumnValueFilter
TimestampsFilter
FamilyFilter
QualifierFilter
ColumnPrefixFilter
RowFilter
MultipleColumnPrefixFilter
InclusiveStopFilter
PageFilter
ValueFilter
ColumnPaginationFilter
Took 0.0026 seconds
=> #<Java::JavaUtil::HashMap::KeySet:0x2aafa84f>
```

alter_namespace

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Добавление нового свойства
  - Изменение значения свойства
  - Удаление свойства

Определение

Изменяет свойства, определенные при создании пространства имен.

Использование

- Добавление/изменение свойств пространства имен:

```
alter_namespace '<namespace_name>',
                {METHOD => 'set',
                '<property_name>' => '<new_property_value>'}
```

- Удаление свойств пространства имен:

```
alter_namespace '<namespace_name>',
                {METHOD => 'unset',
                NAME => '<property_name>'}
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
property_name	Имя свойства
new_property_value	Новое значение для указанного свойства

Примеры

Добавление нового свойства

```
hbase(main):009:0> alter_namespace 'ns2', {METHOD => 'set', 'prop3' => 'value3'}
Took 0.2304 seconds
hbase(main):010:0> describe_namespace 'ns2'
DESCRIPTION
{NAME => 'ns2', prop1 => 'val1', prop2 => 'val2', prop3 => 'value3'}
Took 0.0039 seconds
=> 1
```

Изменение значения свойства

```
hbase(main):011:0> alter_namespace 'ns2', {METHOD => 'set', 'prop3' => 'value3_new'}
Took 0.2364 seconds
hbase(main):012:0> describe_namespace 'ns2'
DESCRIPTION
{NAME => 'ns2', prop1 => 'val1', prop2 => 'val2', prop3 => 'value3_new'}
Took 0.0072 seconds
=> 1
```

Удаление свойства

```
hbase(main):013:0>  alter_namespace 'ns2', {METHOD => 'unset', NAME => 'prop3'}
Took 0.2491 seconds
hbase(main):014:0> describe_namespace 'ns2'
DESCRIPTION
{NAME => 'ns2', prop1 => 'val1', prop2 => 'val2'}
Took 0.0069 seconds
=> 1
```

create_namespace

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Создание нового пространства имен без свойств
  - Включение таблицы в созданное пространство имен
  - Создание нового пространства имен со свойствами

Определение

Создает новое пространство имен — логическую группу таблиц, аналогичную базе данных в системах реляционных баз данных. Требуется имя пространства имен и, при необходимости, словарь свойств конфигурации пространства имен.

Использование

```
create_namespace '<namespace_name>'[,
                 {'<property_name>' => '<property_value>'[,...]}]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
property_name	Имя свойства
property_value	Значение свойства

Примеры

Создание нового пространства имен без свойств

```
hbase(main):020:0> create_namespace 'ns1'
Took 0.2599 seconds
```

Включение таблицы в созданное пространство имен

Принадлежность таблицы пространству имен определяется при создании таблицы путем указания полного имени таблицы в формате: `<namespace_name>:<table_name>`.

> **ПРИМЕЧАНИЕ**
>
> Для всех последующих обращений к таблице требуется указывать полное имя таблицы: `<namespace_name>:<table_name>`.

```
hbase(main):021:0> create 'ns1:table1','f1'
Created table ns1:table1
Took 1.3259 seconds
=> Hbase::Table - ns1:table1
```

Создание нового пространства имен со свойствами

```
hbase(main):022:0> create_namespace 'ns2', {'prop1' => 'val1', 'prop2' => 'val2'}
Took 0.2446 seconds
```

describe_namespace

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает описание пространства имен.

Использование

```
describe_namespace '<namespace_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен

Примеры

```
hbase(main):024:0> describe_namespace 'ns1'
DESCRIPTION
{NAME => 'ns1'}
Took 0.0162 seconds
=> 1
hbase(main):025:0> describe_namespace 'ns2'
DESCRIPTION
{NAME => 'ns2', prop1 => 'val1', prop2 => 'val2'}
Took 0.0029 seconds
=> 1
```

drop_namespace

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

- Удаление пустого пространства имен
- Удаление непустого пространства имен

Определение

Безвозвратно удаляет пространство имен.

ВНИМАНИЕ

Пространство имен должно быть пустым. Вы не можете удалить пространство имен, которое включает одну или несколько таблиц.

Использование

```
drop_namespace '<namespace_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен

Примеры

Удаление пустого пространства имен

```
hbase(main):002:0> drop_namespace 'ns2'
Took 0.2507 seconds
hbase(main):003:0> describe_namespace 'ns2'

ERROR: Unknown namespace ns2!

Describe the named namespace. For example:
  hbase> describe_namespace 'ns1'

Took 0.0096 seconds
```

Удаление непустого пространства имен

```
hbase(main):001:0> drop_namespace 'ns1'

ERROR: org.apache.hadoop.hbase.constraint.ConstraintException: Only empty namespaces can be removed. Namespace ns1 has 1 tables
        at org.apache.hadoop.hbase.master.procedure.DeleteNamespaceProcedure.prepareDelete(DeleteNamespaceProcedure.java:217)
        at org.apache.hadoop.hbase.master.procedure.DeleteNamespaceProcedure.executeFromState(DeleteNamespaceProcedure.java:78)
        at org.apache.hadoop.hbase.master.procedure.DeleteNamespaceProcedure.executeFromState(DeleteNamespaceProcedure.java:45)
        at org.apache.hadoop.hbase.procedure2.StateMachineProcedure.execute(StateMachineProcedure.java:184)
        at org.apache.hadoop.hbase.procedure2.Procedure.doExecute(Procedure.java:910)
        at org.apache.hadoop.hbase.procedure2.ProcedureExecutor.execProcedure(ProcedureExecutor.java:1674)
        at org.apache.hadoop.hbase.procedure2.ProcedureExecutor.executeProcedure(ProcedureExecutor.java:1461)
        at org.apache.hadoop.hbase.procedure2.ProcedureExecutor.access$900(ProcedureExecutor.java:76)
        at org.apache.hadoop.hbase.procedure2.ProcedureExecutor$WorkerThread.run(ProcedureExecutor.java:1973)

Drop the named namespace. The namespace must be empty.

Took 1.0703 seconds
```

list_namespace

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Вывод всех пространств имен
  - Вывод списка пространства имен, соответствующих регулярному выражению

Определение

Возвращает все пространства имен в HBase. Необязательный параметр регулярного выражения можно использовать для фильтрации вывода.

Использование

```
list_namespace ['namespace_name' | 'namespace_regex']
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
namespace_regex	Регулярное выражение для поиска пространств имен

Примеры

Вывод всех пространств имен

```
hbase(main):001:0> list_namespace
NAMESPACE
default
hbase
ns1
ns2
4 row(s)
Took 0.3307 seconds
```

Вывод списка пространства имен, соответствующих регулярному выражению

```
hbase(main):002:0> list_namespace 'ns*'
NAMESPACE
ns1
ns2
2 row(s)
Took 0.0074 seconds
```

list_namespace_tables

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает все таблицы, входящие в пространство имен.

Использование

```
list_namespace_tables '<namespace_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен

Примеры

```
hbase(main):001:0> list_namespace_tables 'ns1'
TABLE
table1
1 row(s)
Took 0.3366 seconds
=> ["table1"]
```

append

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Добавление в предыдущее значение ячейки постфикса
  - Изменение атрибутов
  - Изменение области видимости

Определение

Дополняет значение ячейки для указанных таблицы/строки/столбца заданным постфиксом. Также позволяет изменять пользовательские атрибуты или видимость этого значения (только одно за раз).

Использование

```
append '[<namespace_name>:]<table_name>',
       '<row_key>',
       '<column_family>:<column_qualifier>',
       '<cell_value_postfix>'[,
       {ATTRIBUTES => {'<cell_attribute_name>' => '<cell_attribute_value>'[,...]}}] | [, {VISIBILITY => '<visibility_expression>'}]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
row_key	Ключ строки
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)
cell_value_postfix	Постфикс для добавления к указанному значению ячейки
cell_attribute_name	Имя атрибута ячейки
cell_attribute_value	Значение указанного атрибута ячейки
visibility_expression	Логическое выражение, которое относится к существующей метке видимости. Могут использоваться следующие логические операторы: `|` (или), `&` (и). Примеры: `SECRET|PRIVATE`, `(SECRET&PRIVATE)|PUBLIC`, `SECRET&PRIVATE&PUBLIC`

Примеры

Добавление в предыдущее значение ячейки постфикса

```
hbase(main):029:0> get 't4', 'r1'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
 cf1:c2                                  timestamp=2222221111111, value=value2
 cf1:c3                                  timestamp=1637324768581, value=value3
 cf1:c4                                  timestamp=1637324859512, value=value4
1 row(s)
Took 0.0092 seconds
hbase(main):030:0> append 't4', 'r1', 'cf1:c3', '_new'
CURRENT VALUE = value3_new
Took 0.0118 seconds
hbase(main):031:0> get 't4', 'r1'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
 cf1:c2                                  timestamp=2222221111111, value=value2
 cf1:c3                                  timestamp=1637326066133, value=value3_new
 cf1:c4                                  timestamp=1637324859512, value=value4
1 row(s)
Took 0.0114 seconds
```

Изменение атрибутов

```
hbase(main):004:0> append 't4', 'r1', 'cf1:c3', '', ATTRIBUTES => {'mykey' => 'myvalue'}
CURRENT VALUE = value3_new
Took 0.5597 seconds
hbase(main):005:0> get 't4', 'r1'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
 cf1:c2                                  timestamp=2222221111111, value=value2
 cf1:c3                                  timestamp=1637326657852, value=value3_new
 cf1:c4                                  timestamp=1637324859512, value=value4
```

Изменение области видимости

```
hbase(main):006:0> append 't4', 'r1', 'cf1:c3', '', {VISIBILITY => 'PRIVATE'}
CURRENT VALUE = value3_new
Took 0.0220 seconds
hbase(main):007:0> get 't4', 'r1'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
 cf1:c2                                  timestamp=2222221111111, value=value2
 cf1:c3                                  timestamp=1637326735791, value=value3_new
 cf1:c4                                  timestamp=1637324859512, value=value4
1 row(s)
Took 0.0117 seconds
```

count

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Подсчет количества всех строк в таблице
  - Подсчет количества строк, имеющих указанный столбец
  - Подсчет количества строк, соответствующих определенному фильтру
  - Подсчет количества строк, имеющих ключи между STARTROW и STOPROW
  - Использование параметров INTERVAL и CACHE

Определение

Возвращает количество строк в таблице. Поскольку вычисление может занять много времени, выводится текущий счет каждые 1000 строк по умолчанию — этот интервал можно изменить, установив входной параметр `INTERVAL`.

Кеширование сканирования также включено. Размер кеша по умолчанию составляет 10 строк — это значение можно изменить, установив входной параметр `CACHE`. Если ваши строки имеют небольшой размер, вы можете увеличить значение по умолчанию.

Использование

- Базовый синтаксис:

```
count '[<namespace_name>:]<table_name>'[,
      COLUMNS => <columns_array[]>][,
      STARTROW => '<start_row_key>'][,
      STOPROW => '<stop_row_key>']
```

- Использование параметров оптимизации:

```
count '[<namespace_name>:]<table_name>'[,
      INTERVAL => <interval_value>][,
      CACHE => <cache_value>]
```

- Использование фильтров:

```
count '[<namespace_name>:]<table_name>',
      {FILTER => "<filter_description>"}
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
columns_array	Массив строк, указанных в формате `'<column_family>:<column_qualifier>'`, где `<column_family>` — имя семейства столбцов; `<column_qualifier>` — квалификатор столбца
start_row_key	Начальное значение для поиска ключей строки
stop_row_key	Конечное значение для поиска ключей строки
interval_value	Интервал для отображения промежуточного результата
cache_value	Размер кеша
filter_description	Фильтр, используемый для поиска ячеек таблицы

Примеры

Подсчет количества всех строк в таблице

```
hbase(main):024:0> count 't4'
1 row(s)
Took 0.0175 seconds
=> 1
```

Подсчет количества строк, имеющих указанный столбец

```
hbase(main):029:0> count 't4', COLUMNS => ['cf1:1']
0 row(s)
Took 0.0079 seconds
=> 0
hbase(main):030:0> count 't4', COLUMNS => ['cf1:c1']
1 row(s)
Took 0.0053 seconds
=> 1
```

Подсчет количества строк, соответствующих определенному фильтру

```
hbase(main):027:0> count 't4', {FILTER => "ValueFilter(=, 'regexstring:value*')"}
1 row(s)
Took 0.0346 seconds
=> 1
```

Подсчет количества строк, имеющих ключи между STARTROW и STOPROW

```
hbase(main):033:0> count 't4', COLUMNS => ['cf1:c1'], STARTROW => 'r1', STOPROW => 'r2'
1 row(s)
Took 0.0064 seconds
=> 1
```

Использование параметров INTERVAL и CACHE

```
hbase(main):025:0> count 't4', INTERVAL => 100000, CACHE => 1000
1 row(s)
Took 0.0112 seconds
=> 1
```

delete

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Удаление значения в указанной таблице/строке/столбце, у которого нет предыдущих версий
  - Удаление значения в указанной таблице/строке/столбце, у которого существует предыдущая версия
  - Удаление указанной версии значения

Определение

Удаляет значение ячейки для указанных таблицы/строки/столбца и — опционально — метки времени.

Использование

```
delete '[<namespace_name>:]<table_name>',
       '<row_key>',
       '<column_family:column_qualifier>'[,
       <cell_timestamp>][,
       {VISIBILITY => '<visibility_expression>'}]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
row_key	Ключ строки
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)
cell_timestamp	Временная метка ячейки
visibility_expression	Логическое выражение, которое относится к существующей метке видимости. Могут использоваться следующие логические операторы: `|` (или), `&` (и). Примеры: `SECRET|PRIVATE`, `(SECRET&PRIVATE)|PUBLIC`, `SECRET&PRIVATE&PUBLIC`

Примеры

Удаление значения в указанной таблице/строке/столбце, у которого нет предыдущих версий

```
hbase(main):002:0> scan 't4'
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329193429, value=7738718b
1 row(s)
Took 0.3796 seconds
hbase(main):003:0> delete 't4', 'r1', 'cf1:c1'
Took 0.0750 seconds
hbase(main):004:0> scan 't4'
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329193429, value=7738718b
1 row(s)
Took 0.0086 seconds
```

Удаление значения в указанной таблице/строке/столбце, у которого существует предыдущая версия

```
hbase(main):009:0> delete 't4', 'r1', 'cf1:c5'
Took 0.0181 seconds
hbase(main):010:0> scan 't4'
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329190124, value=7738718a
1 row(s)
Took 0.0074 seconds
```

Удаление указанной версии значения

```
hbase(main):011:0> scan 't4', {VERSIONS => 5}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329190124, value=7738718a
 r1                                      column=cf1:c5, timestamp=1637329187604, value=7738718W
 r1                                      column=cf1:c5, timestamp=1637328326920, value=7738718M
1 row(s)
Took 0.0091 seconds
hbase(main):012:0> delete 't4', 'r1', 'cf1:c5', 1637329187604
Took 0.0047 seconds
hbase(main):013:0> scan 't4', {VERSIONS => 5}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329190124, value=7738718a
 r1                                      column=cf1:c5, timestamp=1637328326920, value=7738718M
1 row(s)
Took 0.0078 seconds
```

deleteall

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Удаление всех значений ячеек из указанной строки
  - Использование ROWPREFIXFILTER
  - Удаление всех значений ячеек из указанного столбца

Определение

Удаляет все ячейки с предыдущими версиями из указанной строки. Требует имя таблицы, ключ строки и, опционально, имя столбца и метку времени. Позволяет использовать префиксный фильтр вместо ключа строки.

> **ВНИМАНИЕ**
>
> Даже если указана временная метка, команда удалит все ячейки указанного столбца, а не только имеющие заданную временную метку.

Использование

```
deleteall '[<namespace_name>:]<table_name>',
          '<row_key>' | {ROWPREFIXFILTER => '<row_prefix>'}[,
          '<column_family>:<column_qualifier>'[,
          <cell_timestamp>]][,
          {VISIBILITY => '<visibility_expression>'}]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
row_key	Ключ строки
row_prefix	Префикс для поиска ключей строки
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)
cell_timestamp	Временная метка ячейки
visibility_expression	Логическое выражение, которое относится к существующей метке видимости. Могут использоваться следующие логические операторы: `|` (или), `&` (и). Примеры: `SECRET|PRIVATE`, `(SECRET&PRIVATE)|PUBLIC`, `SECRET&PRIVATE&PUBLIC`

Примеры

Удаление всех значений ячеек из указанной строки

```
hbase(main):079:0> scan 't6', VERSIONS => 5
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637426986008, value=value3
 r1                                      column=cf1:c1, timestamp=1637426972064, value=value2
 r1                                      column=cf1:c1, timestamp=1637426961063, value=value1
 r1                                      column=cf1:c2, timestamp=1637426521080, value=value4
 r2                                      column=cf1:c3, timestamp=1637427026420, value=value6
 r2                                      column=cf1:c3, timestamp=1637426999070, value=value5
2 row(s)
Took 0.0066 seconds
hbase(main):080:0> deleteall 't6', 'r1'
Took 0.0067 seconds
hbase(main):081:0> scan 't6', VERSIONS => 5
ROW                                      COLUMN+CELL
 r2                                      column=cf1:c3, timestamp=1637427026420, value=value6
 r2                                      column=cf1:c3, timestamp=1637426999070, value=value5
1 row(s)
Took 0.0054 seconds
```

Использование ROWPREFIXFILTER

```
hbase(main):048:0> scan 't6', VERSIONS => 5
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637426380000, value=value3
 r1                                      column=cf1:c1, timestamp=1637426378038, value=value2
 r1                                      column=cf1:c1, timestamp=1637426371683, value=value1
 r1                                      column=cf1:c2, timestamp=1637426387651, value=value4
 r2                                      column=cf1:c3, timestamp=1637426397389, value=value6
 r2                                      column=cf1:c3, timestamp=1637426395621, value=value5
2 row(s)
Took 0.0437 seconds
hbase(main):049:0> deleteall 't6', {ROWPREFIXFILTER => 'r'}
Took 0.0546 seconds
hbase(main):050:0> scan 't6', VERSIONS => 5
ROW                                      COLUMN+CELL
0 row(s)
Took 0.0051 seconds
```

Удаление всех значений ячеек из указанного столбца

```
hbase(main):057:0> scan 't6', VERSIONS => 5
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637426519471, value=value3
 r1                                      column=cf1:c1, timestamp=1637426517669, value=value2
 r1                                      column=cf1:c1, timestamp=1637426515563, value=value1
 r1                                      column=cf1:c2, timestamp=1637426521080, value=value4
 r2                                      column=cf1:c3, timestamp=1637426525624, value=value6
 r2                                      column=cf1:c3, timestamp=1637426523363, value=value5
2 row(s)
Took 0.0121 seconds
hbase(main):058:0> deleteall 't6', 'r1', 'cf1:c1'
Took 0.0076 seconds
hbase(main):059:0> scan 't6', VERSIONS => 5
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c2, timestamp=1637426521080, value=value4
 r2                                      column=cf1:c3, timestamp=1637426525624, value=value6
 r2                                      column=cf1:c3, timestamp=1637426523363, value=value5
2 row(s)
Took 0.0061 seconds
```

get

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Получение всей строки
  - Получение указанной ячейки столбца
  - Получение нескольких столбцов
  - Использование TIMERANGE
  - Получение нескольких версий значений одного столбца
  - Получение указанной версии значения
  - Использование фильтра

Определение

Возвращает содержимое указанной ячейки строки или столбца.

Использование

- Базовый синтаксис:

```
[source,sql]
get '[<namespace_name>:]<table_name>',
    '<row_key>'[,
    '<column_family>:<column_qualifier>'[,...]]
```

- С использованием словаря:

```
get '[<namespace_name>:]<table_name>',
    '<row_key>',
    {COLUMN => <columns_array[]>[,
    VERSIONS => <number_displayed_versions>][,
    TIMESTAMP => <cell_timestamp>][,
    TIMERANGE => [<timestamp_start>, <timestamp_end>][,
    ATTRIBUTES => {'<cell_attribute_name>' => '<cell_attribute_value>'[,...]}]}
```

- С использованием временного отрезка на уровне строки:

```
get '[<namespace_name>:]<table_name>',
    '<row_key>',
    {TIMERANGE => [<timestamp_start>, <timestamp_end>]}
```

- С использованием фильтров:

```
get '[<namespace_name>:]<table_name>',
    '<row_key>',
    {FILTER => "<filter_description>"}
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
row_key	Ключ строки
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)
columns_array	Массив строк, указанных в формате `'<column_family>:<column_qualifier>'`, где `<column_family>` — имя семейства столбцов; `<column_qualifier>` — квалификатор столбца
number_displayed_versions	Количество версий ячеек, которые необходимо отобразить
cell_timestamp	Временная метка ячейки
timestamp_start	Начальное значение для поиска временных меток ячеек
timestamp_end	Конечное значение для поиска временных меток ячеек
cell_attribute_name	Имя атрибута ячейки
cell_attribute_value	Значение указанного атрибута ячейки
filter_description	Фильтр, используемый для поиска ячеек таблицы

Примеры

Получение всей строки

```
hbase(main):013:0> get 't4', 'r1'
COLUMN                CELL
 cf1:c1               timestamp=1637324524743, value=value1
 cf1:c2               timestamp=2222221111111, value=value2
 cf1:c3               timestamp=1637326735791, value=value3_new
 cf1:c4               timestamp=1637324859512, value=value4
 cf1:c5               timestamp=1637329193429, value=7738718b
1 row(s)
Took 0.0125 seconds
```

Получение указанной ячейки столбца

```
hbase(main):014:0> get 't4', 'r1', 'cf1:c1'
COLUMN                CELL
 cf1:c1               timestamp=1637324524743, value=value1
1 row(s)
Took 0.0239 seconds
```

Получение нескольких столбцов

```
hbase(main):016:0> get 't4', 'r1', {COLUMN => ['cf1:c1', 'cf1:c5']}
COLUMN                CELL
 cf1:c1               timestamp=1637324524743, value=value1
 cf1:c5               timestamp=1637329193429, value=7738718b
1 row(s)
Took 0.0075 seconds
hbase(main):017:0> get 't4', 'r1', 'cf1:c1', 'cf1:c3'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
 cf1:c3                                  timestamp=1637326735791, value=value3_new
1 row(s)
Took 0.0171 seconds
```

Использование TIMERANGE

```
hbase(main):019:0> get 't4', 'r1', {TIMERANGE => [1637324524743, 1637326735791]}
COLUMN                CELL
 cf1:c1               timestamp=1637324524743, value=value1
 cf1:c4               timestamp=1637324859512, value=value4
1 row(s)
Took 0.0127 seconds
```

Получение нескольких версий значений одного столбца

```
hbase(main):021:0> get 't4', 'r1', {COLUMN => 'cf1:c5', VERSIONS => 5}
COLUMN                CELL
 cf1:c5               timestamp=1637329193429, value=7738718b
 cf1:c5               timestamp=1637329190124, value=7738718a
 cf1:c5               timestamp=1637329187604, value=7738718W
 cf1:c5               timestamp=1637328326920, value=7738718M
1 row(s)
Took 0.0148 seconds
```

Получение указанной версии значения

```
hbase(main):022:0> get 't4', 'r1', {COLUMN => 'cf1:c5', TIMESTAMP => 1637329190124}
COLUMN                CELL
 cf1:c5               timestamp=1637329190124, value=7738718a
1 row(s)
Took 0.0146 seconds
```

Использование фильтра

```
hbase(main):015:0> get 't4', 'r1', {FILTER => "ValueFilter(=, 'regexstring:value*')"}
COLUMN                CELL
 cf1:c1               timestamp=1637324524743, value=value1
 cf1:c2               timestamp=2222221111111, value=value2
 cf1:c3               timestamp=1637326735791, value=value3_new
 cf1:c4               timestamp=1637324859512, value=value4
1 row(s)
Took 0.0973 seconds
```

get_counter

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает счетчик для значения ячейки по указанным координатам таблицы/строки/столбца. Счетчик должен изменяться с помощью функций атомарного инкремента (incr), а значение данных должно быть закодировано в двоичном формате (как long).

Использование

```
get_counter '[<namespace_name>:]<table_name>',
            '<row_key>',
            '<column_family>:<column_qualifier>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
row_key	Ключ строки
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)

Примеры

```
hbase(main):049:0> incr 't4', 'r1', 'cf1:c5', 11
COUNTER VALUE = 3978705112342280269
Took 0.0079 seconds
hbase(main):050:0> get 't4','r1','cf1:c5'
COLUMN                                    CELL
 cf1:c5                                   timestamp=1637328326920, value=7738718M
1 row(s)
Took 0.0091 seconds
hbase(main):001:0> get_counter 't4', 'r1', 'cf1:c5'
COUNTER VALUE = 3978705112342280269
Took 0.4473 seconds
```

get_splits

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
  Таблица без разделения
  Таблица с разделением

Определение

Возвращает splits данной таблицы — значения ключа строки, используемые для определения границ региона. Дополнительная информация доступна в статье Архитектура HBase.

Использование

```
get_splits '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

Таблица без разделения

```
hbase(main):001:0> get_splits 't4'
Total number of splits = 1
Took 0.4184 seconds
=> []
```

Таблица с разделением

```
hbase(main):014:0> get_splits 'people_ages'
Total number of splits = 5
F
K
P
W
Took 0.0128 seconds
=> ["F", "K", "P", "W"]
```

incr

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Увеличение с шагом 1 по умолчанию
  - Увеличение с кастомным шагом

Определение

Увеличивает значение ячейки по указанным координатам таблицы/строки/столбца. Шаг увеличения по умолчанию равен `1`, также можно определить свой собственный. Кроме того, команда позволяет изменять пользовательские атрибуты или видимость значения ячейки.

> **ВНИМАНИЕ**
>
> Команду `incr` можно применить только к ячейкам, содержащим двоичные (long) значения.

Использование

```
$ incr '[<namespace_name>:]<table_name>',
     '<row_key>',
     '<column_family>:<column_qualifier>'[,
     '<incr_step>'][,
     {ATTRIBUTES => {'<cell_attribute_name>' => '<cell_attribute_value>'[,...]}}] | [, {VISIBILITY => '<visibility_expression>'}]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
row_key	Ключ строки
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)
incr_step	Шаг инкремента. Значение по умолчанию: `1`.
cell_attribute_name	Имя атрибута ячейки
cell_attribute_value	Значение указанного атрибута ячейки
visibility_expression	Логическое выражение, которое относится к существующей метке видимости. Могут использоваться следующие логические операторы: `|` (или), `&` (и). Примеры: `SECRET|PRIVATE`, `(SECRET&PRIVATE)|PUBLIC`, `SECRET&PRIVATE&PUBLIC`

Примеры

Увеличение с шагом 1 по умолчанию

```
hbase(main):043:0> put 't4', 'r1', 'cf1:c5', 77387187
Took 0.0055 seconds
hbase(main):044:0> incr 't4', 'r1', 'cf1:c5'
COUNTER VALUE = 3978705112342280248
Took 0.0074 seconds
hbase(main):045:0> get 't4', 'r1', 'cf1:c5'
COLUMN                                    CELL
 cf1:c5                                   timestamp=1637328241031, value=77387188
1 row(s)
Took 0.0050 seconds
```

Увеличение с кастомным шагом

```
hbase(main):047:0> incr 't4', 'r1', 'cf1:c5', 10
COUNTER VALUE = 3978705112342280258
Took 0.0074 seconds
hbase(main):048:0> get 't4', 'r1', 'cf1:c5'
COLUMN                                    CELL
 cf1:c5                                   timestamp=1637328290099, value=7738718B
1 row(s)
Took 0.0098 seconds
```

put

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Сохранение значения без метки времени и каких-либо атрибутов
  - Сохранение значения с предопределенной отметкой времени
  - Сохранение значения с дополнительными атрибутами
  - Сохранение значения с назначением видимости

Определение

Вставляет значение ячейки для указанных таблицы, строки, столбца и, при необходимости, временной метки.

Использование

```
put '[<namespace_name>:]<table_name>',
    '<row_key>',
    '<column_family>:<column_qualifier>',
    '<cell_value>'[,
    <cell_timestamp>][,
    {ATTRIBUTES => {'<cell_attribute_name>' => '<cell_attribute_value>'[,...]}}][,
    {VISIBILITY => '<visibility_expression>'}]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
row_key	Ключ строки
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)
cell_value	Значение вставляемой ячейки
cell_timestamp	Временная метка ячейки
cell_attribute_name	Имя атрибута ячейки
cell_attribute_value	Значение указанного атрибута ячейки
visibility_expression	Логическое выражение, которое относится к существующей метке видимости. Могут использоваться следующие логические операторы: `|` (или), `&` (и). Примеры: `SECRET|PRIVATE`, `(SECRET&PRIVATE)|PUBLIC`, `SECRET&PRIVATE&PUBLIC`

Примеры

Сохранение значения без метки времени и каких-либо атрибутов

```
hbase(main):001:0> put 't4', 'r1', 'cf1:c1', 'value1'
Took 0.5673 seconds
hbase(main):002:0> get 't4', 'r1'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
1 row(s)
Took 0.0396 seconds
```

Сохранение значения с предопределенной отметкой времени

```
hbase(main):009:0> put 't4', 'r1', 'cf1:c2', 'value2', 2222221111111
Took 0.0073 seconds
hbase(main):010:0> get 't4', 'r1'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
 cf1:c2                                  timestamp=2222221111111, value=value2
1 row(s)
Took 0.0104 seconds
```

Сохранение значения с дополнительными атрибутами

```
hbase(main):011:0> put 't4', 'r1', 'cf1:c3', 'value3', {ATTRIBUTES => {'my_prop_name' => 'my_prop_value'}}
Took 0.0123 seconds
hbase(main):012:0> get 't4', 'r1'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
 cf1:c2                                  timestamp=2222221111111, value=value2
 cf1:c3                                  timestamp=1637324768581, value=value3
1 row(s)
Took 0.0086 seconds
```

Сохранение значения с назначением видимости

```
hbase(main):013:0> put 't4', 'r1', 'cf1:c4', 'value4', {VISIBILITY => 'SECRET'}
Took 0.0231 seconds
hbase(main):014:0> get 't4', 'r1'
COLUMN                                   CELL
 cf1:c1                                  timestamp=1637324524743, value=value1
 cf1:c2                                  timestamp=2222221111111, value=value2
 cf1:c3                                  timestamp=1637324768581, value=value3
 cf1:c4                                  timestamp=1637324859512, value=value4
1 row(s)
Took 0.0109 seconds
```

scan

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

- Сканирование всей таблицы
- Сканирование указанной ячейки столбца
- Сканирование нескольких столбцов
- Использование TIMERANGE
- Сканирование нескольких версий значений одного столбца
- Сканирование указанной версии значения
- Использование фильтра для значений столбца
- Использование префикса для поиска ключей строки
- Получение всех статистических показателей
- Получение определенных статистических показателей
- Включая ячейки, отмеченные для удаления

Определение

Сканирует таблицу и возвращает все строки, соответствующие заданным спецификациям.

Использование

- Базовый синтаксис:

```
scan '[<namespace_name>:]<table_name>'[,
     '<column_family>:<column_qualifier>']
```

- С использованием словарей:

```
scan '[<namespace_name>:]<table_name>',
     {COLUMN => <columns_array[]>[,
     VERSIONS => <number_displayed_versions>][,
     TIMESTAMP => <cell_timestamp>][,
     TIMERANGE => [<timestamp_start>, <timestamp_end>][,
     ATTRIBUTES => {'<cell_attribute_name>' => '<cell_attribute_value>'[,...]}]}
```

- На уровне таблицы:

```
scan '[<namespace_name>:]<table_name>',
     {TIMERANGE => [<timestamp_start>, <timestamp_end>]}
```

- С использованием фильтров:

```
scan '[<namespace_name>:]<table_name>',
     {[ROWPREFIXFILTER => '<row_prefix>'][,
     FILTER => "<filter_description>"]}`
```

- С использованием метрик:

```
scan '[<namespace_name>:]<table_name>',
     {ALL_METRICS => true} | {METRICS => <metrics_array[]>}
```

- Включая ячейки, помеченные как удаленные:

```
scan '[<namespace_name>:]<table_name>',
     {RAW => true, VERSIONS => <number_displayed_versions>}
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)
columns_array	Массив строк, указанных в формате `'<column_family>:<column_qualifier>'`, где `<column_family>` — имя семейства столбцов; `<column_qualifier>` — квалификатор столбца
number_displayed_versions	Количество версий ячеек, которые необходимо отобразить
cell_timestamp	Временная метка ячейки
timestamp_start	Начальное значение для поиска временных меток ячеек
timestamp_end	Конечное значение для поиска временных меток ячеек
cell_attribute_name	Имя атрибута ячейки
cell_attribute_value	Значение указанного атрибута ячейки
row_prefix	Префикс для поиска ключей строки
filter_description	Фильтр, используемый для поиска ячеек таблицы
metrics_array	Массив имен метрик статистики

Примеры

Сканирование всей таблицы

```
hbase(main):003:0> scan 't4'
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329193429, value=7738718b
1 row(s)
Took 0.0744 seconds
```

Сканирование указанной ячейки столбца

```
hbase(main):010:0> scan 't4', {COLUMN => 'cf1:c1'}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
1 row(s)
Took 0.0103 seconds
```

Сканирование нескольких столбцов

```
hbase(main):008:0> scan 't4', {COLUMN => ['cf1:c1', 'cf1:c2']}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
1 row(s)
Took 0.0149 seconds
```

Использование TIMERANGE

```
hbase(main):009:0> scan 't4', {TIMERANGE => [1637324524743, 1637326735791]}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
1 row(s)
Took 0.0129 seconds
```

Сканирование нескольких версий значений одного столбца

```
hbase(main):011:0> scan 't4', {COLUMN => 'cf1:c5', VERSIONS => 5}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c5, timestamp=1637329193429, value=7738718b
 r1                                      column=cf1:c5, timestamp=1637329190124, value=7738718a
 r1                                      column=cf1:c5, timestamp=1637329187604, value=7738718W
 r1                                      column=cf1:c5, timestamp=1637328326920, value=7738718M
1 row(s)
Took 0.0085 seconds
```

Сканирование указанной версии значения

```
hbase(main):016:0> scan 't4', {COLUMN => 'cf1:c5', TIMESTAMP => 1637329190124}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c5, timestamp=1637329190124, value=7738718a
1 row(s)
Took 0.0070 seconds
```

Использование фильтра для значений столбца

```
hbase(main):019:0> scan 't4', {FILTER => "ValueFilter(=, 'regexstring:value*')"}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
1 row(s)
Took 0.0380 seconds
```

Использование префикса для поиска ключей строки

```
hbase(main):026:0> scan 't4', {ROWPREFIXFILTER => 'r'}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329193429, value=7738718b
1 row(s)
Took 0.0077 seconds
```

Получение всех статистических показателей

```
hbase(main):022:0> scan 't4', {ALL_METRICS => true}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329193429, value=7738718b
1 row(s)

METRIC                                   VALUE
 BYTES_IN_REMOTE_RESULTS                 191
 BYTES_IN_RESULTS                        191
 MILLIS_BETWEEN_NEXTS                    5
 NOT_SERVING_REGION_EXCEPTION            0
 REGIONS_SCANNED                         1
 REMOTE_RPC_CALLS                        1
 REMOTE_RPC_RETRIES                      0
 ROWS_FILTERED                           0
 ROWS_SCANNED                            1
 RPC_CALLS                               1
 RPC_RETRIES                             0
Took 0.0079 seconds
```

Получение определенных статистических показателей

```
hbase(main):028:0> scan 't4', {METRICS => ['RPC_RETRIES', 'ROWS_FILTERED']}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329193429, value=7738718b
1 row(s)

METRIC                                   VALUE
 ROWS_FILTERED                           0
 RPC_RETRIES                             0
Took 0.0131 seconds
```

Включая ячейки, отмеченные для удаления

```
hbase(main):001:0> scan 't4', {RAW => true, VERSIONS => 10}
ROW                                      COLUMN+CELL
 r1                                      column=cf1:c1, timestamp=1637324524743, value=value1
 r1                                      column=cf1:c2, timestamp=2222221111111, value=value2
 r1                                      column=cf1:c3, timestamp=1637326735791, value=value3_new
 r1                                      column=cf1:c4, timestamp=1637324859512, value=value4
 r1                                      column=cf1:c5, timestamp=1637329193429, value=7738718b
 r1                                      column=cf1:c5, timestamp=1637329190124, value=7738718a
 r1                                      column=cf1:c5, timestamp=1637329187604, value=7738718W
 r1                                      column=cf1:c5, timestamp=1637328326920, value=7738718M
 r1                                      column=cf2:c1, timestamp=1637329303310, type=Delete
 r1                                      column=cf2:c1, timestamp=1637329303310, value=last
 r1                                      column=cf2:c1, timestamp=1637329264711, type=Delete
 r1                                      column=cf2:c1, timestamp=1637329264711, value=value_new_new2
 r1                                      column=cf2:c1, timestamp=1637329262403, type=Delete
 r1                                      column=cf2:c1, timestamp=1637329262403, value=value_new
 r1                                      column=cf2:c1, timestamp=1637329253123, type=Delete
 r1                                      column=cf2:c1, timestamp=1637329253123, value=value
1 row(s)
Took 0.3665 seconds
```

truncate

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Отключает, удаляет и воссоздает указанную таблицу.

Использование

```
truncate '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):004:0> truncate 't4'
Truncating 't4' table (it may take a while):
Disabling table...
Truncating table...
Took 2.1331 seconds
```

truncate_preserve

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Отключает, удаляет и воссоздает указанную таблицу, сохраняя прежние границы региона.

Использование

```
truncate_preserve '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):005:0> truncate_preserve 't4'
Truncating 't4' table (it may take a while):
Disabling table...
Truncating table...
Took 2.0099 seconds
```

assign

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Назначает определенный регион одному из Region Servers. Больше информации о назначении регионов доступно в документации HBase.

ВНИМАНИЕ

Если определенный регион уже назначен, эта команда принудительно переназначит его. Используйте команду с осторожностью.

Использование

```
assign '<region_name>' | '<encoded_region_name>'
```

Аргументы

Параметр	Описание
region_name	Имя региона без закодированной части. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то `region_name` — это `wordcount2,m,1637833871529`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
encoded_region_name	Закодированная часть названия региона. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то закодированная часть — это `2ff3683a90c4106432c540ddd0f57cba`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)

Примеры

```
hbase(main):007:0> assign '2ff3683a90c4106432c540ddd0f57cba'
Took 1.2233 seconds
hbase(main):008:0> assign 'wordcount2,m,1637833871529'
Took 1.1642 seconds
```

balancer

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Запускает балансировщик кластера (Balancer). Возвращает `true`, если балансировщик работает и может сообщить Region Servers отменить назначение всех регионов для балансировки (само переназначение является асинхронным). В противном случае возвращается `false` — в большинстве случаев, когда некоторые регионы находятся в переходном состоянии (в таких случаях можно использовать входной параметр `force` для принудительного выполнения команды).

ВНИМАНИЕ

Эта команда для экспертного пользования. Принудительная балансировка может нанести урон больший, чем восстановление, если назначение регионов закончилось ошибкой.

Использование

```
balancer ['force']
```

Аргументы

Параметр	Описание
force	Принудительно запускает балансировку, даже если некоторые регионы находятся в переходном состоянии. По умолчанию команда в таком случае не запускается

Примеры

```
hbase(main):050:0> balancer
true
Took 0.0196 seconds
=> 1
```

balancer_enabled

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Проверяет состояние балансировщика (Balancer). Возвращает `true`, если балансировщик включен, иначе возвращает `false`.

Использование

```
balancer_enabled
```

Примеры

```
hbase(main):009:0> balancer_enabled
true
Took 0.0161 seconds
=> true
```

balance_switch

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
  Отключение
  Включение

Определение

Включает/отключает балансировщик кластера (Balancer). Возвращает предыдущее состояние балансировщика.

Использование

```
balance_switch <new_state>
```

Аргументы

Параметр	Описание
new_state	Логическое значение, определяющее новое состояние балансировщика. Значение `true` включает балансировщик, `false` — выключает

Примеры

Отключение

```
hbase(main):047:0> balance_switch false
Previous balancer state : true
Took 0.0249 seconds
=> "true"
```

Включение

```
hbase(main):048:0> balance_switch true
Previous balancer state : false
Took 0.0089 seconds
=> "false"
```

catalogjanitor_enabled

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Проверяет состояние CatalogJanitor. Возвращает `true`, если CatalogJanitor включен, в противном случае возвращает `false`.

Использование

```
catalogjanitor_enabled
```

Примеры

```
hbase(main):038:0> catalogjanitor_enabled
true
Took 0.0112 seconds
=> 1
```

catalogjanitor_run

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Запускает CatalogJanitor для проверки и очистки таблицы hbase:meta (ранее называлась `.META.`).

Использование

```
catalogjanitor_run
```

Примеры

```
hbase(main):043:0> catalogjanitor_run
Took 0.0118 seconds
=> 0
```

catalogjanitor_switch

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
  Отключение
  Включение

Определение

Включает/отключает CatalogJanitor. Возвращает предыдущее состояние CatalogJanitor.

Использование

```
catalogjanitor_switch <new_state>
```

Аргументы

Параметр	Описание
new_state	Логическое значение, определяющее новое состояние CatalogJanitor. Значение `true` включает CatalogJanitor, `false` — выключает.

Примеры

Отключение

```
hbase(main):040:0> catalogjanitor_switch false
true
Took 0.0179 seconds
=> 1
```

Включение

```
hbase(main):041:0> catalogjanitor_switch true
false
Took 0.0033 seconds
=> 1
```

cleaner_chore_enabled

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Проверяет состояние Cleaner chore. Возвращает `true`, если Cleaner chore активен, в противном случае возвращает `false`.

Использование

```
cleaner_chore_enabled
```

Примеры

```
hbase(main):008:0> cleaner_chore_enabled
true
Took 0.0081 seconds
=> 1
```

cleaner_chore_run

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Запускает сборку мусора файлов HFiles и WAL, используя Cleaner chore.

Использование

```
cleaner_chore_run
```

Примеры

```
hbase(main):002:0> cleaner_chore_run
Took 0.3230 seconds
=> false
```

cleaner_chore_switch

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Отключение
  - Включение

Определение

Включает/отключает Cleaner chore. Возвращает предыдущее состояние Cleaner chore.

Использование

```
cleaner_chore_switch <new_state>
```

Аргументы

Параметр	Описание
new_state	Логическое значение, которое определяет новое состояние Cleaner chore. Значение `true` включает, `false` — отключает

Примеры

Отключение

```
hbase(main):010:0> cleaner_chore_switch false
true
Took 0.0123 seconds
=> 1
```

Включение

```
hbase(main):011:0> cleaner_chore_switch true
false
Took 0.0034 seconds
=> 1
```

clear_block_cache

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Удаляет все блоки, соответствующие указанной таблице, из BlockCache.

ВНИМАНИЕ

Эта команда может значительно повлиять на производительность запроса, поскольку последующие запросы должны будут извлекать блоки из файловой системы. Используйте с осторожностью.

Использование

```
clear_block_cache '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):006:0> clear_block_cache 'ns1:temp2'
CacheEvictionStats{evictedBlocks=0, maxCacheSize=1638085504, failedRegionsSize=0, failedRegions=[]}
Took 0.1503 seconds
=> 1
```

clear_compaction_queues

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Очищает очереди сжатия (compaction) на указанном Region Server.

Использование

```
clear_compaction_queues '<region_server_name>,<port_number>'[, <queue_names_array[]>]
```

Аргументы

Параметр	Описание
region_server_name	Имя Region Server
port_number	Номер порта
queue_names_array	Массив с возможными строковыми значениями: - `long` — longCompactions-очередь (поток Large Compaction); - `short` — shortCompactions-очередь (поток Small Compaction).

TIP

Для получения `<server_name>`, `<port_number>` и `<start_code>` используйте команду status 'simple'.

Примеры

```
hbase(main):002:0> clear_compaction_queues 'bds-adh-1.ru-central1.internal,16020'
Took 0.0233 seconds
hbase(main):003:0> clear_compaction_queues 'bds-adh-1.ru-central1.internal,16020', ['long']
Took 0.0047 seconds
```

clear_deadservers

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Удаляет все отключенные (dead) или указанные Region Servers из HBase. Необязательный входной параметр можно использовать для фильтрации dead-серверов. Возвращает ошибку, если список dead-серверов пуст.

Использование

```
clear_deadservers ['<region_server_name>,<port_number>,<start_code>'[,...]]
```

Аргументы

Параметр	Описание
region_server_name	Имя Region Server
port_number	Номер порта
start_code	Стартовый код сервера

TIP

Для получения `<server_name>`, `<port_number>` и `<start_code>` используйте команду status 'simple'.

Примеры

```
hbase(main):008:0> clear_deadservers

ERROR: servers cannot be null or empty

          Clear the dead region servers that are never used.
          Examples:
          Clear all dead region servers:
          hbase> clear_deadservers
          Clear the specified dead region servers:
          hbase> clear_deadservers 'host187.example.com,60020,1289493121758'
          or
          hbase> clear_deadservers 'host187.example.com,60020,1289493121758',
                                   'host188.example.com,60020,1289493121758'

Took 0.0166 seconds
```

close_region

Константин Алпашкин

Назад к содержанию

ВАЖНО

Команда устарела. Пожалуйста, используйте unassign.

compact

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Таблица
  - Семейство столбцов (в таблице)
  - Регион
  - Семейство столбцов (в пределах региона)

Определение

Выполняет сжатие (compaction) для определенного семейства таблиц/регионов/столбцов. Смотрите также major_compact.

Использование

```
compact '[<namespace_name>:]<table_name>' | '<region_name>' | '<encoded_region_name>'[,
        '<column_family>' | nil[,
        '<compact_type>']]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
region_name	Имя региона без закодированной части. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то `region_name` — это `wordcount2,m,1637833871529`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
encoded_region_name	Закодированная часть названия региона. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то закодированная часть — это `2ff3683a90c4106432c540ddd0f57cba`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
column_family	Имя семейства столбцов
compact_type	Возможные значения: ▪ `NORMAL` — обычное сжатие; ▪ `MOB` — сжатие объектов средней величины (medium-sized objects, MOB). По умолчанию, используется значение `NORMAL`. Больше информации доступно в документации HBase

Примеры

Таблица

```
hbase(main):023:0> compact 'ns1:temp2'
Took 0.0483 seconds
```

Семейство столбцов (в таблице)

```
hbase(main):026:0> compact 'ns1:temp2', 'cf1'
Took 0.0252 seconds
```

Регион

```
hbase(main):027:0> compact 'wordcount2,m,1637833871529'
Took 0.2334 seconds
hbase(main):028:0> compact '2ff3683a90c4106432c540ddd0f57cba'
Took 0.0151 seconds
```

Семейство столбцов (в пределах региона)

```
hbase(main):030:0> compact '2ff3683a90c4106432c540ddd0f57cba', 'f'
Took 0.0121 seconds
```

compaction_state

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Получает статус сжатия (compaction) для определенной таблицы.

Использование

```
compaction_state '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):002:0> compaction_state 'ns1:temp2'
Took 0.1477 seconds
=> "NONE"
```

compact_rs

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Сжимает все регионы в пределах определенного Region Server.

Использование

```
compact_rs '<region_server_name>,<port_number>[,<start_code>]'[, <is_major>]
```

Аргументы

Параметр	Описание
region_server_name	Имя Region Server
port_number	Номер порта
start_code	Стартовый код сервера
is_major	Значение `true` запускает Major compaction

TIP

Для получения `<server_name>`, `<port_number>` и `<start_code>` используйте команду status 'simple'.

Примеры

```
hbase(main):003:0> compact_rs 'bds-adh-1.ru-central1.internal,16020,1642487399069'
Took 0.1098 seconds
hbase(main):004:0> compact_rs 'bds-adh-1.ru-central1.internal,16020,1642487399069', true
Took 0.0795 seconds
```

flush

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Таблица
  - Region Server
  - Имя региона
  - Закодированное имя региона

Определение

Сбрасывает (flush) все регионы для определенной таблицы/Region Server или все регионы с указанным именем. Сброс означает перемещение информации из MemStore в HFiles. По умолчанию этот процесс выполняется автоматически, когда MemStore заполняется.

Использование

```
flush '[<namespace_name>:]<table_name>' | '<region_server_name>,<port_number>,<start_code>' | '<region_name>' |
'<encoded_region_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
region_server_name	Имя Region Server
port_number	Номер порта
start_code	Стартовый код сервера
region_name	Имя региона без закодированной части. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то `region_name` — это `wordcount2,m,1637833871529`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
encoded_region_name	Закодированная часть названия региона. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то закодированная часть — это `2ff3683a90c4106432c540ddd0f57cba`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)

TIP

Для получения `<server_name>`, `<port_number>` и `<start_code>` используйте команду status 'simple'.

Примеры

Таблица

```
hbase(main):019:0> flush 'BEST_BOOKS'
Took 0.2079 seconds
```

Region Server

```
hbase(main):020:0> flush 'bds-adh-2.ru-central1.internal,16020,1642400781522'
Took 0.1736 seconds
```

Имя региона

```
hbase(main):021:0> flush 'BEST_BOOKS,K,1642431463389'
Took 0.0034 seconds
```

Закодированное имя региона

```
hbase(main):022:0> flush 'd6e6be2ad047bc3f2ea5ed6842f851b0'
Took 0.0076 seconds
```

is_in_maintenance_mode

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Проверяет, находится ли Master-сервер в режиме maintenance.

Использование

```
is_in_maintenance_mode
```

Примеры

```
hbase(main):004:0> is_in_maintenance_mode
false
Took 0.3552 seconds
=> false
```

list_deadservers

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Список всех отключенных (dead) Region Servers.

Использование

```
list_deadservers
```

Примеры

```
hbase(main):006:0> list_deadservers
SERVERNAME
0 row(s)
Took 0.0316 seconds
```

major_compact

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Таблица
  - Семейство столбцов (в таблице)
  - Регион
  - Семейство столбцов (в пределах региона)

Определение

Запускает процедуру major compaction для определенных таблицы/региона/семейства столбцов. Больше информации доступно в разделе compact.

Использование

```
major_compact '[<namespace_name>:]<table_name>' | '<region_name>' | '<encoded_region_name>'[,
              '<column_family>' | nil[,
              '<compact_type>']]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
region_name	Имя региона без закодированной части. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то `region_name` — это `wordcount2,m,1637833871529`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
encoded_region_name	Закодированная часть названия региона. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то закодированная часть — это `2ff3683a90c4106432c540ddd0f57cba`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
column_family	Имя семейства столбцов
compact_type	Возможные значения: ▪ `NORMAL` — обычное сжатие; ▪ `MOB` — сжатие объектов средней величины (medium-sized objects, MOB). По умолчанию, используется значение `NORMAL`. Больше информации доступно в документации HBase

Примеры

Таблица

```
hbase(main):031:0> major_compact 'ns1:temp2'
Took 0.0349 seconds
```

Семейство столбцов (в таблице)

```
hbase(main):032:0> major_compact 'ns1:temp2', 'cf1'
Took 0.0175 seconds
```

Регион

```
hbase(main):033:0> major_compact 'wordcount2,m,1637833871529'
Took 0.0179 seconds
hbase(main):034:0> major_compact '2ff3683a90c4106432c540ddd0f57cba'
Took 0.0105 seconds
```

Семейство столбцов (в пределах региона)

```
hbase(main):035:0> major_compact '2ff3683a90c4106432c540ddd0f57cba', 'f'
Took 0.0125 seconds
```

merge_region

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Объединяет два региона. Принимает полные или закодированные имена регионов в качестве входных данных.

Использование

```
merge_region '<full_region_name1>' | '<encoded_region_name1>',
             '<full_region_name2>' | '<encoded_region_name2>'[,
             '<force_merge>']
```

Аргументы

Параметр	Описание
full_region_name1	Полное название первого региона. Можно получить через вывод команды list_regions (поле `REGION_NAME`)
encoded_region_name1	Закодированная часть имени первого региона. Например, если полное имя региона — `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то закодированная часть будет `2ff3683a90c4106432c540ddd0f57cba`. Полное имя региона и его закодированную часть можно найти в выводе команды list_regions (поле `REGION_NAME`)
full_region_name2	Полное название второго региона
encoded_region_name2	Закодированная часть имени второго региона
force_merge	Значение `true` принудительно запускает процесс объединения, даже если переданы смежные регионы (только для экспертного пользования)

Примеры

```
hbase(main):001:0> list_regions 'BEST_BOOKS'
                                        SERVER_NAME |                                                 REGION_NAME |  START_KEY |
END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | ------------------------------------------------------------ | ---------- |
---------- | ----- | ----- | ---------- |
 bds-adh-2.ru-central1.internal,16020,1642400781522 |  BEST_BOOKS,,1642431463389.dc0dd6bee8c1b1d7db462a17fbd33673. |            |
K |     0 |     0 |        1.0 |
 bds-adh-2.ru-central1.internal,16020,1642400781522 | BEST_BOOKS,K,1642431463389.d6e6be2ad047bc3f2ea5ed6842f851b0. |          K |
|     0 |     0 |        1.0 |
 2 rows
Took 0.5506 seconds
hbase(main):002:0> merge_region 'BEST_BOOKS,,1642431463389.dc0dd6bee8c1b1d7db462a17fbd33673.', 'BEST_BOOKS,K,1642431463389.
d6e6be2ad047bc3f2ea5ed6842f851b0.'
Took 0.1575 seconds
hbase(main):004:0> list_regions 'BEST_BOOKS'
                                        SERVER_NAME |                                                REGION_NAME |  START_KEY |
END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | ----------------------------------------------------------- | ---------- |
---------- | ----- | ----- | ---------- |
 bds-adh-2.ru-central1.internal,16020,1642400781522 | BEST_BOOKS,,1642432455266.182c4a28649924be67aba3eeb9d15a04. |            |
|     0 |     0 |        1.0 |
 1 rows
Took 0.0185 seconds
```

move

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Перемещает указанный регион. Если целевой Region Server не указан, он выбирается случайным образом.

РЕКОМЕНДАЦИЯ

Новое расположение региона можно проверить с помощью команды locate_region. Обратите внимание, что для того, чтобы команда вернула обновленное расположение, может потребоваться некоторое время.

Использование

```
move '<encoded_region_name>'[, '<region_server_name>,<port_number>,<start_code>']
```

Аргументы

Параметр	Описание
encoded_region_name	Закодированная часть названия региона. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то закодированная часть — это `2ff3683a90c4106432c540ddd0f57cba`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
region_server_name	Имя Region Server
port_number	Номер порта
start_code	Стартовый код сервера

TIP

Для получения `<server_name>`, `<port_number>` и `<start_code>` используйте команду status 'simple'.

Примеры

```
hbase(main):025:0> locate_region 'wordcount2','m'
HOST                                     REGION
 bds-adh-1.ru-central1.internal:16020    {ENCODED => 2ff3683a90c4106432c540ddd0f57cba, NAME => 'wordcount2,m,1637833871529.
2ff3683a90c4106432c540ddd0f57cba.'
                                         , STARTKEY => 'm', ENDKEY => 'r'}
1 row(s)
Took 0.0011 seconds
=> #<Java::OrgApacheHadoopHbase::HRegionLocation:0x403f7aa2>
hbase(main):026:0> move '2ff3683a90c4106432c540ddd0f57cba', 'bds-adh-3.ru-central1.internal,16020,1642400781435'
Took 1.1472 seconds
hbase(main):030:0> locate_region 'wordcount2','m'
HOST                                     REGION
 bds-adh-3.ru-central1.internal:16020    {ENCODED => 2ff3683a90c4106432c540ddd0f57cba, NAME => 'wordcount2,m,1637833871529.
2ff3683a90c4106432c540ddd0f57cba.'
                                         , STARTKEY => 'm', ENDKEY => 'r'}
1 row(s)
Took 0.0056 seconds
=> #<Java::OrgApacheHadoopHbase::HRegionLocation:0x42b43a6>
```

normalize

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Запускает Region Normalizer для всех таблиц, для которых установлен флаг `NORMALIZATION_ENABLED`. Возвращает `true`, если Region Normalizer успешно запустился, в противном случае возвращает `false`.

ВАЖНО

Эта команда не действует, если Region Normalizer отключен. Убедитесь, что Region Normalizer включен, используя команду normalizer_switch.

Использование

```
normalize
```

Примеры

```
hbase(main):036:0> normalize
true
Took 0.0198 seconds
=> 1
```

normalizer_enabled

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Проверяет состояние Region Normalizer. Возвращает `true`, если Region Normalizer включен, иначе возвращает `false`.

Использование

```
normalizer_enabled
```

Примеры

```
hbase(main):029:0> normalizer_enabled
true
Took 0.0177 seconds
=> 1
```

normalizer_switch

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Отключение
  - Включение

Определение

Включает/отключает Region Normalizer. Возвращает предыдущее состояние Region Normalizer.

Использование

```
normalizer_switch <new_state>
```

Аргументы

Параметр	Описание
new_state	Логическое значение, определяющее новое состояние Region Normalizer. Значение `true` включает нормализатор, `false` — отключает

Примеры

Отключение

```
hbase(main):033:0> normalizer_switch false
true
Took 0.0098 seconds
=> 1
```

Включение

```
hbase(main):034:0> normalizer_switch true
false
Took 0.0060 seconds
=> 1
```

split

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Разделение региона
  - Разделение таблицы

Определение

Разбивает всю таблицу или указанный регион на два новых региона.

Использование

```
split '[<namespace_name>:]<table_name>' | '<region_name>'[,
      '<split_key>']
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
region_name	Имя региона без закодированной части. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то `region_name` — это `wordcount2,m,1637833871529`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
split_key	Значение ключа строки для разделения таблицы на две части

Примеры

Разделение региона

```
hbase(main):005:0> list_regions 'people_ages'
                                        SERVER_NAME |                                                   REGION_NAME |  START_KEY
|    END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | -------------------------------------------------------------- | ----------
| ---------- | ----- | ----- | ---------- |
 bds-adh-3.ru-central1.internal,16020,1642400781435 |  people_ages,,1638455640269.6375fc55e8ad8f9948b22cfcc34a4563. |            |
F |     0 |   238 |        1.0 |
 bds-adh-3.ru-central1.internal,16020,1642400781435 | people_ages,F,1638455640269.508ad2a04ec106517eaa8ec6fd01fe83. |          F |
K |     0 |   218 |        1.0 |
 bds-adh-1.ru-central1.internal,16020,1642400781296 | people_ages,K,1638455640269.42f95f9f14c8345d2c9940ea68fc7c30. |          K |
P |     0 |   195 |        1.0 |
 bds-adh-1.ru-central1.internal,16020,1642400781296 | people_ages,P,1638455640269.edf909e0f6785817b4517f044a3ebfd4. |          P |
W |     0 |   254 |        1.0 |
 bds-adh-2.ru-central1.internal,16020,1642400781522 | people_ages,W,1638455640269.bc245646e72b0dcf1a06ca061ed538f4. |          W |
|     0 |    92 |        1.0 |
 5 rows
Took 0.0332 seconds
hbase(main):006:0> split 'people_ages,P,1638455640269', 'S'
Took 0.2193 seconds
hbase(main):007:0> list_regions 'people_ages'
                                        SERVER_NAME |                                                   REGION_NAME |  START_KEY
|    END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | -------------------------------------------------------------- | ----------
| ---------- | ----- | ----- | ---------- |
 bds-adh-3.ru-central1.internal,16020,1642400781435 |  people_ages,,1638455640269.6375fc55e8ad8f9948b22cfcc34a4563. |            |
F |     0 |   238 |        1.0 |
 bds-adh-3.ru-central1.internal,16020,1642400781435 | people_ages,F,1638455640269.508ad2a04ec106517eaa8ec6fd01fe83. |          F |
K |     0 |   218 |        1.0 |
 bds-adh-1.ru-central1.internal,16020,1642400781296 | people_ages,K,1638455640269.42f95f9f14c8345d2c9940ea68fc7c30. |          K |
P |     0 |   195 |        1.0 |
 bds-adh-1.ru-central1.internal,16020,1642400781296 | people_ages,P,1642431183346.3636b23eae41ba162fde908cb245d325. |          P |
S |     0 |     0 |        1.0 |
 bds-adh-1.ru-central1.internal,16020,1642400781296 | people_ages,S,1642431183346.f3a24f3f04bd9d126d8f30fe40e466a9. |          S |
W |     0 |     0 |        1.0 |
 bds-adh-2.ru-central1.internal,16020,1642400781522 | people_ages,W,1638455640269.bc245646e72b0dcf1a06ca061ed538f4. |          W |
|     0 |    92 |        1.0 |
 6 rows
Took 0.0355 seconds
```

Разделение таблицы

```
hbase(main):013:0> list_regions 'BEST_BOOKS'
                                        SERVER_NAME |                                                 REGION_NAME |  START_KEY |
END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | ------------------------------------------------------------ | ---------- |
---------- | ----- | ----- | ---------- |
 bds-adh-2.ru-central1.internal,16020,1642400781522 | BEST_BOOKS,,1638430003108.36ee4f69dc1ab9d4a075091ff9ea3b80. |            |
|     0 |     0 |        1.0 |
 1 rows
Took 0.0159 seconds
hbase(main):016:0> split 'BEST_BOOKS', 'K'
Took 0.1780 seconds
hbase(main):017:0> list_regions 'BEST_BOOKS'
                                        SERVER_NAME |                                                  REGION_NAME |  START_KEY |
END_KEY |  SIZE |   REQ |   LOCALITY |
 -------------------------------------------------- | ------------------------------------------------------------- | ---------- |
---------- | ----- | ----- | ---------- |
 bds-adh-2.ru-central1.internal,16020,1642400781522 |  BEST_BOOKS,,1642431463389.dc0dd6bee8c1b1d7db462a17fbd33673. |            |
K |     0 |     0 |        1.0 |
 bds-adh-2.ru-central1.internal,16020,1642400781522 | BEST_BOOKS,K,1642431463389.d6e6be2ad047bc3f2ea5ed6842f851b0. |          K |
|     0 |     0 |        1.0 |
 2 rows
Took 0.0164 seconds
```

splitormerge_enabled

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Проверяет, включены ли операции `split` или `merge`.

Использование

```
splitormerge_enabled '<operation_type>'
```

Аргументы

Параметр	Описание
operation_type	Возможные значения: `SPLIT` — проверяет, разрешены ли операции split; `MERGE` — проверяет, включены ли операции merge.

Примеры

```
hbase(main):001:0> splitormerge_enabled 'SPLIT'
true
Took 0.3520 seconds
=> 1
hbase(main):002:0> splitormerge_enabled 'MERGE'
true
Took 0.0046 seconds
=> 1
```

splitormerge_switch

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
- Отключение
- Включение

Определение

Включает/отключает операции split или merge. Возвращает предыдущее состояние.

Использование

```
splitormerge_switch '<operation_type>', <new_state>
```

Аргументы

Параметр	Описание
operation_type	Возможные значения: `SPLIT` — split-операции; `MERGE` — merge-операции.
new_state	Логическое значение, определяющее новое состояние для операций split или merge. Значение `true` включает опцию, `false` — выключает

Примеры

Отключение

```
hbase(main):012:0> splitormerge_switch 'SPLIT', false
true
Took 0.0080 seconds
=> 1
hbase(main):013:0> splitormerge_switch 'MERGE', false
true
Took 0.0080 seconds
=> 1
```

Включение

```
hbase(main):014:0> splitormerge_switch 'SPLIT', true
false
Took 0.0085 seconds
=> 1
hbase(main):015:0> splitormerge_switch 'MERGE', true
false
Took 0.0087 seconds
=> 1
```

trace

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
  - Запуск трассировки
  - Остановка трассировки
  - Использование имени этапа

Определение

Запускает/останавливает/проверяет трассировку через HTrace. Тип операции определяется первым аргументом. Возвращает `true`, если трассировка запущена, иначе возвращает `false`.

Использование

```
trace '<trace_type>'[, '<trace_span_name>']
```

Аргументы

Параметр	Описание
trace_type	Возможные значения: - `start` — запускает трассировку. Возвращает `true` в случае успеха. - `stop` — останавливает текущий этап трассировки. Возвращает `false` в случае успеха. - `status` — показывает, запущена ли трассировка.
trace_span_name	Опциональное имя для нового этапа трассировки, которое можно задать при запуске трассировки. Имя по умолчанию — `HBaseShell`.

Примеры

Запуск трассировки

```
hbase(main):006:0> trace 'status'
Took 0.0004 seconds
=> false
hbase(main):007:0> trace 'start'
Took 0.1438 seconds
=> true
hbase(main):008:0> trace 'status'
Took 0.0002 seconds
=> true
```

Остановка трассировки

```
hbase(main):009:0> trace 'stop'
Took 0.0003 seconds
=> false
hbase(main):010:0> trace 'status'
Took 0.0001 seconds
=> false
```

Использование имени этапа

```
hbase(main):024:0> trace 'start', 'my'
Took 0.0003 seconds
=> true
hbase(main):025:0> trace 'status', 'my'
Took 0.0002 seconds
=> true
hbase(main):026:0> trace 'stop', 'my'
Took 0.0002 seconds
=> false
hbase(main):027:0> trace 'status', 'my'
Took 0.0002 seconds
=> false
```

unassign

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Отменяет назначение определенного региона. Команда закрывает регион в его текущем местоположении, а затем вновь открывает его. Может применяться только к назначенным регионам.

ВНИМАНИЕ

Эта команда предназначена только для использования экспертами. В случае двойных назначений используйте `hbck -fix` для разрешения конфликтов.

Использование

```
unassign '<region_name>' | '<encoded_region_name>'[,
         <force_unassign>]
```

Аргументы

Параметр	Описание
region_name	Имя региона без закодированной части. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то `region_name` — это `wordcount2,m,1637833871529`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
encoded_region_name	Закодированная часть названия региона. Например, если полное название региона `wordcount2,m,1637833871529.2ff3683a90c4106432c540ddd0f57cba.`, то закодированная часть — это `2ff3683a90c4106432c540ddd0f57cba`. Вы можете получить полное имя региона и его закодированную часть, используя вывод команды list_regions (поле `REGION_NAME`)
force_unassign	Значение `true` принудительно отменяет назначение, то есть очищает все состояния в памяти в Master перед повторным назначением (только для экспертного пользования)

Примеры

```
hbase(main):011:0> unassign 'wordcount2,m,1637833871529'
Took 1.1734 seconds
hbase(main):013:0> assign 'wordcount2,m,1637833871529'
Took 1.1425 seconds
hbase(main):014:0> unassign '2ff3683a90c4106432c540ddd0f57cba'
Took 1.1506 seconds
hbase(main):015:0> assign '2ff3683a90c4106432c540ddd0f57cba'
Took 1.1368 seconds
```

wal_roll

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Сбрасывает WAL-writer, т. е. начинает записывать логи в новый файл. Требует имя Region Server.

Использование

```
wal_roll '<region_server_name>,<port_number>,<start_code>'
```

Аргументы

Параметр	Описание
region_server_name	Имя Region Server
port_number	Номер порта
start_code	Стартовый код сервера

TIP

Для получения `<server_name>`, `<port_number>` и `<start_code>` используйте команду status 'simple'.

Примеры

```
hbase(main):004:0> wal_roll 'bds-adh-1.ru-central1.internal,16020,1642400781296'
Took 0.0119 seconds
```

zk_dump

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Выводит состояние дампа кластера HBase, как его видит ZooKeeper.

Использование

```
zk_dump
```

Примеры

```
hbase(main):001:0> zk_dump
HBase is rooted at /hbase
Active master address: bds-adh-2.ru-central1.internal,16000,1642400778136
Backup master addresses:
Region server holding hbase:meta: bds-adh-2.ru-central1.internal,16020,164240078
1522
Region servers:
 bds-adh-3.ru-central1.internal,16020,1642400781435
 bds-adh-2.ru-central1.internal,16020,1642400781522
 bds-adh-1.ru-central1.internal,16020,1642400781296
/hbase/replication:
/hbase/replication/peers:
/hbase/replication/rs:
/hbase/replication/rs/bds-adh-1.ru-central1.internal,16020,1642400781296:
/hbase/replication/rs/bds-adh-2.ru-central1.internal,16020,1642400781522:
/hbase/replication/rs/bds-adh-3.ru-central1.internal,16020,1642400781435:
Quorum Server Statistics:
 bds-adh-1.ru-central1.internal:2181
  Zookeeper version: 3.4.14-f42ef4b166f1e5c63117c75aed651589a0cce1b2, built on 0
8/06/2019 14:49 GMT
  Clients:
   /10.92.6.9:54898[1](queued=0,recved=655,sent=677)
   /10.92.6.217:50280[0](queued=0,recved=1,sent=0)
   /10.92.6.217:43832[1](queued=0,recved=687,sent=710)

  Latency min/avg/max: 0/0/21
  Received: 1531
  Sent: 1575
  Connections: 3
  Outstanding: 0
  Zxid: 0x2600000118
  Mode: follower
  Node count: 283
 bds-adh-2.ru-central1.internal:2181
  Zookeeper version: 3.4.14-f42ef4b166f1e5c63117c75aed651589a0cce1b2, built on 0
8/06/2019 14:49 GMT
  Clients:
   /10.92.6.217:39292[0](queued=0,recved=1,sent=0)
   /10.92.6.217:60694[1](queued=0,recved=2258,sent=2258)
   /10.92.6.217:39256[1](queued=0,recved=2,sent=2)
   /10.92.6.217:39288[1](queued=0,recved=16,sent=16)
   /10.92.6.9:59798[1](queued=0,recved=1581,sent=1619)
   /10.92.6.90:42248[1](queued=0,recved=651,sent=671)

  Latency min/avg/max: 0/0/17
  Received: 4680
  Sent: 4737
  Connections: 6
  Outstanding: 0
  Zxid: 0x2600000118
  Mode: follower
  Node count: 283
 bds-adh-3.ru-central1.internal:2181
  Zookeeper version: 3.4.14-f42ef4b166f1e5c63117c75aed651589a0cce1b2, built on 0
8/06/2019 14:49 GMT
  Clients:
   /10.92.6.9:53374[1](queued=0,recved=632,sent=632)
   /10.92.6.90:53890[1](queued=0,recved=2260,sent=2260)
   /10.92.6.217:40910[0](queued=0,recved=1,sent=0)

  Latency min/avg/max: 0/0/12
  Received: 3060
  Sent: 3059
  Connections: 3
  Outstanding: 0
  Zxid: 0x2600000118
  Mode: leader
  Node count: 283
  Proposal sizes last/min/max: 36/32/842
Took 0.1005 seconds
```

add_peer

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

- Репликация между кластерами HBase
- Кастомная replication endpoint

Определение

Добавляет новре отношение репликации (peer) между текущим кластером HBase и либо другим кластером HBase, либо настраиваемой конечной точкой репликации на основе предопределенного класса.

Следует помнить о следующих особенностях и ограничениях:

- Чтобы настроить репликацию между двумя кластерами, обозначьте второй кластер, используя входной параметр `CLUSTER_KEY`. В противном случае используйте параметр `ENDPOINT_CLASSNAME`, чтобы определить кастомную replication endpoint. Следует указать либо `CLUSTER_KEY`, либо `ENDPOINT_CLASSNAME`. Если `ENDPOINT_CLASSNAME` задан, то `CLUSTER_KEY` является необязательным и должен указываться только в том случае, если этого требует конкретная настраиваемая replication endpoint.

- По умолчанию реплицируются только выбранные пространства имен и таблицы. Они должны быть определены параметрами `NAMESPACES` и `TABLE_CFS`. Но также можно настроить репликацию для всех таблиц, установив флаг `REPLICATE_ALL` с помощью команды set_peer_replicate_all.

- Независимо от значения флага `REPLICATE_ALL`, реплицируются только таблицы с установленным флагом `REPLICATION_SCOPE`. Чтобы установить этот флаг и сделать таблицу реплицируемой, используйте команду enable_table_replication. Чтобы сбросить флаг и отключить репликацию для таблицы, используйте команду disable_table_replication. Обратите внимание, что процесс репликации не может быть запущен для таблиц с отключенной репликацией, даже если эти таблицы выбраны в конфигурации узла.

- Входные параметры `NAMESPACES` и `TABLE_CFS` связаны между собой. Нельзя устанавливать таблицы из пространств имен, которые уже определены параметром `NAMESPACES`.

- Вы можете установить пространства имен репликации для peer позже с помощью команд append_peer_namespaces или set_peer_namespaces. Чтобы удалить пространства имен, выполните команду remove_peer_namespaces.

- Вы можете установить таблицы/семейства столбцов репликации для peer позже с помощью команд append_peer_tableCFs или set_peer_tableCFs. Чтобы удалить таблицы/семейства столбцов, используется команда remove_peer_tableCFs.

> **ПРИМЕЧАНИЕ**
>
> Больше информации о настройке репликации доступно в документации HBase.

Использование

- Репликация между кластерами HBase:

```
add_peer '<peer_id>',
         CLUSTER_KEY => '<cluster_key>'[,
         STATE => '<state>'][,
         NAMESPACES => <namespaces_array[]>][,
         TABLE_CFS => {'[<namespace_name>:]<table_name>' => <column_families_array[]>[,...]}]
```

- Кастомная replication endpoint:

```
add_peer '<peer_id>',
         ENDPOINT_CLASSNAME => '<class_name>'[,
         CLUSTER_KEY => '<cluster_key>'][,
         STATE => '<state>'][,
         DATA => {'<key_name>' => <key_value>, [...]}][,
         CONFIG => {'<config_name>' => '<config_value>', [...]}][,
         NAMESPACES => <namespaces_array[]>][,
         TABLE_CFS => {'[<namespace_name>:]<table_name>' => <column_families_array[]>[,...]}]
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
cluster_key	Полный адрес второго кластера HBase в репликации. Обязательно в случае настройки репликации между двумя кластерами. Принимает следующий формат: `<hbase.zookeeper.quorum>:<hbase.zookeeper.property.clientPort>:<zookeeper.znode.parent>`. Все аргументы можно получить через Master UI (страница ***HBase configuration***)
state	Состояние peer репликации. Возможные значения: `ENABLED` и `DISABLED`. Значение по умолчанию — `ENABLED`. Позже можно изменить с помощью команд disable_peer и enable_peer.
namespaces_array	Массив строк, указанных в формате `'<namespace_name>'`, где `<namespace_name>` — имя пространства имен
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_families_array	Массив строк, указанных в формате `'<column_family>'`, где `<column_family>` — имя семейства столбцов
class_name	Имя кастомного класса. Обязательно, если репликация настраивается для кастомной replication endpoint. Пример: `org.apache.hadoop.hbase.MyReplicationEndpoint`
key_name	Имя peer data-параметра
key_value	Значение peer data-параметра
config_name	Имя конфигурационного peer-параметра
config_value	Значение конфигурационного peer-параметра

Примеры

Репликация между кластерами HBase

```
hbase(main):001:0> add_peer '1', CLUSTER_KEY => "serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-
central1.internal:2181:/hbase", TABLE_CFS => { "articles" => []}
Took 0.3667 seconds
```

Кастомная replication endpoint

```
hbase(main):018:0> add_peer '2', ENDPOINT_CLASSNAME => 'test', STATE => 'DISABLED', DATA => {'key1' => 'value1'}, CONFIG =>
{'config1' => 'value1'}, NAMESPACES => ['ns1'], TABLE_CFS => { "articles" => []}
Took 0.0306 seconds
```

append_peer_namespaces

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Успешное добавление пространства имен
  - Неудачное добавление пространства имен

Определение

Добавляет реплицируемые пространства имен к указанному отношению репликации (peer). Следует помнить о следующих ограничениях:

- Флаг `REPLICATE_ALL` должен иметь значение `false` в конфигурации отношения репликации (set_peer_replicate_all).
- Установка пространства имен означает, что все таблицы в этом пространстве имен с флагом `REPLICATION_SCOPE` != 0 реплицируются. Если конфигурация peer уже содержит какую-либо таблицу (добавленную через append_peer_tableCFs или set_peer_tableCFs), то установка ее пространства имен с помощью этой команды не допускается.

ПРИМЕЧАНИЕ

Эта команда не удаляет ранее определенные пространства имен из конфигурации peer (в отличие от set_peer_namespaces).

Использование

```
append_peer_namespaces '<peer_id>', <namespaces_array[]>
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
namespaces_array	Массив строк, указанных в формате `'<namespace_name>'`, где `<namespace_name>` — имя пространства имен

Примеры

Успешное добавление пространства имен

Peer без пространств имен:

```
hbase(main):019:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
default.articles 0
1 row(s)
Took 0.0119 seconds
=> #<Java::JavaUtil::ArrayList:0xe886caf>
hbase(main):022:0> append_peer_namespaces '1', ['ns1']
Took 0.0250 seconds
hbase(main):025:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1 default.articles 0
1 row(s)
Took 0.0063 seconds
=> #<Java::JavaUtil::ArrayList:0x6a902015>
```

Peer с пространствами имен:

```
hbase(main):031:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1 default.articles 0
1 row(s)
Took 0.0063 seconds
=> #<Java::JavaUtil::ArrayList:0x6a902015>
hbase(main):032:0> append_peer_namespaces '1', ['ns2']
Took 0.0114 seconds
hbase(main):034:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1;ns2 default.articles 0
1 row(s)
Took 0.0084 seconds
=> #<Java::JavaUtil::ArrayList:0x2c1ea7be>
```

Неудачное добавление пространства имен

```
hbase(main):053:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED false
ns1 default.articles 0
1 row(s)
Took 0.0113 seconds
=> #<Java::JavaUtil::ArrayList:0x6dde1bf5>
hbase(main):054:0> append_peer_namespaces '1', ['default']

ERROR: java.io.IOException: Table-cfs articles is conflict with namespaces default in peer config
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:472)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)
Caused by: org.apache.hadoop.hbase.replication.ReplicationException: Table-cfs articles is conflict with namespaces default in
peer config
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.
checkNamespacesAndTableCfsConfigConflict(ReplicationManager.java:181)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.checkPeerConfig(ReplicationManager.java:148)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.updatePeerConfig(ReplicationManager.java:103)
        at org.apache.hadoop.hbase.master.HMaster.updateReplicationPeerConfig(HMaster.java:3535)
        at org.apache.hadoop.hbase.master.MasterRpcServices.updateReplicationPeerConfig(MasterRpcServices.java:1943)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        ... 3 more

  Append some namespaces to be replicable for the specified peer.

  Set a namespace in the peer config means that all tables in this
  namespace (with replication_scope != 0 ) will be replicated.

  Examples:

    # append ns1,ns2 to be replicable for peer '2'.
    hbase> append_peer_namespaces '2', ["ns1", "ns2"]


Took 8.1496 seconds
```

append_peer_tableCFs

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
  Успешное добавление таблицы
  Неудачное добавление таблицы

Определение

Добавляет реплицируемые таблицы/семейства колонок к указанному отношению репликации (peer). Следует помнить о следующих ограничениях:

- Флаг `REPLICATE_ALL` должен иметь значение `false` в конфигурации отношения репликации (set_peer_replicate_all).
- Если конфигурация peer уже содержит некоторое пространство имен (добавленное через append_peer_namespaces или set_peer_namespaces), то установка его таблиц с помощью этой команды не допускается.

ПРИМЕЧАНИЕ

Эта команда не удаляет ранее определенные таблицы/семейства колонок из настройки отношения репликации (в отличие от set_peer_namespaces).

Использование

```
append_peer_tableCFs '<peer_id>',
                     {'[<namespace_name>:]<table_name>' => <column_families_array[]>[,...]}
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_families_array	Массив строк, указанных в формате `'<column_family>'`, где `<column_family>` — имя семейства столбцов

Примеры

Успешное добавление таблицы

Без семейств столбцов

```
hbase(main):037:0> show_peer_tableCFs '1'
default.articles
Took 0.0077 seconds
=> "default.articles"
hbase(main):042:0> append_peer_tableCFs '1', {'table10' => []}
Took 0.0079 seconds
hbase(main):043:0> show_peer_tableCFs '1'
default.articles;default.table10
Took 0.0033 seconds
=> "default.articles;default.table10"
```

С семейством столбцов

```
hbase(main):045:0> append_peer_tableCFs '1', {'temp1' => ['cf1']}
Took 0.0087 seconds
hbase(main):046:0> show_peer_tableCFs '1'
default.articles;default.table10;default.temp1:cf1
Took 0.0036 seconds
=> "default.articles;default.table10;default.temp1:cf1"
```

Неудачное добавление таблицы

```
hbase(main):057:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED false
ns1 default.articles 0
1 row(s)
Took 0.0112 seconds
=> #<Java::JavaUtil::ArrayList:0x1ba7db2a>
hbase(main):058:0> append_peer_tableCFs '1', {'ns1:temp2' => []}

ERROR: java.io.IOException: Table-cfs ns1:temp2 is conflict with namespaces ns1 in peer config
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:472)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)
Caused by: org.apache.hadoop.hbase.replication.ReplicationException: Table-cfs ns1:temp2 is conflict with namespaces ns1 in peer
config
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.
checkNamespacesAndTableCfsConfigConflict(ReplicationManager.java:181)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.checkPeerConfig(ReplicationManager.java:148)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.updatePeerConfig(ReplicationManager.java:103)
        at org.apache.hadoop.hbase.master.HMaster.updateReplicationPeerConfig(HMaster.java:3535)
        at org.apache.hadoop.hbase.master.MasterRpcServices.updateReplicationPeerConfig(MasterRpcServices.java:1943)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        ... 3 more

Append a replicable table-cf config for the specified peer
Examples:

  # append a table / table-cf to be replicable for a peer
  hbase> append_peer_tableCFs '2', { "ns1:table4" => ["cfA", "cfB"]}


Took 8.1481 seconds
```

disable_peer

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Отключает указанное отношение репликации (peer). HBase больше не будет направлять поток реплицируемых данных в целевой кластер (указанный в peer), но по-прежнему будет отслеживать все новые WAL, необходимые для репликации, если и когда узел будет повторно включен. При включении или отключении репликации WAL сохраняются до тех пор, пока существует peer.

РЕКОМЕНДАЦИЯ

Если вы хотите навсегда остановить поток репликации, используйте команду remove_peer.

Использование

```
disable_peer '<peer_id>'
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов

Примеры

```
hbase(main):013:0> disable_peer '1'
Took 0.0179 seconds
```

disable_table_replication

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Отключает репликацию для указанной таблицы. Флаг `REPLICATION_SCOPE` устанавливается равным `0` для всех семейств столбцов таблицы. Кроме того, таблица более не появляется в выводе команды list_replicated_tables.

Использование

```
disable_table_replication '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):010:0> disable_table_replication 'articles'
Replication of table 'articles' successfully disabled.
Took 1.4515 seconds
hbase(main):069:0> list_replicated_tables
TABLE:COLUMNFAMILY                                  ReplicationType
0 row(s)
Took 0.0066 seconds
hbase(main):070:0> describe 'articles'
Table articles is ENABLED
articles
COLUMN FAMILIES DESCRIPTION
{NAME => 'basic', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => '
false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false'
, IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE
=> 'true', BLOCKSIZE => '6553
6'}
{NAME => 'tags', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'f
alse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
 IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536
'}
2 row(s)
Took 0.0208 seconds
```

enable_peer

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Делает доступными ранее отключенные отношения репликации (peer). После выполнения команды поток реплицируемых данных в целевой кластер (указанный в peer) начинается с точки, в которой peer был disabled.

Использование

```
enable_peer '<peer_id>'
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов

Примеры

```
hbase(main):023:0> enable_peer '1'
Took 0.0148 seconds
```

enable_table_replication

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Включает репликацию для указанной таблицы. Флаг `REPLICATION_SCOPE` становится равен `1` для всех семейств столбцов таблицы. Кроме того, таблица появляется в выводе команды list_replicated_tables.

Использование

```
enable_table_replication '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):021:0> enable_table_replication 'articles'
The replication of table 'articles' successfully enabled
Took 1.2512 seconds
hbase(main):064:0> list_replicated_tables
TABLE:COLUMNFAMILY                                    ReplicationType
 articles:basic                                       GLOBAL
 articles:tags                                        GLOBAL
2 row(s)
Took 0.0381 seconds
hbase(main):065:0> describe 'articles'
Table articles is ENABLED
articles
COLUMN FAMILIES DESCRIPTION
{NAME => 'basic', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => '
false', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '1', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false'
, IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE
=> 'true', BLOCKSIZE => '6553
6'}
{NAME => 'tags', VERSIONS => '5', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', KEEP_DELETED_CELLS =>
'FALSE', CACHE_DATA_ON_WRITE => 'f
alse', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '1', BLOOMFILTER => 'ROW',
CACHE_INDEX_ON_WRITE => 'false',
 IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE =>
'true', BLOCKSIZE => '65536
'}
2 row(s)
Took 0.1300 seconds
```

get_peer_config

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Репликация между кластерами HBase
  - Кастомная replication endpoint

Определение

Выводит список параметров конфигурации определенного отношения репликации (peer).

Использование

```
get_peer_config '<peer_id>'
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов

Примеры

Репликация между кластерами HBase

```
hbase(main):002:0> get_peer_config '1'
 Cluster Key          serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.i
                      nternal,serov-adh-3.ru-central1.internal:2181:/hbase
Took 0.0017 seconds
=> #<Java::OrgApacheHadoopHbaseReplication::ReplicationPeerConfig:0x6dc2279c>
```

Кастомная replication endpoint

```
hbase(main):019:0> get_peer_config '2'
 Cluster Key                         bds-adh-1.ru-central1.internal,bds-adh-2.ru-central1.internal,bds-adh-3.ru-central1.
internal:2181:/hbase
 Replication Endpoint                test
 config1                             value1
Took 0.0008 seconds
=> #<Java::OrgApacheHadoopHbaseReplication::ReplicationPeerConfig:0x4a4ff5ed>
```

list_peers

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит список всех отношений репликации (peers) между исходным и целевым кластерами HBase или пользовательскими replication endpoints.

Использование

```
list_peers
```

Выходные параметры

Параметр	Описание
PEER_ID	Уникальный идентификатор отношения репликации
CLUSTER_KEY	Полный адрес второго кластера HBase в отношении репликации. Имеет следующий формат на основе параметров конфигурации HBase: `<hbase.zookeeper.quorum>:<hbase.zookeeper.property.clientPort>:<zookeeper.znode.parent>`
ENDPOINT_CLASSNAME	Имя пользовательского класса. Заполняется, если репликация настроена для кастомной replication endpoint
STATE	Состояние узла репликации. Можно изменить с помощью команд disable_peer и enable_peer.
REPLICATE_ALL	Логическое значение флага `REPLICATE_ALL`. Можно изменить с помощью команды set_peer_replicate_all
NAMESPACES	Пространства имен, включенные или исключенные из peer-конфигурации. Исключенные объекты отображаются с префиксом `!` (например, `!ns1;ns2`)
TABLE_CFS	Таблицы/семейства столбцов, включенные или исключенные из peer-конфигурации. Исключенные объекты отображаются с префиксом `!` (например, `!default.articles`)
BANDWIDTH	Исходная пропускная способность (на каждый Region Server). Можно изменить с помощью команды set_peer_bandwidth.

Примеры

```
hbase(main):033:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
default.articles 0
1 row(s)
Took 0.0060 seconds
=> #<Java::JavaUtil::ArrayList:0x41059616>
```

list_peer_configs

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Выводит параметры конфигурации всех отношений репликации (peers), которые определены для текущего кластера HBase.

Использование

```
list_peer_configs
```

Примеры

```
hbase(main):011:0> list_peer_configs
 PeerId                                  1
 Cluster Key                             serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase

Took 0.0076 seconds
=> {"1"=>#<Java::OrgApacheHadoopHbaseReplication::ReplicationPeerConfig:0x37cf91d8>}
```

list_replicated_tables

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

- Без регулярных выражений
- С регулярными выражениями

Определение

Выводит список всех таблиц и семейств столбцов, реплицированных из текущего кластера HBase.

Использование

```
list_replicated_tables ['<table_regex>']
```

Аргументы

Параметр	Описание
table_regex	Регулярное выражение для поиска таблиц

Примеры

Без регулярных выражений

```
hbase(main):017:0> list_replicated_tables
TABLE:COLUMNFAMILY                                  ReplicationType
 articles:basic                                     GLOBAL
 articles:tags                                      GLOBAL
2 row(s)
Took 0.0130 seconds
```

С регулярными выражениями

```
hbase(main):031:0> list_replicated_tables 'art.*'
TABLE:COLUMNFAMILY                                  ReplicationType
 articles:basic                                     GLOBAL
 articles:tags                                      GLOBAL
2 row(s)
Took 0.0102 seconds
```

remove_peer

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Останавливает поток репликации для указанного отношения (peer), затем отключает и удаляет его. Также очищает всю сохраненную метаинформацию об этом peer.

Использование

```
remove_peer '<peer_id>'
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов

Примеры

```
hbase(main):002:0> remove_peer '1'
Took 0.0281 seconds
hbase(main):003:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
0 row(s)
Took 0.0049 seconds
=> #<Java::JavaUtil::ArrayList:0x35329a05>
```

remove_peer_namespaces

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Удаляет пространства имен из конфигурации отношения репликации (peer).

Использование

```
remove_peer_namespaces '<peer_id>', <namespaces_array[]>
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
namespaces_array	Массив строк, указанных в формате `'<namespace_name>'`, где `<namespace_name>` — имя пространства имен

Примеры

```
hbase(main):022:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1 default.articles 0
1 row(s)
Took 0.0090 seconds
=> #<Java::JavaUtil::ArrayList:0x1e5b33e5>
hbase(main):002:0> remove_peer_namespaces '1', ['ns1']
Took 0.0269 seconds
hbase(main):003:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
default.articles 0
1 row(s)
Took 0.0060 seconds
=> #<Java::JavaUtil::ArrayList:0x7645b7d>
```

remove_peer_tableCFs

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Удаляет таблицы/семейства столбцов из конфигурации отношения репликации (peer).

Использование

```
remove_peer_tableCFs '<peer_id>',
                     {'[<namespace_name>:]<table_name>' => <column_families_array[]>[,...]}
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_families_array	Массив строк, указанных в формате `'<column_family>'`, где `<column_family>` — имя семейства столбцов

Примеры

```
hbase(main):014:0> show_peer_tableCFs '1'
default.articles
Took 0.0066 seconds
=> "default.articles"
hbase(main):014:0> remove_peer_tableCFs '1', {'articles' => []}
Took 0.0306 seconds
hbase(main):017:0> show_peer_tableCFs '1'

Took 0.0037 seconds
```

set_peer_bandwidth

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Установка пропускной способности, равной 2 МБ
  - Сброс пропускной способности

Определение

Задает исходную пропускную способность (на каждый Region Server) для указанного отношения репликации (peer).

РЕКОМЕНДАЦИЯ

Чтобы вернуться к значению пропускной способности по умолчанию, настроенной на стороне сервера, необходимо вызвать эту команду без аргументов.

Использование

```
set_peer_bandwidth '<peer_id>'[, <bandwidth_value>]
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
bandwidth_value	Желаемое значение пропускной способности (в байтах)

Примеры

Установка пропускной способности, равной 2 МБ

```
hbase(main):049:0> set_peer_bandwidth '1', 2097152
Took 0.0241 seconds
hbase(main):050:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1;ns2 default.articles;default.table10;default.temp1:cf1 2097152
1 row(s)
Took 0.0069 seconds
=> #<Java::JavaUtil::ArrayList:0x780c0>
```

Сброс пропускной способности

```
hbase(main):051:0> set_peer_bandwidth '1'
Took 0.0096 seconds
hbase(main):052:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1;ns2 default.articles;default.table10;default.temp1:cf1 0
1 row(s)
Took 0.0080 seconds
=> #<Java::JavaUtil::ArrayList:0x4ce18cec>
```

set_peer_exclude_namespaces

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Добавление исключенных пространств имен
  - Удаление исключенных пространств имен

Определение

Определяет нереплицируемые пространства имен для отношения репликации (peer). Следует помнить о следующих ограничениях:

- Флаг `REPLICATE_ALL` должен иметь значение `true` в конфигурации отношения репликации (set_peer_replicate_all).
- Установка исключенного пространства имен означает, что все таблицы в этом пространстве имен не реплицируются. Если конфигурация peer уже содержит какую-либо исключенную таблицу (добавленную через set_peer_exclude_tableCFs), то установка ее пространства имен с помощью этой команды не допускается.
- Если нужно очистить исключенные пространства имен, необходимо запустить текущую команду с пустым массивом в качестве второго входного параметра. Не используйте команду remove_peer_namespaces.

Использование

```
set_peer_exclude_namespaces '<peer_id>', <namespaces_array[]>
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
namespaces_array	Массив строк, указанных в формате `'<namespace_name>'`, где `<namespace_name>` — имя пространства имен

Примеры

Добавление исключенных пространств имен

```
hbase(main):013:0> set_peer_exclude_namespaces '1', ['ns1', 'ns2']
Took 0.0135 seconds
hbase(main):014:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED true
!ns1;ns2  0
1 row(s)
Took 0.0148 seconds
=> #<Java::JavaUtil::ArrayList:0x7d977a20>
```

Удаление исключенных пространств имен

```
hbase(main):017:0> set_peer_exclude_namespaces '1', []
Took 0.0095 seconds
hbase(main):018:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED true
0
1 row(s)
Took 0.0112 seconds
=> #<Java::JavaUtil::ArrayList:0x71a2df1>
```

set_peer_exclude_tableCFs

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Исключение таблиц/семейств столбцов
  - Удаление исключенных таблиц/семейств столбцов

Определение

Добавляет все нереплицируемые таблицы/семейства колонок к указанному отношению репликации (peer). Следует помнить о следующих ограничениях:

- Флаг `REPLICATE_ALL` должен иметь значение `true` в конфигурации отношения репликации (set_peer_replicate_all).
- Если конфигурация peer уже содержит некоторое исключенное пространство имен (добавленное через set_peer_exclude_namespaces), то установка его таблиц с помощью этой команды не допускается.
- При необходимости очистить исключенные таблицы/семейства столбцов нужно запустить текущую команду с пустым словарем в качестве второго входного параметра. Не используйте команду remove_peer_tableCFs.

Использование

```
set_peer_exclude_tableCFs '<peer_id>',
                          {'[<namespace_name>:]<table_name>' => <column_families_array[]>[,...]}
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_families_array	Массив строк, указанных в формате `'<column_family>'`, где `<column_family>` — имя семейства столбцов

Примеры

Исключение таблиц/семейств столбцов

```
hbase(main):022:0> set_peer_exclude_tableCFs '1', {'articles' => []}
Took 0.0090 seconds
hbase(main):023:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED true
!default.articles 0
1 row(s)
Took 0.0088 seconds
=> #<Java::JavaUtil::ArrayList:0x1d585fb>
```

Удаление исключенных таблиц/семейств столбцов

```
hbase(main):027:0> set_peer_exclude_tableCFs '1', {}
Took 0.0085 seconds
hbase(main):028:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED true
0
1 row(s)
Took 0.0057 seconds
=> #<Java::JavaUtil::ArrayList:0x26c59348>
```

set_peer_namespaces

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Успешная установка пространства имен
  - Неудачная настройка пространства имен

Определение

Добавляет все реплицируемые пространства имен к указанному отношению репликации (peer). Следует помнить о следующих ограничениях:

- Флаг `REPLICATE_ALL` должен иметь значение `false` в конфигурации отношения репликации (set_peer_replicate_all).
- Установка пространства имен означает, что все таблицы в этом пространстве имен с флагом `REPLICATION_SCOPE` != 0 реплицируются. Если конфигурация peer уже содержит какую-либо таблицу (добавленную через append_peer_tableCFs или set_peer_tableCFs), то установка ее пространства имен с помощью этой команды не допускается.
- При необходимости очистить включенные пространства имен нужно запустить текущую команду с пустым массивом в качестве второго входного параметра. Вы также можете использовать команду remove_peer_namespaces.

ВНИМАНИЕ

Эта команда удаляет ранее определенные пространства имен из конфигурации peer (в отличие от append_peer_namespaces).

Использование

```
set_peer_namespaces '<peer_id>', <namespaces_array[]>
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
namespaces_array	Массив строк, указанных в формате `'<namespace_name>'`, где `<namespace_name>` — имя пространства имен

Примеры

Успешная установка пространства имен

```
hbase(main):054:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1;ns2 default.articles;default.table10;default.temp1:cf1 0
1 row(s)
Took 0.0090 seconds
=> #<Java::JavaUtil::ArrayList:0x51c008fd>
hbase(main):055:0> set_peer_namespaces '1', ['ns1']
Took 0.0130 seconds
hbase(main):056:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1 default.articles;default.table10;default.temp1:cf1 0
1 row(s)
Took 0.0084 seconds
=> #<Java::JavaUtil::ArrayList:0x3c98981e>
```

Неудачная настройка пространства имен

```
hbase(main):051:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED false
ns1 default.articles 0
1 row(s)
Took 0.0081 seconds
=> #<Java::JavaUtil::ArrayList:0x126af200>
hbase(main):052:0> set_peer_namespaces '1', ['default']

ERROR: java.io.IOException: Table-cfs articles is conflict with namespaces default in peer config
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:472)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)
Caused by: org.apache.hadoop.hbase.replication.ReplicationException: Table-cfs articles is conflict with namespaces default in
peer config
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.
checkNamespacesAndTableCfsConfigConflict(ReplicationManager.java:181)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.checkPeerConfig(ReplicationManager.java:148)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.updatePeerConfig(ReplicationManager.java:103)
        at org.apache.hadoop.hbase.master.HMaster.updateReplicationPeerConfig(HMaster.java:3535)
        at org.apache.hadoop.hbase.master.MasterRpcServices.updateReplicationPeerConfig(MasterRpcServices.java:1943)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        ... 3 more

  Set the replicable namespaces config for the specified peer.

  1. The replicate_all flag need to be false when set the replicable namespaces.
  2. Set a namespace in the peer config means that all tables in this namespace
     will be replicated to the peer cluster. If peer config already has a table,
     then not allow set this table's namespace to the peer config.

  Examples:

    # set namespaces config is null, then the table-cfs config decide
    # which table to be replicated.
    hbase> set_peer_namespaces '1', []
    # set namespaces to be replicable for a peer.
    # set a namespace in the peer config means that all tables in this
    # namespace (with replication_scope != 0 ) will be replicated.
    hbase> set_peer_namespaces '2', ["ns1", "ns2"]


Took 8.1730 seconds
```

set_peer_replicate_all

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

- Установка true
- Установка false

Определение

Устанавливает флаг `REPLICATE_ALL` в значение `true` или `false` для указанного отношения репликации (peer). Следует помнить о следующих ограничениях:

- Перед установкой флага `REPLICATE_ALL` в значение `true` необходимо удалить ранее добавленные пространства имен и таблицы/семейства столбцов из конфигурации отношения репликации. Для этих целей используются команды remove_peer_namespaces и remove_peer_tableCFs. В противном случае будут выводиться ошибки.
- Если флаг `REPLICATE_ALL` равен `true`, то реплицируются все пользовательские таблицы с флагом `REPLICATION_SCOPE` != 0. Также можно использовать команды set_peer_exclude_namespaces и set_peer_exclude_tableCFs, чтобы исключить некоторые пространства имен и таблицы/семейства столбцов из процесса репликации.
- Перед установкой флага `REPLICATE_ALL` в значение `false` необходимо удалить ранее исключенные пространства имен и таблицы/семейства столбцов из конфигурации отношения репликации. Для этой цели используются команды set_peer_exclude_namespaces и set_peer_exclude_tableCFs с пустым вторым параметром. В противном случае будут выводиться ошибки.
- Если флаг `REPLICATE_ALL` равен `false`, реплицируются только выбранные таблицы. Чтобы выбрать реплицируемые пространства имен, используйте команды set_peer_namespaces и append_peer_namespaces. Чтобы выбрать реплицируемые таблицы семейства столбцов, используйте set_peer_tableCFs и append_peer_tableCFs.

Использование

```
set_peer_replicate_all <flag_value>
```

Аргументы

Параметр	Описание
flag_value	Новое значение флага `REPLICATE_ALL`

Примеры

Установка true

```
hbase(main):039:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED false
ns1;ns2 default.articles 0
1 row(s)
Took 0.0108 seconds
=> #<Java::JavaUtil::ArrayList:0x2633d09c>
hbase(main):040:0> set_peer_replicate_all '1', true

ERROR: java.io.IOException: Need clean namespaces or table-cfs config firstly when replicate_all flag is true
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:472)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)
Caused by: org.apache.hadoop.hbase.replication.ReplicationException: Need clean namespaces or table-cfs config firstly when
replicate_all flag is true
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.checkPeerConfig(ReplicationManager.java:134)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.updatePeerConfig(ReplicationManager.java:103)
        at org.apache.hadoop.hbase.master.HMaster.updateReplicationPeerConfig(HMaster.java:3535)
        at org.apache.hadoop.hbase.master.MasterRpcServices.updateReplicationPeerConfig(MasterRpcServices.java:1943)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        ... 3 more

  Set the replicate_all flag to true or false for the specified peer.

  If replicate_all flag is true, then all user tables (REPLICATION_SCOPE != 0)
  will be replicate to peer cluster. But you can use 'set_peer_exclude_namespaces'
  to set which namespaces can't be replicated to peer cluster. And you can use
  'set_peer_exclude_tableCFs' to set which tables can't be replicated to peer
  cluster.

  If replicate_all flag is false, then all user tables cannot be replicate to
  peer cluster. Then you can use 'set_peer_namespaces' or 'append_peer_namespaces'
  to set which namespaces will be replicated to peer cluster. And you can use
  'set_peer_tableCFs' or 'append_peer_tableCFs' to set which tables will be
  replicated to peer cluster.

  Notice: When you want to change a peer's replicate_all flag from false to true,
          you need clean the peer's NAMESPACES and TABLECFS config firstly.
          When you want to change a peer's replicate_all flag from true to false,
          you need clean the peer's EXCLUDE_NAMESPACES and EXCLUDE_TABLECFS
          config firstly.

  Examples:

    # set replicate_all flag to true
    hbase> set_peer_replicate_all '1', true
    # set replicate_all flag to false
    hbase> set_peer_replicate_all '1', false

Took 8.1956 seconds
hbase(main):042:0> remove_peer_namespaces '1', ['ns1', 'ns2']
Took 0.0100 seconds
hbase(main):043:0> remove_peer_tableCFs '1', {'articles' => []}
Took 0.0082 seconds
hbase(main):044:0> set_peer_replicate_all '1', true
Took 0.0081 seconds
hbase(main):045:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED true
0
1 row(s)
Took 0.0086 seconds
=> #<Java::JavaUtil::ArrayList:0x4f82248f>
```

Установка false

```
hbase(main):032:0> set_peer_replicate_all '1', false

ERROR: java.io.IOException: Need clean exclude-namespaces or exclude-table-cfs config firstly when replicate_all flag is false
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:472)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)
Caused by: org.apache.hadoop.hbase.replication.ReplicationException: Need clean exclude-namespaces or exclude-table-cfs config
firstly when replicate_all flag is false
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.checkPeerConfig(ReplicationManager.java:144)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.updatePeerConfig(ReplicationManager.java:103)
        at org.apache.hadoop.hbase.master.HMaster.updateReplicationPeerConfig(HMaster.java:3535)
        at org.apache.hadoop.hbase.master.MasterRpcServices.updateReplicationPeerConfig(MasterRpcServices.java:1943)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        ... 3 more

  Set the replicate_all flag to true or false for the specified peer.

  If replicate_all flag is true, then all user tables (REPLICATION_SCOPE != 0)
  will be replicate to peer cluster. But you can use 'set_peer_exclude_namespaces'
  to set which namespaces can't be replicated to peer cluster. And you can use
  'set_peer_exclude_tableCFs' to set which tables can't be replicated to peer
  cluster.

  If replicate_all flag is false, then all user tables cannot be replicate to
  peer cluster. Then you can use 'set_peer_namespaces' or 'append_peer_namespaces'
  to set which namespaces will be replicated to peer cluster. And you can use
  'set_peer_tableCFs' or 'append_peer_tableCFs' to set which tables will be
  replicated to peer cluster.

  Notice: When you want to change a peer's replicate_all flag from false to true,
          you need clean the peer's NAMESPACES and TABLECFS config firstly.
          When you want to change a peer's replicate_all flag from true to false,
          you need clean the peer's EXCLUDE_NAMESPACES and EXCLUDE_TABLECFS
          config firstly.

  Examples:

    # set replicate_all flag to true
    hbase> set_peer_replicate_all '1', true
    # set replicate_all flag to false
    hbase> set_peer_replicate_all '1', false

Took 8.6671 seconds
hbase(main):033:0> set_peer_exclude_tableCFs '1', {}
Took 0.0110 seconds
hbase(main):034:0> set_peer_exclude_namespaces '1', []
Took 0.0080 seconds
hbase(main):035:0> set_peer_replicate_all '1', false
Took 0.0077 seconds
hbase(main):036:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED false
0
1 row(s)
Took 0.0052 seconds
=> #<Java::JavaUtil::ArrayList:0x3909a854>
```

set_peer_tableCFs

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Успешная установка таблицы
  - Неудачная установка таблицы

Определение

Устанавливает все реплицируемые таблицы/семейства столбцов для отношения репликации (peer). Следует помнить о следующих ограничениях:

- Флаг `REPLICATE_ALL` должен иметь значение `false` в конфигурации отношения репликации (set_peer_replicate_all).
- Если конфигурация peer уже содержит некоторое пространство имен (добавленное через append_peer_namespaces или set_peer_namespaces), то установка его таблиц с помощью этой команды не допускается.
- При необходимости очистить включенные таблицы/семейства столбцов нужно запустить текущую команду с пустым словарем в качестве второго входного параметра. Вы также можете использовать команду remove_peer_tableCFs.

> **ВНИМАНИЕ**
>
> Эта команда удаляет ранее определенные таблицы/семейства столбцов из конфигурации отношения репликации (в отличие от append_peer_tableCFs).

Использование

```
set_peer_tableCFs '<peer_id>',
                  {'[<namespace_name>:]<table_name>' => <column_families_array[]>[,...]}
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_families_array	Массив строк, указанных в формате `'<column_family>'`, где `<column_family>` — имя семейства столбцов

Примеры

Успешная установка таблицы

```
hbase(main):060:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1 default.articles;default.table10;default.temp1:cf1 0
1 row(s)
Took 0.0089 seconds
=> #<Java::JavaUtil::ArrayList:0x572b4072>
hbase(main):061:0> set_peer_tableCFs '1', {'articles' => []}
Took 0.0147 seconds
hbase(main):062:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  ENABLED false
ns1 default.articles 0
1 row(s)
Took 0.0079 seconds
=> #<Java::JavaUtil::ArrayList:0x53e166ad>
```

Неудачная установка таблицы

```
hbase(main):055:0> list_peers
 PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES TABLE_CFS BANDWIDTH
 1 serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase  DISABLED false
ns1 default.articles 0
1 row(s)
Took 0.0089 seconds
=> #<Java::JavaUtil::ArrayList:0x785d6a98>
hbase(main):056:0> set_peer_tableCFs '1', {'ns1:temp2' => []}

ERROR: java.io.IOException: Table-cfs ns1:temp2 is conflict with namespaces ns1 in peer config
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:472)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)
Caused by: org.apache.hadoop.hbase.replication.ReplicationException: Table-cfs ns1:temp2 is conflict with namespaces ns1 in peer
config
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.
checkNamespacesAndTableCfsConfigConflict(ReplicationManager.java:181)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.checkPeerConfig(ReplicationManager.java:148)
        at org.apache.hadoop.hbase.master.replication.ReplicationManager.updatePeerConfig(ReplicationManager.java:103)
        at org.apache.hadoop.hbase.master.HMaster.updateReplicationPeerConfig(HMaster.java:3535)
        at org.apache.hadoop.hbase.master.MasterRpcServices.updateReplicationPeerConfig(MasterRpcServices.java:1943)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        ... 3 more

  Set the replicable table-cf config for the specified peer.

  Note:
  1. The replicate_all flag need to be false when set the replicable table-cfs.
  2. Can't set a table to table-cfs config if it's namespace already was in
     namespaces config of this peer.

  Examples:

    # set table-cfs config is null, then the namespaces config decide which
    # table to be replicated.
    hbase> set_peer_tableCFs '1'
    # set table / table-cf to be replicable for a peer, for a table without
    # an explicit column-family list, all replicable column-families (with
    # replication_scope == 1) will be replicated
    hbase> set_peer_tableCFs '2',
     { "ns1:table1" => [],
     "ns2:table2" => ["cf1", "cf2"],
     "ns3:table3" => ["cfA", "cfB"]}


Took 8.1736 seconds
```

show_peer_tableCFs

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит реплицируемые таблицы/семейства столбцов для указанного отношения репликации (peer).

Использование

```
show_peer_tableCFs '<peer_id>'
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов

Примеры

```
hbase(main):014:0> show_peer_tableCFs '1'
default.articles
Took 0.0066 seconds
=> "default.articles"
```

update_peer_config

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Обновляет параметры конфигурации указанного отношения репликации (peer). Может быть полезно для настройки кастомных replication endpoints.

Использование

```
update_peer_config '<peer_id>'[,
                   DATA => {'<key_name>' => <key_value>, [...]}][,
                   CONFIG => {'<config_name>' => '<config_value>', [...]}]
```

Аргументы

Параметр	Описание
peer_id	Уникальный идентификатор отношения репликации (peer). Значение не должно содержать дефисов
key_name	Имя peer data-параметра
key_value	Значение peer data-параметра
config_name	Имя конфигурационного peer-параметра
config_value	Значение конфигурационного peer-параметра

Примеры

```
hbase(main):010:0> update_peer_config '1', CONFIG => {'config1' => 'value1'}
Took 0.0139 seconds
hbase(main):011:0> get_peer_config '1'
 Cluster Key                             serov-adh-1.ru-central1.internal,serov-adh-2.ru-central1.internal,serov-adh-3.ru-central1.internal:2181:/hbase
 config1                                 value1
Took 0.0047 seconds
=> #<Java::OrgApacheHadoopHbaseReplication::ReplicationPeerConfig:0x1ff463bb>
```

clone_snapshot

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Создает новую таблицу путем клонирования указанного снепшота. Последующая запись во вновь созданную таблицу не повлияет на данные в снепшоте.

Использование

```
clone_snapshot '<snapshot_name>',
               '[<namespace_name>:][table_name]'[,
               {RESTORE_ACL => <restore_acl_value>}]
```

Аргументы

Параметр	Описание
snapshot_name	Имя снепшота
namespace_name	Имя пространства имен
table_name	Имя таблицы
restore_acl_value	Логическое поле, определяющее, следует ли восстанавливать списки управления доступом (ACL)

Примеры

```
hbase(main):044:0> create 'temp10', 'cf1'
Created table temp10
Took 1.3420 seconds
=> Hbase::Table - temp10
hbase(main):045:0> clone_snapshot 'snap1', 'table10'
Took 1.2917 seconds
hbase(main):046:0> scan 'table10'
ROW                   COLUMN+CELL
 r1                   column=cf1:c1, timestamp=1637434534105, value=value3
 r1                   column=cf1:c2, timestamp=1637434285911, value=value4
 r2                   column=cf1:c1, timestamp=1637434293457, value=value5
2 row(s)
Took 0.0304 seconds
```

delete_all_snapshot

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Удаляет все снепшоты с именем, соответствующим заданному регулярному выражению.

Использование

```
delete_all_snapshot '<snapshot_regex>'
```

Аргументы

Параметр	Описание
snapshot_regex	Регулярное выражение для поиска снепшотов

Примеры

```
hbase(main):056:0> delete_all_snapshot 'snap.*'
SNAPSHOT              TABLE + CREATION TIME
No snapshots matched the regex snap.*
Took 0.0093 seconds
```

delete_snapshot

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Удаляет снепшот с указанным именем.

Использование

```
delete_snapshot '<snapshot_name>'
```

Аргументы

Параметр	Описание
snapshot_name	Имя снепшота

Примеры

```
hbase(main):048:0> list_snapshots
SNAPSHOT              TABLE + CREATION TIME
 snap1                temp1 (2021-11-24 06:31:27 UTC)
 snap2                ns1:temp2 (2021-11-24 06:32:21 UTC)
2 row(s)
Took 0.0153 seconds
=> ["snap1", "snap2"]
hbase(main):049:0> delete_snapshot 'snap2'
Took 0.0160 seconds
hbase(main):050:0> list_snapshots
SNAPSHOT              TABLE + CREATION TIME
 snap1                temp1 (2021-11-24 06:31:27 UTC)
1 row(s)
Took 0.0086 seconds
=> ["snap1"]
```

delete_table_snapshots

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Без использования регулярных выражений
  - С использованием регулярных выражений

Определение

Удаляет снепшоты указанной таблицы. Позволяет использовать регулярные выражения для определения имени таблицы и, опционально, имени снепшота.

Использование

```
delete_table_snapshots '<table_regex>'[, '<snapshot_regex>']
```

Аргументы

Параметр	Описание
table_regex	Регулярное выражение для поиска таблиц
snapshot_regex	Регулярное выражение для поиска снепшотов

Примеры

Без использования регулярных выражений

```
hbase(main):053:0> delete_table_snapshots 'temp1'
SNAPSHOT              TABLE + CREATION TIME
 snap1                temp1 (2021-11-24 06:31:27 UTC)

Delete the above 1 snapshots (y/n)?
y
Successfully deleted snapshot: snap1

Took 0.0207 seconds
=> [#<Java::OrgApacheHadoopHbaseClient::SnapshotDescription:0x3578a45b>]
```

С использованием регулярных выражений

```
hbase(main):054:0> delete_table_snapshots '.*', 'snap.*'
SNAPSHOT              TABLE + CREATION TIME
No snapshots matched the table name regular expression .* and the snapshot name regular expression snap.*
Took 0.0066 seconds
```

list_snapshots

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

- Без использования регулярных выражений
- С использованием регулярных выражений

Определение

Возвращает список снепшотов — всех или соответствующих заданному регулярному выражению.

Использование

```
list_snapshots ['<snapshot_regex>']
```

Аргументы

Параметр	Описание
snapshot_regex	Регулярное выражение для поиска снепшотов

Примеры

Без использования регулярных выражений

```
hbase(main):011:0> list_snapshots
SNAPSHOT              TABLE + CREATION TIME
 snap1                temp1 (2021-11-24 06:31:27 UTC)
 snap2                ns1:temp2 (2021-11-24 06:32:21 UTC)
2 row(s)
Took 0.0403 seconds
=> ["snap1", "snap2"]
```

С использованием регулярных выражений

```
hbase(main):013:0> list_snapshots 'snap1*'
SNAPSHOT              TABLE + CREATION TIME
 snap1                temp1 (2021-11-24 06:31:27 UTC)
1 row(s)
Took 0.0157 seconds
=> ["snap1"]
```

list_table_snapshots

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Без использования регулярных выражений
  - С использованием регулярных выражений

Определение

Возвращает снепшоты указанной таблицы. Позволяет использовать регулярные выражения для определения имени таблицы и, опционально, имени снепшота.

Использование

```
list_table_snapshots '<table_regex>'[, '<snapshot_regex>']
```

Аргументы

Параметр	Описание
table_regex	Регулярное выражение для поиска таблиц
snapshot_regex	Регулярное выражение для поиска снепшотов

Примеры

Без использования регулярных выражений

```
hbase(main):015:0> list_table_snapshots 'temp1'
SNAPSHOT              TABLE + CREATION TIME
 snap1                temp1 (2021-11-24 06:31:27 UTC)
1 row(s)
Took 0.0285 seconds
=> ["snap1"]
```

С использованием регулярных выражений

```
hbase(main):022:0> list_table_snapshots '.*', 'snap.*'
SNAPSHOT              TABLE + CREATION TIME
 snap1                temp1 (2021-11-24 06:31:27 UTC)
 snap2                ns1:temp2 (2021-11-24 06:32:21 UTC)
2 row(s)
Took 0.0180 seconds
=> ["snap1", "snap2"]
```

restore_snapshot

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Восстанавливает указанный снепшот. Может применяться только к таблицам типа disabled.

Использование

```
restore_snapshot '<snapshot_name>'[, {RESTORE_ACL => <restore_acl_value>}]
```

Аргументы

Параметр	Описание
snapshot_name	Имя снепшота
restore_acl_value	Логическое поле, определяющее, следует ли восстанавливать списки управления доступом (ACL)

Примеры

```
hbase(main):026:0> delete 'temp1', 'r2', 'cf1:c1'
Took 0.0769 seconds
hbase(main):028:0> delete 'temp1', 'r1', 'cf1:c2'
Took 0.0087 seconds
hbase(main):029:0> scan 'temp1'
ROW                   COLUMN+CELL
 r1                   column=cf1:c1, timestamp=1637434534105, value=value3
1 row(s)
Took 0.0236 seconds
hbase(main):038:0> disable 'temp1'
Took 0.8044 seconds
hbase(main):039:0> restore_snapshot 'snap1'
Took 1.3029 seconds
hbase(main):041:0> enable 'temp1'
Took 1.2668 seconds
hbase(main):042:0> scan 'temp1'
ROW                   COLUMN+CELL
 r1                   column=cf1:c1, timestamp=1637434534105, value=value3
 r1                   column=cf1:c2, timestamp=1637434285911, value=value4
 r2                   column=cf1:c1, timestamp=1637434293457, value=value5
2 row(s)
Took 0.0321 seconds
```

snapshot

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Создает снепшот указанной таблицы.

Использование

```
snapshot '[<namespace_name>:]<table_name>',
         '<snapshot_name>'[,
         {SKIP_FLUSH => <skip_flush_value>}]
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
snapshot_name	Имя снепшота
skip_flush_value	Логическое поле, определяющее, следует ли пропускать flush-операции.

Примеры

```
hbase(main):004:0> snapshot 'temp1', 'snap1'
Took 0.6258 seconds
hbase(main):009:0> snapshot 'ns1:temp2', 'snap2'
Took 0.5650 seconds
```

update_all_config

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Перезагружает подмножество параметров конфигурации на всех серверах кластера.

Использование

```
update_all_config
```

Примеры

```
hbase(main):018:0> update_all_config
Took 0.2048 seconds
```

update_config

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Перезагружает подмножество параметров конфигурации на указанных серверах кластера.

Использование

```
update_config '<server_name>,<port_number>,<start_code>'
```

Аргументы

Параметр	Описание
server_name	Имя сервера
port_number	Номер порта
start_code	Стартовый код сервера

TIP

Для получения `<server_name>`, `<port_number>` и `<start_code>` используйте команду status 'simple'.

Примеры

```
hbase(main):023:0> update_config 'bds-adh-2.ru-central1.internal,16000,1637647837862'
Took 0.0168 seconds
```

list_quotas

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

- Вывод списка всех квот
- Вывод списка квот, соответствующих регулярному выражению

Определение

Возвращает квоты, добавленные в HBase — все или соответствующие заданным регулярным выражениям.

Использование

```
list_quotas [USER => '<user_regex>'][,
             TABLE => '<table_regex>'][,
             NAMESPACE => '<namespace_regex>']
```

Аргументы

Параметр	Описание
namespace_regex	Регулярное выражение для поиска пространств имен
table_regex	Регулярное выражение для поиска таблиц
user_regex	Регулярное выражение для поиска пользователей

Примеры

Вывод списка всех квот

```
hbase(main):006:0> list_quotas
OWNER                   QUOTAS
 TABLE => temp1         TYPE => SPACE, TABLE => temp1, LIMIT => 1073741824, VIOLATION_POL
                        ICY => NO_INSERTS
1 row(s)
Took 0.1327 seconds
```

Вывод списка квот, соответствующих регулярному выражению

```
hbase(main):007:0> list_quotas TABLE => 't.*'
OWNER                   QUOTAS
 TABLE => temp1         TYPE => SPACE, TABLE => temp1, LIMIT => 1073741824, VIOLATION_POL
                        ICY => NO_INSERTS
1 row(s)
Took 0.0670 seconds
hbase(main):008:0> list_quotas NAMESPACE => 'ns.*'
OWNER                   QUOTAS
0 row(s)
Took 0.0241 seconds
```

list_snapshot_sizes

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Возвращает размер снепшотов, сделанных для таблиц с заданными квотами пространства. Эта информация используется в алгоритмах расчета размера квоты пространства.

РЕКОМЕНДАЦИЯ

С момента создания снепшота может потребоваться время, прежде чем информация о снепшоте появится в выводе команды. Размер файла можно получить, если таблица, из которой был создан снепшот, больше не ссылается на этот файл.

Использование

```
list_snapshot_sizes
```

Примеры

```
hbase(main):023:0> snapshot "temp1", "snap1"
Took 0.6061 seconds
hbase(main):026:0> list_snapshot_sizes
SNAPSHOT                 SIZE
 snap1                   0
1 row(s)
Took 0.0095 seconds
```

list_quota_snapshots

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Вывод без каких-либо параметров
  - Вывод для указанной таблицы
  - Вывод для указанного пространства имен
  - Запрос информации с указанного Region Server

Определение

Возвращает информацию о таблицах с определенными квотами дискового пространства: текущее использование пространства, настроенные ограничения, сведения о превышении квоты. По умолчанию команда считывает эти данные из снепшотов, хранящихся в системной таблице `hbase:quota`. Поскольку Region Servers хранят копии этих снепшотов (обновляются через регулярные промежутки времени), вы можете использовать входной параметр `REGIONSERVER` для получения информации с конкретного Region Server вместо `hbase:quota`.

Имя таблицы или пространства имен можно использовать для фильтрации вывода команды.

Использование

```
list_quota_snapshots({[TABLE => '<table_name>'][,
                       NAMESPACE => '<namespace_name>'][,
                       REGIONSERVER => '<region_server_name>']})
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
region_server_name	Имя Region Server

Выходные параметры

Параметр	Описание
TABLE	Имя таблицы
USAGE	Текущее использование пространства (в байтах)
LIMIT	Установленная квота пространства (в байтах)
IN_VIOLATION	Указывает на зафиксированное превышение квоты
POLICY	Политика, работающая в случае превышения квоты

Примеры

Вывод без каких-либо параметров

```
hbase(main):095:0> list_quota_snapshots
 TABLE USAGE LIMIT IN_VIOLATION POLICY
 temp1 9834 1073741824 false None
1 row(s)
Took 0.0103 seconds
```

Вывод для указанной таблицы

```
hbase(main):100:0> list_quota_snapshots({TABLE => 'temp1'})
 TABLE USAGE LIMIT IN_VIOLATION POLICY
 temp1 9834 1073741824 false None
1 row(s)
Took 0.0049 seconds
```

Вывод для указанного пространства имен

```
hbase(main):102:0> list_quota_snapshots({NAMESPACE => 'default'})
 TABLE USAGE LIMIT IN_VIOLATION POLICY
 temp1 9834 1073741824 false None
1 row(s)
Took 0.0052 seconds
```

Запрос информации с указанного Region Server

```
hbase(main):104:0> list_quota_snapshots({REGIONSERVER => 'bds-adh-1.ru-central1.internal,16020,1637744748523'})
 TABLE USAGE LIMIT IN_VIOLATION POLICY
 temp1 9834 1073741824 false None
1 row(s)
Took 0.0224 seconds
```

list_quota_table_sizes

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает вычисленный размер каждой таблицы в кластере. Master Server использует эту информацию для принятия решений о квотах пространства. В большинстве случаев вызов команды list_quota_snapshots дает более информативный результат.

Использование

```
list_quota_table_sizes
```

Примеры

```
hbase(main):001:0> list_quota_table_sizes
TABLE                                    SIZE
 SYSTEM.CATALOG                          34766
 t4                                      0
 SYSTEM.SEQUENCE                         0
 t5                                      0
 hbase:meta                              174888
 t6                                      0
 hbase:namespace                         5116
 temp10                                  0
 SYSTEM.STATS                            0
 hbase:acl                               9474
 temp1                                   9834
 hbase:quota                             15006
 SYSTEM.MUTEX                            4451
 SYSTEM.FUNCTION                         0
 hbase:rsgroup                           5318
 hbase:labels                            4897
 ns1:temp2                               9726
 SYSTEM.LOG                              0
 table10                                 0
19 row(s)
Took 0.3489 seconds
```

set_quota

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Применение квоты SPACE к таблице
  - Применение квоты THROTTLE к пространству имен
  - Удаление определенной квоты THROTTLE

Определение

Устанавливает новую квоту для определенного пользователя, таблицы или пространства имен. Предусматривает две формы квот:

- `SPACE` — может применяться как к таблицам, так и к пространствам имен. Каждая квота ограничивает размер таргета в файловой системе и определяет действия, которые должны выполняться, когда таргет превышает этот предел. К одной таблице можно применять разные квоты на уровне таблицы и пространства имен. Квоты на уровне таблицы имеют приоритет над квотами на уровне пространства имен.

- `THROTTLE` — могут применяться к пользователям, таблицам или пространствам имен. Каждая квота ограничивает количество запросов или объем данных, которые могут быть прочитаны, записаны или одновременно прочитаны и записаны за определенный период времени.

Использование

- Установка квоты `SPACE`:

```
set_quota TYPE => SPACE[,
          TABLE => '[<namespace_name>:]<table_name>'] | [, NAMESPACE => '<namespace_name>'],
          LIMIT => '<space_limit_value>',
          POLICY => <space_policy_value>
```

- Удаление квоты `SPACE`:

```
set_quota TYPE => SPACE[,
          TABLE => '[<namespace_name>:]<table_name>'] | [, NAMESPACE => '<namespace_name>'],
          LIMIT => NONE
```

- Установка квоты `THROTTLE`:

```
set_quota TYPE => THROTTLE[,
          THROTTLE_TYPE => READ | WRITE][,
          USER => '<user_name>'][,
          TABLE => '[<namespace_name>:]<table_name>'] | [, NAMESPACE => '<namespace_name>'],
          LIMIT => '<throttle_limit_value>'
```

- Удаление квоты `THROTTLE`:

```
set_quota TYPE => THROTTLE[,
          THROTTLE_TYPE => READ | WRITE][,
          USER => '<user_name>'][,
          TABLE => '[<namespace_name>:]<table_name>'] | [, NAMESPACE => '<namespace_name>'],
          LIMIT => NONE
```

ВНИМАНИЕ

Для квот `SPACE` требуется имя таблицы или имя пространства имен, но не оба значения. Для квот `THROTTLE` требуется имя пользователя, или имя таблицы, или имя пространства имен, или комбинация пользователь/пространство имен, или комбинация пользователь/таблица. Другие варианты запрещены.

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
space_limit_value	Ограничение дискового пространства, определенное для таблицы или пространства имен. Можно использовать стандартные метрические суффиксы: `B`, `K`, `M`, `G`, `T`, `P`. Формат по умолчанию — байты (без суффиксов). Примеры: `1G`, `2T`
space_policy_value	Политика, которую нужно применить, если зафиксировано нарушение квоты занимаемого пространства. Возможные значения: - `NO_INSERTS` — запрещает вставку данных (например, put, incr, append); - `NO_WRITES` — работает так же, как `NO_INSERTS`, но запрещает delete-операции; - `NO_WRITES_COMPACTIONS` — работает так же, как `NO_WRITES`, но запрещает сжатие (compactions); - `DISABLE` — делает таблицу недоступной (disabled).
user_name	Имя пользователя
throttle_limit_value	Лимит, ограничивающий количество запросов или объем данных для операций чтения/записи. Предоставляет две формы синтаксиса: - `<requests_number>req/sec|min|hour|day` — количество запросов за определенный период времени. Примеры: `100req/min`, `10req/sec`; - `<data_amount>B|K|M|G|T|P/sec|min|hour|day` — количество данных за определенный период времени, где `B`, `K`, `M`, `G`, `T`, `P` — стандартные суффиксы метрик. Примеры: `10M/sec`, `5K/min`.

Примеры

Применение квоты SPACE к таблице

```
hbase(main):002:0> set_quota TYPE => SPACE, TABLE => 'temp1', LIMIT => '1G', POLICY => NO_INSERTS
Took 0.4164 seconds
```

Применение квоты THROTTLE к пространству имен

```
hbase(main):003:0> set_quota TYPE => THROTTLE, NAMESPACE => 'ns1', LIMIT => '10req/sec'
Took 0.0240 seconds
```

Удаление определенной квоты THROTTLE

```
hbase(main):004:0> set_quota TYPE => THROTTLE, NAMESPACE => 'ns1', LIMIT => NONE
Took 0.0129 seconds
```

grant

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

- Предоставление прав без использования пространств имен и имен таблиц
- Предоставление прав по указанному пространству имен
- Предоставление прав по указанному столбцу таблицы

Определение

Предоставляет пользователю или группе пользователей определенные права.

ВАЖНО

Обратите внимание, что эту команду могут выполнять только пользователи с правами `ADMIN`. Таким образом, команду необходимо запускать либо суперпользователем `hbase`, либо пользователем с соответствующими правами:

```
sudo -u hbase hbase shell
grant '<user_name>', 'A'
```

Использование

```
grant '<user_name>' | '@<group_name>',
      '<permissions>' [,
      '@<namespace_name>'] | [, '[<namespace_name>:]<table_name>' [, '<column_family>' [, '<column_qualifier>']]]
```

Аргументы

Параметр	Описание
user_name	Имя пользователя
group_name	Имя группы пользователей
permissions	Ноль или более букв из набора `RWXCA`, где `R` означает ЧТЕНИЕ, `W` — ЗАПИСЬ, `X` — ВЫПОЛНЕНИЕ, `C` — СОЗДАНИЕ, `A` — АДМИНИСТРИРОВАНИЕ. Примеры: `RW`, `RWX`, `RWXCA`, и так далее
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)

ПРИМЕЧАНИЕ

Пространству имен и имени группы должен предшествовать символ `@`.

Примеры

Предоставление прав без использования пространств имен и имен таблиц

```
hbase(main):003:0> grant 'dasha', 'RWXCA'
Took 0.2991 seconds
```

Предоставление прав по указанному пространству имен

```
hbase(main):004:0> grant 'dasha', 'RWXCA', '@ns1'
Took 0.0422 seconds
hbase(main):010:0> user_permission '@ns1'
User                     Namespace,Table,Family,Qualifier:Permission
 dasha                   ns1,,,: [Permission: actions=READ,WRITE,EXEC,CREATE,ADMIN]
1 row(s)
Took 0.0419 seconds
```

Предоставление прав по указанному столбцу таблицы

```
hbase(main):009:0> grant 'dasha', 'RW', 'ns1:temp2', 'cf1', 'c1'
Took 0.0795 seconds
hbase(main):011:0> user_permission 'ns1:temp2'
User                     Namespace,Table,Family,Qualifier:Permission
 dasha                   ns1,ns1:temp2,,: [Permission: actions=READ,WRITE,EXEC,CREATE,ADMIN]
 dasha                   ns1,ns1:temp2,cf1,c1: [Permission: actions=READ,WRITE]
2 row(s)
Took 0.0315 seconds
```

list_security_capabilities

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает поддерживаемые возможности безопасности.

Использование

```
list_security_capabilities
```

Примеры

```
hbase(main):001:0> list_security_capabilities
AUTHORIZATION
SIMPLE_AUTHENTICATION
CELL_VISIBILITY
CELL_AUTHORIZATION
Took 0.3205 seconds
=> ["AUTHORIZATION", "SIMPLE_AUTHENTICATION", "CELL_VISIBILITY", "CELL_AUTHORIZATION"]
```

revoke

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Отзыв прав на указанный столбец таблицы
  - Отзыв прав на указанную таблицу
  - Отзыв прав на указанное пространство имен
  - Отзыв прав без использования пространств имен и таблиц

Определение

Отзывает права доступа пользователя.

ВАЖНО

Обратите внимание, что эту команду могут выполнять только пользователи с правами `ADMIN`. Таким образом, команду необходимо запускать либо суперпользователем `hbase`, либо пользователем с соответствующими правами:

```
sudo -u hbase hbase shell
grant '<user_name>', 'A'
```

Использование

```
revoke '<user_name>' | '@<group_name>' [,
      '@<namespace_name>'] | [, '[<namespace_name>:]<table_name>' [, '<column_family>' [, '<column_qualifier>']]]
```

Аргументы

Параметр	Описание
user_name	Имя пользователя
group_name	Имя группы пользователей
namespace_name	Имя пространства имен
table_name	Имя таблицы
column_family	Имя семейства столбцов
column_qualifier	Квалификатор столбца (column qualifier)

ПРИМЕЧАНИЕ

Пространству имен и имени группы должен предшествовать символ `@`.

Примеры

Отзыв прав на указанный столбец таблицы

```
hbase(main):011:0> user_permission 'ns1:temp2'
User                     Namespace,Table,Family,Qualifier:Permission
 dasha                   ns1,ns1:temp2,,: [Permission: actions=READ,WRITE,EXEC,CREATE,ADMIN]
 dasha                   ns1,ns1:temp2,cf1,c1: [Permission: actions=READ,WRITE]
2 row(s)
Took 0.0315 seconds
hbase(main):001:0> revoke 'dasha', 'ns1:temp2', 'cf1', 'c1'
Took 0.5811 seconds
hbase(main):003:0> user_permission 'ns1:temp2'
User                  Namespace,Table,Family,Qualifier:Permission
 dasha                ns1,ns1:temp2,,: [Permission: actions=READ,WRITE,EXEC,CREA
                      TE,ADMIN]
1 row(s)
Took 0.0458 seconds
```

Отзыв прав на указанную таблицу

```
hbase(main):003:0> user_permission 'ns1:temp2'
User                  Namespace,Table,Family,Qualifier:Permission
 dasha                ns1,ns1:temp2,,: [Permission: actions=READ,WRITE,EXEC,CREA
                      TE,ADMIN]
1 row(s)
Took 0.0458 seconds
hbase(main):008:0> revoke 'dasha', 'ns1:temp2'
Took 0.0295 seconds
hbase(main):009:0> user_permission 'ns1:temp2'
User                  Namespace,Table,Family,Qualifier:Permission
0 row(s)
Took 0.0374 seconds
```

Отзыв прав на указанное пространство имен

```
hbase(main):010:0> user_permission '@ns1'
User                     Namespace,Table,Family,Qualifier:Permission
 dasha                   ns1,,,: [Permission: actions=READ,WRITE,EXEC,CREATE,ADMIN]
1 row(s)
Took 0.0419 seconds
hbase(main):004:0> revoke 'dasha', '@ns1'
Took 0.0268 seconds
hbase(main):006:0> user_permission '@ns1'
User                  Namespace,Table,Family,Qualifier:Permission
0 row(s)
Took 0.0355 seconds
```

Отзыв прав без использования пространств имен и таблиц

```
hbase(main):011:0> revoke 'dasha'
Took 0.0125 seconds
hbase(main):012:0> user_permission
User                  Namespace,Table,Family,Qualifier:Permission

ERROR: org.apache.hadoop.hbase.security.AccessDeniedException: Insufficient permissions for user 'dasha' (global, action=ADMIN)
        at org.apache.hadoop.hbase.security.access.AccessChecker.requireGlobalPermission(AccessChecker.java:158)
        at org.apache.hadoop.hbase.security.access.AccessChecker.requirePermission(AccessChecker.java:129)
        at org.apache.hadoop.hbase.security.access.AccessController.getUserPermissions(AccessController.java:2182)
        at org.apache.hadoop.hbase.protobuf.generated.AccessControlProtos$AccessControlService$1.
getUserPermissions(AccessControlProtos.java:10039)
        at org.apache.hadoop.hbase.protobuf.generated.AccessControlProtos$AccessControlService.callMethod(AccessControlProtos.
java:10197)
        at org.apache.hadoop.hbase.regionserver.HRegion.execService(HRegion.java:8049)
        at org.apache.hadoop.hbase.regionserver.RSRpcServices.execServiceOnRegion(RSRpcServices.java:2409)
        at org.apache.hadoop.hbase.regionserver.RSRpcServices.execService(RSRpcServices.java:2391)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.ClientProtos$ClientService$2.callBlockingMethod(ClientProtos.
java:42010)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)

Show all permissions for the particular user.
Syntax : user_permission <table>

Note: A namespace must always precede with '@' character.

For example:

    hbase> user_permission
    hbase> user_permission '@ns1'
    hbase> user_permission '@.*'
    hbase> user_permission '@^[a-c].*'
    hbase> user_permission 'table1'
    hbase> user_permission 'namespace1:table1'
    hbase> user_permission '.*'
    hbase> user_permission '^[A-C].*'

Took 0.5466 seconds
```

user_permission

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Выводит все разрешения для конкретного пользователя.

ВАЖНО

Обратите внимание, что эту команду могут выполнять только пользователи с правами `ADMIN`. Таким образом, команду необходимо запускать либо суперпользователем `hbase`, либо пользователем с соответствующими правами:

```
sudo -u hbase hbase shell
grant '<user_name>', 'A'
```

Использование

```
user_permission ['@<namespace_name>' | '@<namespace_regex>' | '<table_name>' | '<table_regex>']
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
namespace_regex	Регулярное выражение для поиска пространств имен
table_regex	Регулярное выражение для поиска таблиц

ПРИМЕЧАНИЕ

Перед пространством имен должен стоять символ `@`.

Примеры

```
hbase(main):003:0> user_permission
User                     Namespace,Table,Family,Qualifier:Permission
 dasha                   hbase,hbase:acl,,: [Permission: actions=READ,WRITE,EXEC,CREATE,ADMIN
                         ]
 hbase                   hbase,hbase:acl,,: [Permission: actions=READ,WRITE,EXEC,CREATE,ADMIN
                         ]
2 row(s)
Took 0.6043 seconds
```

abort_procedure

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Прерывает процедуру с указанным идентификатором. Возвращает `true`, если команда принята и прерывание процедуры находится в процессе. Возвращает `false`, если процедура не может быть прервана, то есть если определенная процедура не существует, или уже завершена, или вызов команды может привести к сбою.

ВНИМАНИЕ

Используйте команду с осторожностью. Она предназначена только для специалистов. Некоторые процедуры нельзя прервать.

Использование

```
abort_procedure <proc_id>[, <may_interrupt_if_running>]
```

Аргументы

Параметр	Описание
proc_id	Идентификатор процедуры (можно получить, вызвав list_procedures)
may_interrupt_if_running	Опциональное логическое значение, определяющее, разрешено ли прерывать запущенную процедуру. Значение по умолчанию — `true`

Примеры

```
hbase(main):036:0> abort_procedure 123
false
Took 0.0157 seconds
=> 1
```

list_locks

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает все блокировки (locks) в HBase.

Использование

```
list_locks
```

Примеры

```
hbase(main):035:0> list_locks
Took 0.0158 seconds
=> []
```

list_procedures

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает все процедуры в HBase.

Использование

```
list_procedures
```

Примеры

```
hbase(main):001:0> list_procedures
 Id Name State Submitted_Time Last_Update Parameters
 84 org.apache.hadoop.hbase.procedure2.ProcedureExecutor$FailedProcedure ROLLEDBACK 2021-11-29 11:35:46 UTC 2021-11-29 11:35:46
UTC
1 row(s)
Took 0.5023 seconds
```

add_labels

Константин Алпашкин

Содержание

Назад к содержанию

[Определение](#)

[Использование](#)

[Примеры](#)

- [Добавление одной метки](#)
- [Добавление набора меток](#)

Определение

Добавляет набор меток видимости (visibility labels).

Использование

```
add_labels <labels_array[]>
```

Аргументы

Параметр	Описание
labels_array	Массив строк с именами для новых меток видимости

Примеры

Добавление одной метки

```
hbase(main):002:0> add_labels ['B1']
Took 0.1502 seconds
```

Добавление набора меток

```
hbase(main):005:0> add_labels ['SECRET','PRIVATE']
Took 0.0127 seconds
```

clear_auths

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Очищает метки видимости (visibility labels), назначенные определенному пользователю или группе.

Использование

```
clear_auths '@<group_name>' | '<user_name>', <labels_array[]>
```

Аргументы

Параметр	Описание
group_name	Имя группы пользователей
user_name	Имя пользователя
labels_array	Массив меток видимости

ВНИМАНИЕ

Вызов команды без параметра `labels_array` возможен, но не приводит к каким-либо изменениям.

Примеры

```
hbase(main):005:0> clear_auths 'dasha', ['SECRET']
Took 0.1608 seconds
hbase(main):006:0> get_auths 'dasha'
PRIVATE
Took 0.0129 seconds
=> #<Java::JavaUtil::Collections::UnmodifiableRandomAccessList:0x526893f>
```

get_auths

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Возвращает метки видимости (visibility labels), назначенные определенному пользователю или группе.

Использование

```
get_auths '@<group_name>' | '<user_name>'
```

Аргументы

Параметр	Описание
group_name	Имя группы пользователей
user_name	Имя пользователя

ПРИМЕЧАНИЕ

Перед именем группы должен стоять символ `@`.

Примеры

```
hbase(main):006:0> get_auths 'dasha'
SECRET
PRIVATE
Took 0.0176 seconds
=> #<Java::JavaUtil::Collections::UnmodifiableRandomAccessList:0x56f2c9e8>
```

list_labels

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Получение всех меток видимости
  - Получение меток, соответствующих регулярному выражению

Определение

Возвращает список меток видимости (visibility labels), определенных в HBase. Опциональный параметр регулярного выражения можно использовать для фильтрации вывода.

Использование

```
list_labels ['<label_regex>']
```

Аргументы

Параметр	Описание
label_regex	Регулярное выражение для поиска меток видимости

Примеры

Получение всех меток видимости

```
hbase(main):001:0> list_labels
A1
SECRET
B1
PRIVATE
Took 0.6317 seconds
=> #<Java::JavaUtil::Collections::UnmodifiableRandomAccessList:0x534d0e20>
```

Получение меток, соответствующих регулярному выражению

```
hbase(main):002:0> list_labels '.1'
A1
B1
Took 0.0087 seconds
=> #<Java::JavaUtil::Collections::UnmodifiableRandomAccessList:0x30c8c6ab>
```

set_auths

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Назначает набор меток видимости (visibility labels) определенному пользователю или группе.

Использование

```
set_auths '@<group_name>' | '<user_name>', <labels_array[]>
```

Аргументы

Параметр	Описание
group_name	Имя группы пользователей
user_name	Имя пользователя
labels_array	Массив меток видимости

ПРИМЕЧАНИЕ

Перед именем группы должен стоять символ `@`.

Примеры

```
hbase(main):004:0> set_auths 'dasha', ['SECRET','PRIVATE']
Took 0.2063 seconds
```

set_visibility

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Установка видимости для всех существующих ячеек определенного столбца
  - Попытка использования всех доступных параметров
  - Попытка использования несуществующих меток видимости

Определение

Задает выражение видимости для одной или нескольких существующих ячеек таблицы. Требует имя таблицы, выражение с использованием доступных меток видимости и словарь, содержащий спецификации для сканирования нужных ячеек.

ПРИМЕЧАНИЕ

Эта команда изменяет только существующие ячейки и предназначена в основном для тестирования функций и функциональной проверки. Возможно определить выражение видимости для новых ячеек, используя команды put или append.

Использование

```
set_visibility '[<namespace_name>:]<table_name>',
               '<visibility_expression>',
               {COLUMNS => <columns_array[]>[,
               TIMERANGE => [<timestamp_start>, <timestamp_end>]][,
               TIMESTAMP => <cell_timestamp>][,
               ROWPREFIXFILTER => '<row_prefix>'][,
               STARTROW => '<start_row_key>'][,
               STOPROW => '<stop_row_key>'][,
               FILTER => "<filter_description>"]}
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы
visibility_expression	Логическое выражение, которое относится к существующей метке видимости. Могут использоваться следующие логические операторы: `|` (или), `&` (и). Примеры: `SECRET|PRIVATE`, `(SECRET&PRIVATE)|PUBLIC`, `SECRET&PRIVATE&PUBLIC`
columns_array	Массив строк, указанных в формате `'<column_family>:<column_qualifier>'`, где `<column_family>` — имя семейства столбцов; `<column_qualifier>` — квалификатор столбца
timestamp_start	Начальное значение для поиска временных меток ячеек
timestamp_end	Конечное значение для поиска временных меток ячеек
cell_timestamp	Временная метка ячейки
row_prefix	Префикс для поиска ключей строки
start_row_key	Начальное значение для поиска ключей строки
stop_row_key	Конечное значение для поиска ключей строки
filter_description	Фильтр, используемый для поиска ячеек таблицы

Примеры

Установка видимости для всех существующих ячеек определенного столбца

```
hbase(main):010:0> set_visibility 'ns1:temp2', 'PRIVATE|SECRET', {COLUMNS => ['cf1:c1']}
1 row(s)
Took 0.0841 seconds
```

Попытка использования всех доступных параметров

```
hbase(main):009:0> set_visibility 'ns1:temp2', 'PRIVATE|SECRET', {COLUMNS => ['cf1:c1'],  TIMERANGE => [1303668804000,
1303668904000], TIMESTAMP => 1303668804000, ROWPREFIXFILTER => 'r', STARTROW => 'r1', STOPROW => 'r2', FILTER => "ValueFilter(=,
'regexstring:value*')"}
0 row(s)
Took 0.0427 seconds
```

Попытка использования несуществующих меток видимости

```
hbase(main):013:0> set_visibility 'ns1:temp2', 'Z&Y', {COLUMNS => ['cf1:c1']}

ERROR: org.apache.hadoop.hbase.exceptions.FailedSanityCheckException: Invalid visibility label Z
        at org.apache.hadoop.hbase.regionserver.HRegion.doBatchMutate(HRegion.java:4119)
        at org.apache.hadoop.hbase.regionserver.HRegion.put(HRegion.java:2975)
        at org.apache.hadoop.hbase.regionserver.RSRpcServices.mutate(RSRpcServices.java:2806)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.ClientProtos$ClientService$2.callBlockingMethod(ClientProtos.
java:42000)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)

Set the visibility expression on one or more existing cells.

Pass table name, visibility expression, and a dictionary containing
scanner specifications.  Scanner specifications may include one or more
of: TIMERANGE, FILTER, STARTROW, STOPROW, ROWPREFIXFILTER, TIMESTAMP, or COLUMNS

If no columns are specified, all columns will be included.
To include all members of a column family, leave the qualifier empty as in
'col_family:'.

The filter can be specified in two ways:
1. Using a filterString - more information on this is available in the
Filter Language document attached to the HBASE-4176 JIRA
2. Using the entire package name of the filter.

Examples:

    hbase> set_visibility 't1', 'A|B', {COLUMNS => ['c1', 'c2']}
    hbase> set_visibility 't1', '(A&B)|C', {COLUMNS => 'c1',
        TIMERANGE => [1303668804000, 1303668904000]}
    hbase> set_visibility 't1', 'A&B&C', {ROWPREFIXFILTER => 'row2',
        FILTER => "(QualifierFilter (>=, 'binary:xyz')) AND
        (TimestampsFilter ( 123, 456))"}

This command will only affect existing cells and is expected to be mainly
useful for feature testing and functional verification.

Took 0.5326 seconds
```

add_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Создает новую группу Region Server.

Использование

```
add_rsgroup '<rsgroup_name>'
```

Аргументы

Параметр	Описание
rsgroup_name	Имя новой группы Region Server

Примеры

```
hbase(main):001:0> add_rsgroup 'test_group'
Took 0.4366 seconds
```

balance_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выполняет балансировку группы Region Server.

Использование

```
balance_rsgroup '<rsgroup_name>'
```

Аргументы

Параметр	Описание
rsgroup_name	Имя группы Region Server

Примеры

```
hbase(main):030:0> balance_rsgroup 'default'
Ran the balancer.
Took 0.7246 seconds
=> true
```

get_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Получает информацию о группе Region Server.

Использование

```
get_rsgroup '<rsgroup_name>'
```

Аргументы

Параметр	Описание
rsgroup_name	Имя группы Region Server

Примеры

```
hbase(main):031:0> get_rsgroup 'default'
SERVERS
bds-adh-1.ru-central1.internal:16020
bds-adh-2.ru-central1.internal:16020
2 row(s)
TABLES
hbase:meta
hbase:acl
SYSTEM.CATALOG
ns1:temp2
t5
t6
hbase:labels
hbase:namespace
SYSTEM.SEQUENCE
SYSTEM.LOG
SYSTEM.FUNCTION
SYSTEM.MUTEX
SYSTEM.STATS
hbase:rsgroup
14 row(s)
Took 0.0044 seconds
hbase(main):032:0> get_rsgroup 'test_group'
SERVERS
bds-adh-3.ru-central1.internal:16020
1 row(s)
TABLES
t4
temp1
2 row(s)
Took 0.0035 seconds
```

get_server_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает имя группы, которой принадлежит данный Region Server.

Использование

```
get_server_rsgroup '<region_server_name>:<port_number>'
```

Аргументы

Параметр	Описание
region_server_name	Имя Region Server
port_number	Номер порта

Примеры

```
hbase(main):033:0> get_server_rsgroup 'bds-adh-3.ru-central1.internal:16020'
test_group
1 row(s)
Took 0.0170 seconds
hbase(main):034:0> get_server_rsgroup 'bds-adh-1.ru-central1.internal:16020'
default
1 row(s)
Took 0.0039 seconds
hbase(main):035:0> get_server_rsgroup 'bds-adh-2.ru-central1.internal:16020'
default
1 row(s)
Took 0.0042 seconds
```

get_table_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Возвращает имя группы, которой принадлежит данная таблица.

Использование

```
get_table_rsgroup '[<namespace_name>:]<table_name>'
```

Аргументы

Параметр	Описание
namespace_name	Имя пространства имен
table_name	Имя таблицы

Примеры

```
hbase(main):036:0> get_table_rsgroup 'temp1'
test_group
1 row(s)
Took 0.0124 seconds
hbase(main):037:0> get_table_rsgroup 'ns1:temp2'
default
1 row(s)
Took 0.0041 seconds
```

list_rsgroups

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры
- Получение всех групп Region Server
- Получение групп, соответствующих регулярному выражению

Определение

Возвращает все группы Region Server. Опциональный параметр регулярного выражения можно использовать для фильтрации вывода.

Использование

```
list_rsgroups ['<rsgroup_regex>']
```

Аргументы

Параметр	Описание
rsgroup_regex	Регулярное выражение для поиска групп Region Server

Примеры

Получение всех групп Region Server

```
hbase(main):004:0> list_rsgroups
NAME                     SERVER / TABLE
 test_group
 default                 server bds-adh-1.ru-central1.internal:16020
                         server bds-adh-2.ru-central1.internal:16020
                         server bds-adh-3.ru-central1.internal:16020
                         table hbase:meta
                         table hbase:acl
                         table SYSTEM.CATALOG
                         table t4
                         table t5
                         table t6
                         table hbase:labels
                         table temp1
                         table hbase:namespace
                         table SYSTEM.SEQUENCE
                         table SYSTEM.LOG
                         table SYSTEM.FUNCTION
                         table SYSTEM.MUTEX
                         table SYSTEM.STATS
                         table hbase:rsgroup
2 row(s)
Took 0.0289 seconds
```

Получение групп, соответствующих регулярному выражению

```
hbase(main):005:0> list_rsgroups 'test*'
NAME                     SERVER / TABLE
 test_group
1 row(s)
Took 0.0069 seconds
```

move_namespaces_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Назначает указанные пространства имен в определенную группу Region Server.

Использование

```
move_namespaces_rsgroup '<rsgroup_name>', <namespaces_array[]>
```

Аргументы

Параметр	Описание
rsgroup_name	Имя группы Region Server
namespaces_array	Массив строк, указанных в формате `'<namespace_name>'`, где `<namespace_name>` — имя пространства имен

Примеры

```
hbase(main):022:0> create 'ns1:temp2', 'cf1'
Created table ns1:temp2
Took 1.0197 seconds
=> Hbase::Table - ns1:temp2
hbase(main):023:0> list_namespace_tables 'ns1'
TABLE
temp2
1 row(s)
Took 0.0096 seconds
=> ["temp2"]
hbase(main):024:0> list_rsgroups
NAME                     SERVER / TABLE
 test_group              server bds-adh-2.ru-central1.internal:16020
                         server bds-adh-3.ru-central1.internal:16020
                         table t4
                         table temp1
 default                 server bds-adh-1.ru-central1.internal:16020
                         table hbase:meta
                         table hbase:acl
                         table SYSTEM.CATALOG
                         table ns1:temp2
                         table t5
                         table t6
                         table hbase:labels
                         table hbase:namespace
                         table SYSTEM.SEQUENCE
                         table SYSTEM.LOG
                         table SYSTEM.FUNCTION
                         table SYSTEM.MUTEX
                         table SYSTEM.STATS
                         table hbase:rsgroup
2 row(s)
Took 0.0117 seconds
hbase(main):025:0> move_namespaces_rsgroup 'test_group', ['ns1']
Took 1.1491 seconds
hbase(main):026:0> list_rsgroups
NAME                     SERVER / TABLE
 test_group              server bds-adh-2.ru-central1.internal:16020
                         server bds-adh-3.ru-central1.internal:16020
                         table ns1:temp2
                         table t4
                         table temp1
 default                 server bds-adh-1.ru-central1.internal:16020
                         table hbase:meta
                         table hbase:acl
                         table SYSTEM.CATALOG
                         table t5
                         table t6
                         table hbase:labels
                         table hbase:namespace
                         table SYSTEM.SEQUENCE
                         table SYSTEM.LOG
                         table SYSTEM.FUNCTION
                         table SYSTEM.MUTEX
                         table SYSTEM.STATS
                         table hbase:rsgroup
2 row(s)
Took 0.0055 seconds
```

move_servers_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Назначает указанные серверы в определенную группу Region Server.

Использование

```
move_servers_rsgroup '<rsgroup_name>', <region_servers_array[]>
```

Аргументы

Параметр	Описание
rsgroup_name	Имя группы Region Server
region_servers_array	Массив строк, указанных в формате `'<region_server_name>:<port_number>'`, где `<region_server_name>` — имя Region Server; `<port_number>` — номер порта

Примеры

```
hbase(main):014:0> move_servers_rsgroup 'test_group', ['bds-adh-3.ru-central1.internal:16020']
Took 19.6359 seconds
hbase(main):015:0> list_rsgroups
NAME                     SERVER / TABLE
 test_group              server bds-adh-3.ru-central1.internal:16020
 default                 server bds-adh-1.ru-central1.internal:16020
                         server bds-adh-2.ru-central1.internal:16020
                         table hbase:meta
                         table hbase:acl
                         table SYSTEM.CATALOG
                         table t4
                         table t5
                         table t6
                         table hbase:labels
                         table temp1
                         table hbase:namespace
                         table SYSTEM.SEQUENCE
                         table SYSTEM.LOG
                         table SYSTEM.FUNCTION
                         table SYSTEM.MUTEX
                         table SYSTEM.STATS
                         table hbase:rsgroup
2 row(s)
Took 0.0125 seconds
```

move_servers_namespaces_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Назначает указанные серверы и пространства имен в определенную группу Region Server.

Использование

```
move_servers_namespaces_rsgroup '<rsgroup_name>',
                                <region_servers_array[]>,
                                <namespaces_array[]>
```

Аргументы

Параметр	Описание
rsgroup_name	Имя группы Region Server
region_servers_array	Массив строк, указанных в формате `'<region_server_name>:<port_number>'`, где `<region_server_name>` — имя Region Server; `<port_number>` — номер порта
namespaces_array	Массив строк, указанных в формате `'<namespace_name>'`, где `<namespace_name>` — имя пространства имен

Примеры

```
hbase(main):027:0> move_servers_namespaces_rsgroup 'default', ['bds-adh-2.ru-central1.internal:16020'], ['ns1']
Took 4.2444 seconds
hbase(main):028:0> list_rsgroups
NAME                     SERVER / TABLE
 test_group              server bds-adh-3.ru-central1.internal:16020
                         table t4
                         table temp1
 default                 server bds-adh-1.ru-central1.internal:16020
                         server bds-adh-2.ru-central1.internal:16020
                         table hbase:meta
                         table hbase:acl
                         table SYSTEM.CATALOG
                         table ns1:temp2
                         table t5
                         table t6
                         table hbase:labels
                         table hbase:namespace
                         table SYSTEM.SEQUENCE
                         table SYSTEM.LOG
                         table SYSTEM.FUNCTION
                         table SYSTEM.MUTEX
                         table SYSTEM.STATS
                         table hbase:rsgroup
2 row(s)
Took 0.0063 seconds
```

move_servers_tables_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Назначает указанные серверы и таблицы в определенную группу Region Server.

Использование

```
move_servers_rsgroup '<rsgroup_name>',
                     <region_servers_array[]>,
                     <tables_array[]>
```

Аргументы

Параметр	Описание
rsgroup_name	Имя группы Region Server
region_servers_array	Массив строк, указанных в формате `'<region_server_name>:<port_number>'`, где `<region_server_name>` — имя Region Server; `<port_number>` — номер порта
tables_array	Массив строк, указанных в формате `'[<namespace_name>:]<table_name>'`, где `<namespace_name>` — имя пространства имен; `<table_name>` — имя таблицы

Примеры

```
hbase(main):019:0> move_servers_tables_rsgroup 'test_group',['bds-adh-2.ru-central1.internal:16020'], ['t4']
Took 27.6992 seconds
hbase(main):020:0> list_rsgroups
NAME                     SERVER / TABLE
 test_group              server bds-adh-2.ru-central1.internal:16020
                         server bds-adh-3.ru-central1.internal:16020
                         table t4
                         table temp1
 default                 server bds-adh-1.ru-central1.internal:16020
                         table hbase:meta
                         table hbase:acl
                         table SYSTEM.CATALOG
                         table t5
                         table t6
                         table hbase:labels
                         table hbase:namespace
                         table SYSTEM.SEQUENCE
                         table SYSTEM.LOG
                         table SYSTEM.FUNCTION
                         table SYSTEM.MUTEX
                         table SYSTEM.STATS
                         table hbase:rsgroup
2 row(s)
Took 0.0063 seconds
```

move_tables_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Назначает указанные таблицы в определенную группу Region Server.

Использование

```
move_tables_rsgroup '<rsgroup_name>', <tables_array[]>
```

Аргументы

Параметр	Описание
rsgroup_name	Имя группы Region Server
tables_array	Массив строк, указанных в формате `'[<namespace_name>:]<table_name>'`, где `<namespace_name>` — имя пространства имен; `<table_name>` — имя таблицы

Примеры

```
hbase(main):016:0> move_tables_rsgroup 'test_group', ['temp1']
Took 1.1676 seconds
hbase(main):017:0> list_rsgroups
NAME                         SERVER / TABLE
 test_group                  server bds-adh-3.ru-central1.internal:16020
                             table temp1
 default                     server bds-adh-1.ru-central1.internal:16020
                             server bds-adh-2.ru-central1.internal:16020
                             table hbase:meta
                             table hbase:acl
                             table SYSTEM.CATALOG
                             table t4
                             table t5
                             table t6
                             table hbase:labels
                             table hbase:namespace
                             table SYSTEM.SEQUENCE
                             table SYSTEM.LOG
                             table SYSTEM.FUNCTION
                             table SYSTEM.MUTEX
                             table SYSTEM.STATS
                             table hbase:rsgroup
2 row(s)
Took 0.0097 seconds
```

remove_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры
  - Попытка удаления группы с таблицами
  - Попытка удаления группы с серверами
  - Удаление группы без серверов и таблиц

Определение

Удаляет указанную группу Region Server.

ВНИМАНИЕ

Перед выполнением этой операции все серверы и таблицы должны быть удалены из группы Region Server.

Использование

```
remove_rsgroup '<rsgroup_name>'
```

Аргументы

Параметр	Описание
rsgroup_name	Имя группы Region Server

Примеры

Попытка удаления группы с таблицами

```
hbase(main):018:0> remove_rsgroup 'test_group'

ERROR: org.apache.hadoop.hbase.constraint.ConstraintException: RSGroup test_group has 2 tables; you must remove these tables from
the rsgroup before the rsgroup can be removed.
        at org.apache.hadoop.hbase.rsgroup.RSGroupAdminServer.removeRSGroup(RSGroupAdminServer.java:460)
        at org.apache.hadoop.hbase.rsgroup.RSGroupAdminEndpoint$RSGroupAdminServiceImpl.removeRSGroup(RSGroupAdminEndpoint.
java:254)
        at org.apache.hadoop.hbase.protobuf.generated.RSGroupAdminProtos$RSGroupAdminService.callMethod(RSGroupAdminProtos.
java:13885)
        at org.apache.hadoop.hbase.master.MasterRpcServices.execMasterService(MasterRpcServices.java:816)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)

Remove a RegionServer group.

  hbase> remove_rsgroup 'my_group'


Took 0.0049 seconds
hbase(main):019:0> move_tables_rsgroup 'default', ['t4', 'temp1']
Took 2.2508 seconds
```

Попытка удаления группы с серверами

```
hbase(main):020:0> remove_rsgroup 'test_group'

ERROR: org.apache.hadoop.hbase.constraint.ConstraintException: RSGroup test_group has 1 servers; you must remove these servers
from the RSGroup beforethe RSGroup can be removed.
        at org.apache.hadoop.hbase.rsgroup.RSGroupAdminServer.removeRSGroup(RSGroupAdminServer.java:466)
        at org.apache.hadoop.hbase.rsgroup.RSGroupAdminEndpoint$RSGroupAdminServiceImpl.removeRSGroup(RSGroupAdminEndpoint.
java:254)
        at org.apache.hadoop.hbase.protobuf.generated.RSGroupAdminProtos$RSGroupAdminService.callMethod(RSGroupAdminProtos.
java:13885)
        at org.apache.hadoop.hbase.master.MasterRpcServices.execMasterService(MasterRpcServices.java:816)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)

Remove a RegionServer group.

  hbase> remove_rsgroup 'my_group'


Took 0.0042 seconds
hbase(main):022:0> move_servers_rsgroup 'default', ['bds-adh-3.ru-central1.internal:16020']
Took 1.0126 seconds
```

Удаление группы без серверов и таблиц

```
hbase(main):023:0> remove_rsgroup 'test_group'
Took 0.1284 seconds
hbase(main):024:0> list_rsgroups
NAME                     SERVER / TABLE
 default                 server bds-adh-1.ru-central1.internal:16020
                         server bds-adh-2.ru-central1.internal:16020
                         server bds-adh-3.ru-central1.internal:16020
                         table hbase:meta
                         table hbase:acl
                         table SYSTEM.CATALOG
                         table ns1:temp2
                         table t4
                         table t5
                         table t6
                         table hbase:labels
                         table temp1
                         table hbase:namespace
                         table SYSTEM.SEQUENCE
                         table SYSTEM.LOG
                         table SYSTEM.FUNCTION
                         table SYSTEM.MUTEX
                         table SYSTEM.STATS
                         table hbase:rsgroup
1 row(s)
Took 0.0097 seconds
```

remove_servers_rsgroup

Константин Алпашкин

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Удаляет определенные выведенные из эксплуатации (decommissioned) серверы из группы серверов. Онлайн-серверы запрещены для этой операции.

Использование

```
remove_servers_rsgroup <region_servers_array[]>
```

Аргументы

Параметр	Описание
region_servers_array	Массив строк, указанных в формате `'<region_server_name>:<port_number>'`, где `<region_server_name>` — имя Region Server; `<port_number>` — номер порта

Примеры

```
hbase(main):013:0> remove_servers_rsgroup 'bds-adh-3.ru-central1.internal:16020'

ERROR: org.apache.hadoop.hbase.constraint.ConstraintException: Server bds-adh-3.ru-central1.internal:16020 is an online server,
not allowed to remove.
        at org.apache.hadoop.hbase.rsgroup.RSGroupAdminServer.checkForDeadOrOnlineServers(RSGroupAdminServer.java:693)
        at org.apache.hadoop.hbase.rsgroup.RSGroupAdminServer.removeServers(RSGroupAdminServer.java:609)
        at org.apache.hadoop.hbase.rsgroup.RSGroupAdminEndpoint$RSGroupAdminServiceImpl.removeServers(RSGroupAdminEndpoint.
java:350)
        at org.apache.hadoop.hbase.protobuf.generated.RSGroupAdminProtos$RSGroupAdminService.callMethod(RSGroupAdminProtos.
java:13905)
        at org.apache.hadoop.hbase.master.MasterRpcServices.execMasterService(MasterRpcServices.java:816)
        at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
        at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:413)
        at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:130)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:324)
        at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:304)

Remove decommissioned servers from rsgroup.
Dead/recovering/live servers will be disallowed.
Example:
  hbase> remove_servers_rsgroup ['server1:port','server2:port']

Took 0.9030 seconds
```

Параметры Hive on Spark

Константин Алпашкин

Содержание

Назад к содержанию

Удаленный драйвер Spark

В данной статье описаны параметры, используемые для настройки взаимодействия Spark и Hive Metastore. Больше информации о назначении параметров и примерах их использования доступно в разделе Spark и Hive.

Параметр	Описание	Значение по умолчанию
hive.spark.job.monitor.timeout	Время ожидания монитора задач для получения состояния задачи Spark (в секундах)	60
hive.spark.dynamic.partition.pruning	При установке значения в `true` включается динамическое сокращение разделов (Dynamic Partition Pruning, DPP) для движка Spark. Таким образом, join для ключей разделов выполняется путем записи во временный HDFS-файл, который считывается позже при удалении ненужных разделов	false
hive.spark.dynamic.partition.pruning.map.join.only	Аналогично свойству `hive.spark.dynamic.partition.pruning`, однако активирует DPP только если join для секционированной таблицы может быть преобразован в MapJoin	false
hive.spark.dynamic.partition.pruning.max.data.size	Максимальный размер данных (в мегабайтах) для таблицы измерений, которая генерирует информацию о сокращении разделов. Если таблица достигает этого предела, оптимизация отключается	100
hive.spark.exec.inplace.progress	Позволяет обновлять прогресс выполнения задач Spark в терминале	true
hive.spark.use.ts.stats.for.mapjoin	При установке значения `true` оптимизация MapJoin в Hive/Spark использует статистику операторов TableScan в корне дерева операторов вместо родительских операторов ReduceSink оператора Join	false
hive.spark.explain.user	Определяет, показывать ли результат команды `EXPLAIN` на уровне пользователя для запросов Hive on Spark. Если опция включена, записывает в лог выходные данные `EXPLAIN` для запроса на уровне пользователя	false
hive.prewarm.spark.timeout	Время ожидания завершения прогрева исполнителей Spark в случае, если `hive.prewarm.enabled=true` (в миллисекундах)	5000
hive.spark.optimize.shuffle.serde	При установке значения `true` Hive on Spark регистрирует пользовательские сериализаторы для типов данных в случайном порядке. Это приводит к меньшему перемешиванию данных	false
hive.merge.sparkfiles	Объединяет небольшие файлы в конце преобразования Spark DAG	false
hive.spark.use.op.stats	Указывает, использовать ли статистику оператора для определения параллелизма reducer для Hive on Spark. Если установлено значение `false`, Hive использует статистику исходной таблицы для определения параллелизма reducer для всех reducer-задач первого уровня и максимальный параллелизм для всех остальных reducer-задач (второго уровня и выше). Установка значения `false` запускает альтернативный алгоритм для расчета количества разделов на перемешивание Spark. Этот новый алгоритм обычно приводит к увеличению количества разделов за одно перемешивание	true
hive.spark.use.ts.stats.for.mapjoin	При установке значения `true` MapJoin-оптимизация в Hive/Spark будет использовать статистику операторов TableScan в корне дерева операторов вместо родительских операторов ReduceSink оператора Join. Использование значения `true` полезно, когда данные статистики оператора, используемые для обычного join-to-map преобразования, неточны	false
hive.spark.use.groupby.shuffle	При установке значения `true` использует метод `RDD#groupByKey()` для группирования. Если установлено значение `false`, используется метод `RDD#repartitionAndSortWithinPartitions()`. Хотя `groupByKey()` имеет лучшую производительность при выполнении группирования, он может потреблять чрезмерный объем памяти. Использование значения `false` может уменьшить использование памяти, но понижает производительность	true

Удаленный драйвер Spark

Удаленный драйвер Spark — это приложение, которое запускается в кластере Spark и отправляет на выполнения задачи Spark. Является долгоживущим приложением, инициализация которого выполняется при первом запросе текущего пользователя и работающее до закрытия пользовательского сеанса.

Следующие свойства управляют удаленным взаимодействием между драйвером Spark и клиентом Hive.

Параметр	Описание	Значение по умолчанию
hive.spark.client.future.timeout	Время ожидания для запросов от клиента Hive к удаленному драйверу Spark (в секундах)	60
hive.spark.client.connect.timeout	Время ожидания для обратного подключения драйвера Spark к клиенту Hive (в миллисекундах)	1000
hive.spark.client.server.connect.timeout	Время ожидания установления связи между клиентом Hive и удаленным драйвером Spark (в миллисекундах). Проверяется обоими процессами	90000
hive.spark.client.secret.bits	Количество случайных бит в сгенерированном секрете для общения клиента Hive с удаленным драйвером Spark. Округляется в меньшую сторону до ближайшего кратного 8	256
hive.spark.client.rpc.server.address	Адрес хоста HiveServer2, который используется для связи между клиентом Hive и удаленным драйвером Spark	hive.spark.client.rpc.server.address; если недоступно — `localhost`
hive.spark.client.rpc.threads	Максимальное количество потоков для цикла событий RPC удаленного драйвера Spark	8
hive.spark.client.rpc.max.size	Максимальный размер сообщения в байтах для связи между клиентом Hive и удаленным драйвером Spark. По умолчанию равно 50 МБ	52,428,800 байт (50 * 1024 * 1024, или 50 МБ)
hive.spark.client.channel.log.level	Уровень логирования для удаленного драйвера Spark. Возможные значения: `DEBUG`, `ERROR`, `INFO`, `TRACE`, `WARN`. Если не установлено, используется уровень `TRACE`	—

Командная строка Beeline

Константин Алпашкин, Михаил Серов

Содержание

Назад к содержанию

Команды Beeline для Hive
Аргументы команд Beeline

Команды Beeline

!<SQLLine command>	Выполняет SQLLine-команду
!delimiter	Устанавливает разделитель для запросов Beeline. Допустимы разделители, состоящие из нескольких символов, но недопустимы кавычки, слеши и символы `--`. По умолчанию используется символ `;`. Использование: `!delimiter $$` Версия: 3.0.0

Свойства Beeline

fetchsize	Стандарт JDBC позволяет указать количество строк для запроса, получаемых при каждом обходе (round trip) базы данных. Это количество называют объёмом выборки (fetch size). Задание объёма выборки в Beeline переписывает стандартное значение объёма выборки драйвера JDBC и влияет на последующие команды, выполняющиеся в текущей сессии. Значение `-1` передаёт для Beeline инструкцию использовать значение объёма выборки драйвера JDBC по умолчанию. Значение `0` или большее передаётся драйверу JDBC для каждой команды. Любое другое отрицательное значение вызовет ошибку. Использование: `!set fetchsize 200` Версия: 4.0.0

Команды Beeline для Hive

Команды, специфичные для Hive (как и команды интерфейса Hive), можно запускать из Beeline, если используется драйвер Hive JDBC. Используйте `;` (точку с запятой), чтобы завершать команды. Для полных версий аргументов можно использовать префикс `--`.

Hive-команды в Beeline

reset	Сбрасывает конфигурацию до значений по умолчанию
reset <key>	Сбрасывает значение конкретной конфигурационной переменной (key) до её значения по умолчанию. Если вы неправильно укажете название переменной, Beeline не выдаст ошибку
set <key>=<value>	Сбрасывает значение конкретной конфигурационной переменной (key). Если вы неправильно укажете название переменной, Beeline не выдаст ошибку
set	Показывает список конфигурационных переменных, которые изменены пользователем или непосредственно процессами Hive
set -v	Показывает все конфигурационные переменные Hadoop и Hive
add FILE[S] <filepath> <filepath>* add JAR[S] <filepath> <filepath>* add ARCHIVE[S] <filepath> <filepath>*	Добавляет один или несколько файлов, JAR-файлов либо архивов в список ресурсов в распределённом кеше. Больше информации можно найти на странице Hive Resources документации Apache Hive
add FILE[S] <ivyurl> <ivyurl>* add JAR[S] <ivyurl> <ivyurl>* add ARCHIVE[S] <ivyurl> <ivyurl>*	В Hive 1.2.0 добавляет один или несколько файлов, JAR-файлов либо архивов в список ресурсов в распределённом кеше, используя Ivy URL вида *ivy://group:module:version?query_string*. Больше информации можно найти на странице Hive Resources документации Apache Hive
list FILE[S] list JAR[S] list ARCHIVE[S]	Показывает список ресурсов, уже добавленных в распределённый кеш. Больше информации можно найти на странице Hive Resources документации Apache Hive
list FILE[S] <filepath>* list JAR[S] <filepath>* list ARCHIVE[S] <filepath>*	Проверяет, добавлены ли данные ресурсы в распределённый кеш. Больше информации можно найти на странице Hive Resources документации Apache Hive
delete FILE[S] <filepath>* delete JAR[S] <filepath>* delete ARCHIVE[S] <filepath>*	Удаляет ресурсы из распределённого кеша
delete FILE[S] <ivyurl> <ivyurl>* delete JAR[S] <ivyurl> <ivyurl>* delete ARCHIVE[S] <ivyurl> <ivyurl>*	В Hive 1.2.0 удаляет из распределённого кеша ресурсы, которые были добавлены с использованием <ivyurl>. Больше информации можно найти на странице Hive Resources документации Apache Hive
reload	В Hive 0.14.0 оповещает HiveServer2 о любых изменениях JAR-файлов по пути, указанному в конфигурационном параметре hive.reloadable.aux.jars.path (без необходимости перезапускать HiveServer2). Изменения могут включать добавление, удаление или обновление JAR-файлов
dfs <dfs command>	Выполняет команду `dfs`
<query string>	Выполняет запрос Hive и выводит результаты в стандартный вывод

Аргументы команд Beeline

Команды Beeline работают со следующими аргументами:

Аргумент	Описание	Пример использования
-u <database URL>	Устанавливает JDBC URL для соединения. Специальные символы в значениях параметров при необходимости должны быть закодированы в кодировке URL	`$ beeline -u db_URL`
-r	Повторно соединяется с Hive с помощью последнего использованного URL (если при последнем соединении были использованы команды `!connect` и `!save`). Версия: 2.1.0	`$ beeline -r`
-n <username>	Устанавливает имя пользователя, необходимое для подключения	`$ beeline -n <valid_user>`
-p <password>	Устанавливает пароль, необходимый для подключения. Опциональное указание пароля: Начиная с Hive 2.2.0, аргумент после `-p` не обязателен. Если после `-p` пароль не указан, Beeline запросит пароль при подключении. Если пароль указан, то Beeline использует его для подключения	`$ beeline -p <valid_password>`
-d <driver class>	Устанавливает класс драйвера, который нужно использовать для подключения	`$ beeline -d <driver_class>`
-e <query>	Определяет запрос, который нужно выполнить. Запрос заключен в двойные либо одинарные кавычки. Этот аргумент можно задать несколько раз	`$ beeline -e "query_string"`
-f <file>	Указывает файл скрипта, который нужно выполнить. Если скрипт содержит отступы (tabs), обработка запроса приводит к ошибке в версии 0.12.0. Этот баг исправлен в версии 0.13.0	`$ beeline -f <path/to/file>`
-i <file> --init <files>	Указывает файлы для инициализации. Версия: 0.14.0 для одного файла. Версия: 2.1.0: для нескольких файлов	`$ beeline -i <path/to/initfile>`
-w <password file> --password-file <password file>	Указывает путь к файлу, который содержит пароль. Версия: 1.2.0	`$ beeline -u '<JDBC_URL>' -n test_user -w pass`
-a <auth_type> --authType <auth type>	Указывает тип аутентификации, который передаётся JDBC в виде свойства `auth`. Версия: 0.13.0	`$ beeline --authType 'kerberos'`
--property-file <file>	Указывает файл, из которого нужно прочитать конфигурационные свойства. Версия: 2.2.0	`$ beeline --property-file <path/to/file>`
--hiveconf property=value	Устанавливает значение текущего конфигурационного свойства. Свойства, которые указаны в `hive.conf.restricted.list`, не могут быть изменены через `hiveconf` (больше информации можно получить на странице Restricted List and Whitelist документации Apache Hive). Версия: 0.13.0	`$ beeline --hiveconf prop1=value1`
--hivevar name=value	Определяет имя и значение для переменной Hive. Эта опция является специфической для Hive и позволяет задавать на уровне сессии переменные, которые могут использоваться в командах и запросах Hive	`$ beeline --hivevar var1=value1`
--color=[true/false]	Управляет подсветкой вывода. По умолчанию принимает значение `false` Не поддерживается для форматов Separated-Value Output	`$ beeline --color=true`
--showHeader=[true/false]	Добавляет (`true`) или удаляет (`false`) названия столбцов из результатов запроса. По умолчанию принимает значение `true`	`$ beeline --showHeader=false`
--headerInterval=ROWS	Определяет интервал для повторного отображения заголовков столбцов после определённого числа строк, когда `outputformat=table`. По умолчанию принимает значение 100. Не поддерживается для форматов Separated-Value Output	`$ beeline --headerInterval=50`
--fastConnect=[true/false]	Определяет, пропускать (`true`) или выполнять (`false`) стадию создания списка всех таблиц и столбцов для автоматической подстановки после tab в выражениях HiveQL. По умолчанию принимает значение `true`	`$ beeline --fastConnect=false`
--autoCommit=[true/false]	Включает/отключает автоматические коммиты для транзакций. По умолчанию принимает значение `false`	`$ beeline --autoCommit=true`
--verbose=[true/false]	Определяет, отображать (`true`) или скрывать (`false`) подробные сообщения об ошибках и информацию для их анализа (debugging). По умолчанию принимает значение `false`	`$ beeline --verbose=true`
--showWarnings=[true/false]	Управляет отображением предупреждений после выполнения любых команд HiveQL. По умолчанию принимает значение `false`	`$ beeline --showWarnings=true`
--showDbInPrompt=[true/false]	Определяет, отображать (`true`) или скрывать (`false`) имя текущей базы данных при запросе. По умолчанию принимает значение `false`. Версия: 2.2.0	`$ beeline --showDbInPrompt=true`
--showNestedErrs=[true/false]	Управляет отображением вложенных (nested) ошибок. По умолчанию принимает значение `false`	`$ beeline --showNestedErrs=true`
--numberFormat=[pattern]	Определяет форматирование чисел, используя DecimalFormat	`$ beeline --numberFormat="#,###,##0.00"`
--force=[true/false]	Определяет, продолжать (`true`) или остановить (`false`) выполнение скрипта в случае ошибок. По умолчанию принимает значение `false`	`$ beeline--force=true`
--maxWidth=MAXWIDTH	Устанавливает максимальную ширину вывода символов в случае, когда `outputformat=table`. Символы, выходящие за границы этой величины, будут обрезаны. По умолчанию принимает значение из терминала для текущей величины параметра, затем снова принимает значение 80	`$ beeline --maxWidth=150`
--maxColumnWidth=MAXCOLWIDTH	Устанавливает максимальную ширину столбца в случае, когда для `outputformat=table`. По умолчанию принимает значение 50 в версии Hive 2.2.0+ или значение 15 в более ранних версиях	`$ beeline --maxColumnWidth=25`
--silent=[true/false]	Отображает (`true`) или скрывает (`false`) информационные сообщения. Также останавливает или возобновляет отображение сообщений логов для запросов из HiveServer2 (в Hive 0.14 и выше) и команд HiveQL (в Hive 1.2.0 и выше). По умолчанию принимает значение `false`	`$ beeline --silent=true`
--autosave=[true/false]	Управляет автоматическим сохранением настроек. По умолчанию принимает значение `false`	`$ beeline --autosave=true`
--outputformat=[table/vertical/csv/tsv/dsv/csv2/tsv2]	Устанавливает форматирование для отображения результатов. По умолчанию принимает значение `table`. Больше информации о типах форматирования можно найти на странице Separated-value output formats. Версия 0.14.0: добавлены форматы dsv/csv2/tsv2	`$ beeline --outputformat=tsv`
--truncateTable=[true/false]	При установке значения `true` столбцы таблицы будут обрезаться на экране (в случае превышения высоты экрана). Версия: 0.14.0	`$ beeline --truncateTable=true`
--delimiterForDSV=DELIMITER	Устанавливает разделитель для значений при формате вывода DSV. По умолчанию используется символ `|`. Версия: 0.14.0	`$ beeline --delimiterForDSV='*'`
--isolation=LEVEL	Устанавливает уровень изоляции для транзакций `TRANSACTION_READ_COMMITTED` или `TRANSACTION_SERIALIZABLE`. Больше информации доступно на странице Java Connection Interface	`$ beeline --isolation=TRANSACTION_SERIALIZABLE`
--nullemptystring=[true/false]	Определяет необходимость вывода null-значений в виде пустой строки (`true`) либо в виде NULL (`false`). По умолчанию принимает значение `false`. Версия: 0.13.0	`$ beeline --nullemptystring=false`
--incremental=[true/false]	По умолчанию принимает значение `true`, начиная с Hive 2.3. В более ранних версиях значением по умолчанию было `false`. В случае `false` результат выборки помещается в буфер перед отображением, что позволяет оптимизировать размер столбцов при отображении. В случае `true` строки результата отображаются немедленно после загрузки, что приводит к меньшей задержке и меньшему потреблению памяти за счет дополнительного паддинга столбцов при отображении. Установка `--incremental=true` рекомендуется, если вы столкнулись с OutOfMemoryError на стороне клиента (в связи с тем, что загруженный результат слишком большого размера)	`$ beeline --incremental=false`
--incrementalBufferRows=NUMROWS	Устанавливает число строк для буфера при выводе строк в stdout, по умолчанию принимает значение 1000; используется только в случае, если `--incremental=true` и `--outputformat=table`. Версия: 2.3.0	`$ beeline --incrementalBufferRows=1000`
--maxHistoryRows=NUMROWS	Устанавливает максимальное число строк для хранения в истории Beeline Версия: 2.3.0	`$ beeline --maxHistoryRows=1000`
--delimiter=;	Устанавливает разделитель для запросов Beeline. Поддерживаются разделители из нескольких символов, но кавычки, слеши и `--` не поддерживаются. По умолчанию используется `;` Версия: 3.0.0	`$ beeline beeline --delimiter=$$`
--convertBinaryArrayToString=[true/false]	Контролирует отображение двоичных данных столбца в виде строки, используя набор символов платформы по умолчанию. Версия: 3.0.0: Поведение по умолчанию (`false`) предполагает отображение двоичных данных с использованием метода `Arrays.toString(byte[] columnValue)`. Версия: 4.0.0: Двоичные данные столбцов отображаются в виде строк с использованием набора символов UTF-8. Поведение по умолчанию (`false`) предполагает отображение двоичных данных с использованием кодировки Base64 без padding	`$ beeline --convertBinaryArrayToString=true`
--help	Отображает строку помощи (help)	`$ beeline --help`

Команды impala-shell

Евгения Кузина

Назад к содержанию

Вы можете использовать команды, описанные ниже, для передачи запросов демону impalad. Можно ввести команду в командной строке или передать ее в качестве аргумента опции `-q` при запуске impala-shell. Большинство команд передаются демону impalad как операторы SQL. За подробной информацией об их синтаксисе обратитесь к соответствующим разделам статьи Impala SQL Statements.

Команды impala-shell

Название	Описание
alter	Изменяет структуру или настройки таблицы Impala или таблицы, совместно используемой Impala и Hive. Для получения подробной информации обратитесь к статьям ALTER TABLE и ALTER VIEW
compute stats	Отображает информацию о производительности для таблицы. Impala использует эту команду для оптимизации запросов. См. COMPUTE STATS Statement
connect	Подключается к экземпляру impalad. Для соединений команда использует порт по умолчанию `21000`. Если необходимо подключиться к экземпляру impalad, который работает на другом порту, укажите этот порт в качестве параметра команды. См. Использование impala-shell для подключения к Impala
describe	Выводит имена столбцов, типы данных и комментарии для указанной таблицы. Также вы можете выполнить команду `describe formatted`, которая отображает дополнительную информацию, такую как директории данных HDFS, партиции и внутренние свойства таблицы. Кроме того, для вызова команды `describe` можно использовать аббревиатуру `desc`. См. DESCRIBE Statement
drop	Удаляет объект, а в некоторых случаях также удаляет ассоциированные с ним файлы данных. За дополнительной информацией обратитесь к статьям DROP TABLE, DROP VIEW, DROP DATABASE и DROP FUNCTION
explain	Выводит план выполнения запроса. Команда `explain` представляет запрос как серию шагов. Этими шагами могут быть операции map/reduce, операции metastore или операции с файловой системой (перемещение, переименование). За дополнительной информацией обратитесь к статьям EXPLAIN Statement и Using the EXPLAIN Plan for Performance Tuning
help	Отображает список всех доступных команд и опций
history	Выводит пронумерованный список команд, выполненных в последних сессиях. История хранится в файле *~/.impalahistory*
insert	Записывает результат запроса в указанную таблицу. Команда может либо перезаписать данные таблицы, либо добавить данные к существующему содержимому. См. INSERT Statement
invalidate metadata	Обновляет метаданные impalad для таблиц. Используйте эту команду после создания, удаления или изменения базы данных, таблицы или партиций в Hive. См. INVALIDATE METADATA Statement
profile	Отображает низкоуровневую информацию о последнем запросе. Команда используется для диагностики производительности. Отчет начинается с той же информации, что и отчеты `EXPLAIN` и `SUMMARY`. См. Using the Query Profile for Performance Tuning
quit	Выход из impala-shell. Необходимо добавить к команде завершающую точку с запятой, чтобы оболочка распознала конец команды
refresh	Обновляет метаданные impalad, содержащие информацию о расположения блоков HDFS, соответствующих файлам данных Impala. Используйте эту команду после загрузки новых файлов данных в таблицу Impala через Hive или с помощью команд HDFS. См. REFRESH Statement
rerun или @	Выполняет одну из предыдущих команд impala-shell из списка истории команд, отображаемых командой history. Это может быть оператор SQL или команда, специфическая для impala-shell (`quit` или `profile`). Команда принимает целочисленный параметр. Положительное целое число представляет порядковый номер из списка, выводимого командой `history`. Отрицательное целое число `-N` представляет `N-ю` команду с конца списка. Например, `-1` означает последнюю команду. Команды, которые выполняются повторно, не создают новых записей в списке команды `history`
select	Производит выборку данных из таблиц. Результат может быть отправлен на вывод консоли, в файл или может быть частью другого запроса. См. SELECT Statement
set	Устанавливает параметры запроса для сессии impala-shell. Доступные параметры перечислены на странице Query Options for the SET Statement. Эти параметры используются для настройки запросов и устранения проблем с ними. Синтаксис команды: `set <option>=<value>`. Если вам нужно отобразить текущие параметры запроса, запустите `set` без параметров. Чтобы изменить значение опции, выполните команду `set` с новым значением. Чтобы восстановить значение по умолчанию, используйте команду `unset`. Команда `set` не применяет новые значения опций, если impala-shell не будет подключена к экземпляру impalad. Вы также можете использовать оператор `set` как оператор SQL. См. SET Statement
shell	Позволяет выполнить команду в командной строке операционной системы, не выходя из impala-shell. Можно использовать символ `!` в качестве сокращения для команды `shell`. Чтобы вставить комментарии в команды `source` или `!`, используйте символ комментария командной строки операционной системы `#` перед строкой, которую нужно закомментировать
show	Отображает данные metastore для объектов (баз данных, таблиц и функций), созданных и доступных через Impala или Hive. См. SHOW Statement
source или src	Команда принимает в качестве параметра путь к файлу локальной файловой системы и выполняет инструкции из него. Файл может содержать операторы SQL и команды impala-shell, включая также другие команды `source`. Команда `source` позволяет выполнять пакетные операции так же, как и вызов impala-shell с опцией `-f`, но в интерактивном режиме. Каждая команда или оператор в файле, кроме последнего, должны заканчиваться точкой с запятой. См. Running Commands and SQL Statements in impala-shell
summary	Отображает сводную информацию об операциях, производимых на разных этапах выполнения запроса. Команда предоставляет информацию более высокого уровня, чем данные, отображаемые командой `explain`. См. Using the SUMMARY Report for Performance Tuning. Вы также можете видеть постоянно обновляемый отчет со сводной информацией во время выполнения запроса. См. LIVE_SUMMARY Query Option
unset	Удаляет значение параметра запроса, указанное пользователем, и возвращает параметру его значение по умолчанию. Информацию о параметрах, для которых доступна эта команда, можно найти по ссылке Query Options for the SET Statement. Вы также можете удалить подстановочные переменные. Для этого используйте следующую запись: `unset var:<variable_name>`
use	Указывает базу данных, для которой будут выполняться последующие команды. Команда `use` позволяет избежать использования полных имен, когда нужно получить доступ к базе данных, отличной от базы данных по умолчанию. См. USE Statement
version	Отображает информацию о версии Impala

MapReduce CLI

Константин Алпашкин

Содержание

Назад к содержанию

[Пользовательские команды](#)

[Административные команды](#)

Все MapReduce-команды вызываются скриптом *bin/mapred*. Запуск скрипта *mapred* без аргументов выводит описание для всех команд.

Использование скрипта показано ниже:

```
$ mapred [SHELL_OPTIONS] COMMAND [GENERIC_OPTIONS] [COMMAND_OPTIONS]
```

В Hadoop есть инфраструктура парсинга параметров, в которой используются общие параметры анализа, а также запуск классов.

Базовая структура MapReduce shell

Параметры команды	Описание
SHELL_OPTIONS	Общие shell-параметры
GENERIC_OPTIONS	Общие shell-параметры, поддерживаемые несколькими командами
COMMAND_OPTIONS	Различные команды с параметрами

Пользовательские команды

Команда	Описание
archive	Создает Hadoop-архив
archives-logs	Объединяет логи YARN в Hadoop-архив
classpath	Выводит classpath
distcp	Используется для копирования больших файлов и каталогов внутри и между кластерами
job	Позволяет работать с MapReduce-задачами
pipes	Запускает pipe-задачу
queue	Позволяет взаимодействовать с очередями и получать информацию об очередях задач
version	Выводит текущую версию Hadoop
envvars	Выводит вычисленные переменные окружения Hadoop

Административные команды

Команда	Описание
historyserver	Запускает JobHistoryServer
hsadmin	Запускает клиент MapReduce HSAdmin
frameworkuploader	Агрегирует JAR-файлы фреймворка

archive

Константин Алпашкин

Назад к содержанию

Позволяет создавать Hadoop-архивы (*.har*).

Использование показано ниже:

```
$ hadoop archive -archiveName name -p <parent> [-r <replication_factor>] <src>* <dest>
```

Аргументы

-archiveName <name>	Имя для архива
-p <parent>	Указывает относительный путь, где файлы должны быть заархивированы
-r <replication_factor>	Указывает желаемый фактор репликации (replication factor)
<src>	Исходный каталог для включения в архив
<dest>	Каталог для сохранения архива

Пример:

```
$ hadoop archive -archiveName myArchive.har -p /myArcguves
```

archive-logs

Константин Алпашкин

Назад к содержанию

Для кластеров с большим количеством YARN-логов логи можно объединить в Hadoop-архивы, чтобы уменьшить количество небольших файлов и, следовательно, нагрузку на NameNode. Данная команда позволяет выполнить архивацию.

Агрегированные логи в Hadoop-архивах доступны для чтения с помощью Job History Server и команды YARN `logs`.

Использование показано ниже:

```
$ mapred archive-logs
```

Аргументы

-force	Принудительно создает рабочую директорию в случае, если такая директория уже была обнаружена. Этот параметр стоит использовать, если вы уверены, что не запущен другой экземпляр
-maxEligibleApps <n>	Максимальное количество приложений для обработки, которые подходят для архивации. По умолчанию используется `-1` (все приложения)
-maxTotalLogsSize <megabytes>	Максимальный общий размер логов, необходимый для архивирования (в мегабайтах). По умолчанию используется `1024`
-memory <megabytes>	Количество памяти для каждого контейнера (в мегабайтах). По умолчанию используется `1024`
-minNumberLogFiles <n>	Минимальное количество лог-файлов, необходимое для архивирования. По умолчанию используется `20`
-noProxy	Если параметр задан, вся обработка выполняется под пользователем, выполняющим эту команду (или пользователем YARN, если используется DefaultContainerExecutor). Если параметр не задан, вся обработка выполняется под пользователем-владельцем этого приложения; если пользователь, выполняющий эту команду, не имеет прав на имперсонацию под нужным пользователем, команда не будет работать
-help	Выводит справочную информацию

classpath

Константин Алпашкин

Назад к содержанию

Выводит classpath, который содержит JAR-файлы Hadoop и необходимые библиотеки. При вызове без аргументов выводит classpath, установленный командными скриптами, который, вероятно, будет содержать подстановочные знаки (wildcards) в classpath-записях. Дополнительные параметры выводят classpath после резолвинга подстановочных знаков или записывают classpath в манифест JAR-файла. Последнее полезно в средах, где нельзя использовать подстановочные знаки, а развернутый classpath превышает максимально поддерживаемую длину командной строки.

Использование показано ниже:

```
$ mapred classpath [--glob |--jar <path> |-h |--help]
```

Аргументы

--glob	Разворачивает подстановочные знаки
--jar <path/to/.jar>	Записывает classpath-строку в манифест JAR-файла с именем `<path/to/.jar>`
--help	Выводит информацию об использовании

distcp

Константин Алпашкин

Назад к содержанию

Инструмент используется для копирования файлов и каталогов внутри и между кластерами.

Использование показано ниже:

```
$ mapred distcp <src> <dst> [args]
```

Аргументы

-append	Позволяет использовать существующие данные в целевых файлах и добавлять к ним новые данные, если это возможно
-async	Запускает команду `distcp` в асинхронном режиме. Как только Hadoop-задача старовала, работа скрипта завершается
-atomic	Позволяет `distcp` копировать исходные данные во временное местоположение, а затем атомарно перемещать данные из временной локации в окончательное местоположение
-bandwidth <arg>	Указывает пропускную способность для map-задач (в МБ/сек)
-blocksperchunk <arg>	Количество блоков в части (chunk). Если флаг задан, разбивает файл на части для параллельного копирования
-copybuffersize	Размер буфера копирования (в байтах). По умолчанию, используется значение `8192`
-delete	Удаляет файлы, существующие в `<dst>`, но не в `<src>`
-diff <oldSnapshot> <newSnapshot>	Определяет разницу между исходным и целевым файлом и применяет разницу к целевому файлу, чтобы синхронизировать с исходным файлом
-f <urilist_uri>	Указывает путь к файлу со списком URI для копирования
-filelimit <n>	Устанавливает максимальное количество файлов для копирования <= `n`
-filters	Путь к файлу, который содержит список регулярных выражений, по одному в каждой строке, чтобы исключить файлы, соответствующие выражениям
-i	Игнорирует ошибки
-log <path/to/logdir>	Сохраняет логи в *<path/to/logdir>*
-m	Определяет максимальное количество одновременных копий
-numListstatusThreads	Количество потоков, используемых для создания листингов файлов
-overwrite	Если флаг задан, перезаписывает целевые файлы/каталоги
-p <arg>	Сохраняет статус (репликация, размера блока, пользователь, группа, разрешение, хеш-сумма, XATTR и временная метка). Если флаг `-p` задан без `<arg>`, сохраняет статус репликации, размера блоков, пользователя, групп, разрешений, хеш-сумм и временных меток. XATTRs сохраняются, если и исходный, и целевой пути находятся в */.reserved/raw* иерархии (только для HDFS)
-rdiff <newSnapshot> <oldSnapshot>	Позволяет определить изменения в target-объекте с момента создания `<oldSnapshot>` в target-объекте, и применять различия в обратном порядке к target-объекту, а также копировать измененные файлы из исходного `<oldSnapshot>`, чтобы сделать target-объект таким же, как `<oldSnapshot>`
-sizelimit <n>	Флаг устарел. Ограничивает общий размер до значения <= `n` (в байтах)
-skipcrccheck	Определяет, следует ли пропускать CRC-проверки для source- и target-путей
-strategy <arg>	Стратегия копирования, используемая в `distcp`. Возможные значения: `dynamic` и `uniformsize`
-tmp <path/to/dir>	Временная локация, которая будет использоваться для атомарных коммитов
-update	Обновляет target, копируя только отсутствующие файлы или каталоги
-v	Логирует дополнительную информацию (пути, размер) в SKIP/COPY-лог
-xtrack <path>	Сохраняет информацию о недостающих source-файлах по указанному `<path>`

Пример:

```
$ mapred distcp hdfs://nn1:8020/foo/bar hdfs://nn2:8020/bar/foo
```

envvars

Константин Алпашкин

Назад к содержанию

Команда выводит вычисленные переменные среды Hadoop.

Использование показано ниже:

```
$ mapred envvars
```

job

Константин Алпашкин

Назад к содержанию

Команда позволяет взаимодействовать с MapReduce-задачами.

Использование показано ниже:

```
$ mapred job | [GENERIC_OPTIONS] | [-submit <job-file>] | [-status <job-id>] | [-counter <job-id> <group-name> <counter-name>] | [-kill <job-id>] | [-events <job-id> <from-event-#> <#-of-events>] | [-history [all] <jobHistoryFile|jobId> [-outfile <file>] [-format <human|json>]] | [-list [all]] | [-kill-task <task-id>] | [-fail-task <task-id>] | [-set-priority <job-id> <priority>] | [-list-active-trackers] | [-list-blacklisted-trackers] | [-list-attempt-ids <job-id> <task-type> <task-state>] [-logs <job-id> <task-attempt-id>] [-config <job-id> <file>]
```

Аргументы

-submit <job-file>	Запускает задачу
-status <job-id>	Выводит прогресс map/reduce-задач и все счетчики задач
-counter <job-id> | <group-name> | <counter-name>	Выводит значение счетчика
-kill <job-id>	Завершает задачу
-events <job-id> | <from-event-#> | <#-of-events>	Выводит подробную информацию о событиях, полученных от jobtracker в заданном интервале
-history [all] jobHistoryFilejobId [-outfile <file>] [-format human|json]	Выводит сведения о задачах, сведения о неудачных и завершенных задачах. Флаг `all` выводит дополнительные сведения о задачах, такие как количество успешных задач, попытки выполнения каждой задачи, счетчики задач и так далее. `-outfile <file>` позволяет направить вывод в файл (вместо stdout). Формат по умолчанию удобочитаемый, но его также можно изменить на JSON с помощью флага `-format`
-list [all]	Отображает незавершенные задачи. `-list all` отображает все задачи
-kill-task <task-id>	Завершает задачу (kill). Задачи, завершенные таким образом, НЕ считаются неудачными попытками
-fail-task <task-id>	Завершает задачу (fail). Задачи, завершенные таким образом, считаются неудачными попытками
-set-priority <job-id> <priority>	Изменяет приоритет задачи. Валидные значения: `VERY_HIGH`, `HIGH`, `NORMAL`, `LOW`, `VERY_LOW`
-list-active-trackers	Выводит список всех активных NodeManagers в кластере
-list-blacklisted-trackers	Выводит список трекеров задач, занесенных в черный список в кластере. Эта команда не поддерживается в кластерах на основе MRv2
list-attempt-ids <job-id> <task-type> <task-state>	Выводит ID попыток на основе предоставленного типа задачи и состояния. Допустимые типы задач: `REDUCE`, `MAP`. Допустимые состояния задачи: `running`, `pending`, `completed`, `failed`, `killed`
-logs <job-id> <task-attempt-id>	Создает дамп логов контейнера для задачи, если `<task-attempt-id>` не указан. В противном случае создает дамп логов задачи с указанным `<task-attempt-id>`. Логи выгружаются в стандартный вывод
-config <job-id> <file>	Загружает файл конфигурации задачи

Пример:

```
$ mapred job -status job_141643275736_2152
```

pipes

Константин Алпашкин

Назад к содержанию

Команда позволяет запускать pipes-задачи

Использование показано ниже:

```
$ mapred pipes [-conf <path>] [-jobconf <key=value>, <key=value>, ...] [-input <path>] [-output <path>] [-jar <jar file>] [-inputformat <class>] [-map <class>] [-partitioner <class>] [-reduce <class>] [-writer <class>] [-program <executable>] [-reduces <num>]
```

Аргументы

-conf <path>	Путь к файлу конфигурации задачи
-jobconf <key=value,key=value,...>	Добавляет/перезаписывает конфигурационные параметры для задачи
-input <path>	Путь к входному каталогу
-output <path>	Путь к выходному каталогу
-jar <jar file>	Указывает имя JAR-файла
-inputformat <class>	Имя InputFormat-класса
-map <class>	Имя Map-класса
-partitioner <class>	Имя Partitioner-класса
-reduce <class>	Имя Reducer-класса
-writer <class>	Имя RecordWriter-класса
-program <executable>	URI к исполняемому файлу
-reduces <num>	Количество reduces

Пример:

```
$ mapred pipes -conf job_463823057_6457
```

queue

Константин Алпашкин

Назад к содержанию

Команда позволяет взаимодействовать с очередями и получать информацию об очереди заданий.

Использование показано ниже:

```
$ mapred queue [-list] | [-info <job-queue-name> [-showJobs]] | [-showacls]
```

Аргументы

-list	Возвращает список очередей задач, настроенных в системе, вместе с информацией о расписании
-info <job-queue-name> [-showJobs]	Отображает информацию об очереди задач и связанную с ней информацию о расписании для конкретной очереди задач. Если указан флаг `-showJobs`, отображается список задач, отправленных в определенную очередь
-showacls	Отображает имя очереди и связанные с ней операции, разрешенные для текущего пользователя. В список входят только те очереди, к которым у пользователя есть доступ

Пример:

```
$ mapred queue -list
```

version

Константин Алпашкин

Назад к содержанию

Команда выводит текущую версию Hadoop.

Использование показано ниже:

```
$ mapred version
```

frameworkuploader

Константин Алпашкин

Назад к содержанию

Команда собирает JAR-файлы фреймворка и загружает их в HDFS в виде архива.

Использование показано ниже:

```
$ mapred frameworkuploader -target <target> args
```

Аргументы

-input <classpath>	Входной classpath, по которому выполняется поиск JAR-файлов для включения в архив
-fs <filesystem>	Тип целевой файловой системы. По умолчанию используется тип файловой системы, заданной `fs.defaultFS`
-target <target>	Целевое расположение для сохранения архива фреймворка, за которым может следовать символ `#` с локализованным alias. Например, `/usr/lib/framework.tar#framework`. Чтобы обеспечить безопасность кластера, важно, чтобы целевой каталог был доступен для чтения всем пользователям и закрыт на запись для всех, кроме администраторов
-blacklist <list>	Указывает массив регулярных выражений, разделенных запятыми, для фильтрации имен JAR-файлов, чтобы исключить их из classpath. Например, это можно использовать для исключения тестовых JAR-файлов или Hadoop-сервисов, которые не нужны для локализации
-whitelist <list>	Указывает массив регулярных выражений, разделенных запятыми, для включения определенных JAR-файлов. Это можно использовать для обеспечения дополнительной безопасности, чтобы ни один внешний источник не мог включить вредоносный код в classpath при запуске инструмента
-nosymlink	Позволяет исключать символические ссылки, указывающие на один и тот же каталог. Например, путь */a/foo.jar* и символическая ссылка */a/bar.jar*, указывающая на */a/foo.jar*, обычно добавляют *foo.jar* и *bar.jar* в архив как отдельные файлы, несмотря на то, что на самом деле это один и тот же файл. Этот флаг позволяет инструменту исключить */a/bar.jar*, поэтому будет добавлена только одна копия файла
-initialReplication <num>	Фактор репликации, который используется для создания архива фреймворка. Допустимо оставить значение по умолчанию `3`
-finalReplication <num>	Инструмент загрузки устанавливает фактор репликации после того, как все блоки собраны и загружены. Если требуется быстрый первоначальный запуск, рекомендуется установить это значение равным количеству введенных в эксплуатацию (commissioned) узлов, деленному на два, но менее `512`
-acceptableReplication <num>	Инструмент ждет, пока архив не будет реплицирован указанное количество раз, прежде чем выйти. Значение должно быть равно фактору репликации, который меньше или равен значению `finalReplication`. Оптимальное значение составляет 90% от значения `finalReplication` для учета отказавших узлов
-timeout <seconds>	Время до достижения `acceptableReplication` перед тем, как инструмент заканчивает работу (в секундах). В противном случае инструмент логирует ошибку и завершает работу

Пример:

```
$ mapred frameworkuploader -timeout 10
```

historyserver

Константин Алпашкин

Назад к содержанию

Команда запускает JobHistoryServer.

Использование показано ниже:

```
$ mapred historyserver
```

hsadmin

Константин Алпашкин

Назад к содержанию

Команда запускает клиент MapReduce HSAdmin для выполнения административных команд JobHistoryServer.

Использование показано ниже:

```
$ mapred hsadmin args
```

Аргументы

-refreshUserToGroupsMappings	Обновляет сопоставления пользователей и групп
-refreshSuperUserGroupsConfiguration	Обновляет сопоставления суперпользователей и прокси-групп
-refreshAdminAcls	Обновляет списки управления доступом (Access Control List, ACL) для администрирования Job History server
-refreshLoadedJobCache	Обновляет загруженный кеш задач для Job History server
-refreshJobRetentionSettings	Обновляет Job History period и настройки очистки задач
-refreshLogRetentionSettings	Обновляет период хранения логов и интервал проверки хранения логов
-getGroups <username>	Возвращает список групп, в которых состоит пользователь `<username>`
-help <cmd>	Отображает справочную информацию для команды `<cmd>` или для всех команд, если не указан `<cmd>`

Пример:

```
$ mapred hsadmin -refreshLogRetentionSettings
```

Ozone CLI

Сергей Тихомиров

Содержание

Назад к содержанию

- Обзор
- Команды bucket
- Команды key
- Команды prefix
- Команды snapshot
- Команды token
- Команды user
- Команды volume

Обзор

Общий вид команды Ozone shell представлен ниже:

```
$ ozone sh <object> <action> [<params>]
```

где:

- `<object>` — объект Ozone, например: бакет (bucket), ключ, префикс, снепшот, пользователь, токен или том.
- `<action>` — действие, которое можно произвести с объектом. Например, для создания тома соответствующим действием является `create`.
- `[params]` — дополнительные параметры, которые необходимы для выполнения некоторых действий.

Для примеров использования CLI-инструмента см. Подключение к Ozone через CLI.

Команды bucket

Данные команды позволяют управлять бакетами внутри тома.

Команда	Описание
addacl	Добавляет списки контроля доступа для бакета
clrquota	Снимает ограничения квот для бакета
create	Создает новый бакет
delete	Удаляет бакет
getacl	Возвращает все списки контроля доступа для бакета
info	Выводит информацию о бакете
link	Создает символическую ссылку между бакетами
list	Выводит информацию о всех бакетах в томе
removeacl	Удаляет списки контроля доступа для бакета
set-replication-config	Устанавливает настройки репликации бакета
setacl	Переписывает существующие списки контроля доступа
setquota	Устанавливает квоты для бакета
update	Устанавливает нового владельца бакета

Команды key

Данные команды позволяют управлять ключами внутри бакета.

Команда	Описание
addacl	Добавляет ACL для ключа
cat	Выводит содержимое указанного ключа Ozone
checksum	Возвращает информацию о контрольной сумме ключа
cp	Копирует существующий ключ в другой в рамках того же бакета
delete	Удаляет ключ
get	Скачивает содержимое ключа с сервера
getacl	Выводит все ACL для заданного ключа
info	Выводит информацию о ключе
list	Выводит информацию о всех ключах в бакете или томе
put	Добавляет ключ с содержимым файла в бакет
removeacl	Удаляет ACL из ключа
rename	Переименовывает существующий ключ
setacl	Устанавливает один или несколько ACL, заменяя имеющиеся

Команды prefix

Данные команды позволяют управлять настройками доступа к префиксам.

Команда	Описание
addacl	Добавляет ACL для префикса
getacl	Выводит все ACL для заданного префикса
removeacl	Удаляет ACL из префикса
setacl	Устанавливает один или несколько ACL, заменяя имеющиеся

Команды snapshot

Данные команды позволяют работать со снепшотами бакетов.

Команда	Описание
create	Создает снепшот бакета
delete	Удаляет снепшот бакета
diff	Сравнивает два снепшота бакета
info	Выводит информацию о снепшоте бакета
list	Выводит информацию о всех снепшотах бакета
listDiff	Перечисляет исполняющиеся задачи SnapshotDiff для бакета

Команды token

Данные команды позволяют управлять токенами делегирования.

Команда	Описание
cancel	Отменяет токен делегирования
get	Создает токен делегирования и сохраняет его зашифрованную версию в файл
print	Выводит информацию о токене делегирования
renew	Обновляет токен делегирования

Команды user

Данные команды позволяют управлять пользователями.

Команда	Описание
assign-admin	Выдает права администратора по access ID
assign	Присваивает пользователю тенант (организацию)
get-secret	Возвращает secret, выданный на ID доступа
info	Выводит информацию о пользователе
list	Выводит список пользователей в тенанте (организации)
revoke-admin	Отзывает у пользователя роль администратора
revoke	Удаляет пользователя из организации (тенанта)
set-secret	Устанавливает secret для ID доступа

Команды volume

Данные команды позволяют управлять томами.

Команда	Описание
addacl	Добавляет списки контроля доступа для тома
clrquota	Снимает ограничения квот для тома
create	Создает новый том
delete	Удаляет том
getacl	Возвращает все списки контроля доступа для тома
info	Выводит информацию о томе
list	Выводит информацию о всех томах
removeacl	Удаляет списки контроля доступа для тома
setacl	Переписывает существующие списки контроля доступа
setquota	Устанавливает квоты для тома
update	Устанавливает нового владельца тома

bucket addacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Добавляет ACL для бакета.

Использование

```
$ ozone sh bucket addacl [-hV] [-s=<store>]
                               -a=<list>
                               <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`
list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где: ▪ `<type>` — вид системной сущности: `user` или `group`. ▪ `<entity>` — пользователь или группа, для которых устанавливаются права доступа. ▪ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие: ▪ `r` — чтение данных из ресурса. ▪ `w` — запись данных в ресурс. ▪ `c` — создание объектов в ресурсе. ▪ `d` — удаление объектов из ресурса. ▪ `l` — перечисление объектов, хранящихся в ресурсе. ▪ `x` — чтение списка контроля доступа к ресурсу. ▪ `y` — изменение списка контроля доступа к ресурсу. ▪ `a` — все вышеперечисленное. ▪ `n` — никаких прав.
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Добавление ACL для бакета:

```
$ ozone sh bucket addacl vol1/bucket1 -s=OZONE -a=user:sergei:all
```

Вывод, подтверждающий добавление:

```
ACL user:sergei:la[ACCESS] added successfully.
```

bucket clrquota

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Снимает ограничения квоты для бакета.

Использование

```
$ ozone sh bucket clrquota [-hV] [--namespace-quota]
                              [--space-quota]
                              <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
--namespace-quota	Определяет, следует ли снимать ограничения на максимальное число ключей в бакете. Новое значение будет равно `-1` по умолчанию
--space-quota	Определяет, следует ли снимать ограничения на максимальный объем занимаемой памяти. Новое значение будет равно `-1` по умолчанию
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка ACL для бакета:

```
$ ozone sh bucket clrquota --namespace-quota --space-quota vol1/bucket1
```

bucket create

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Создает новый бакет в указанном томе.

Использование

```
$ ozone sh bucket create [-ghV] [-k=<bekName>]
                                [-l=<allowedBucketLayout>]
                                [--namespace-quota=<quotaInNamespace>]
                                [--quota=<quotaInBytes>]
                                [-r=<replication>]
                                [-t=<type>]
                                [-u=<ownerName>]
                                <value>
```

Аргументы

Параметр	Описание
bekName	Имя ключа шифрования бакета
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
allowedBucketLayout	Макет бакета. Возможные значения: `OBJECT_STORE`, `LEGACY` и `FILE_SYSTEM_OPTIMIZED`, являющееся значением по умолчанию
quotaInNamespace	Максимальное количество ключей в бакете
quotaInBytes	Максимальная квота (в байтах)
replication	Определение репликации. Валидные значения зависят от типа репликации. Для `RATIS` подойдут `ONE` или `THREE`. В то время как для `EC` должен соблюдаться формат `CODEC-DATA-PARITY-CHUNK_SIZE`, например `rs-3-2-1024k`
type	Тип репликации. Поддерживаемые типы: `RATIS` и `EC`
ownerName	Владелец бакета. Если не указан явно, то по умолчанию владельцем будет являться пользователь, создающий бакет
-g, --enforcegdpr	Указывает, что создаваемый бакет должен соответствовать GDPR
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Создание бакета с ограниченной квотой:

```
$ ozone sh bucket create --quota=1073741824 vol1/bucket3
```

Создание бакета со специальными настройками репликации:

```
$ ozone sh bucket create -t=RATIS -r=ONE vol1/bucket4
```

bucket delete

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Удаляет бакет.

Использование

```
$ ozone sh bucket delete [-hrVy] <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-r	Определяет, следует ли выполнять удаление рекурсивно
-y	Определяет, следует ли проводить удаление без интерактивного подтверждения от пользователя
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Удаление бакета:

```
$ ozone sh bucket delete vol1/bucket1
```

bucket getacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит все ACL для заданного бакета.

Использование

```
$ ozone sh bucket getacl [-hV] [--source] [-s=<store>] <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`
--source	Определяет, ACL для какого бакета выводить: связанного (link) (`--source=false`) или исходного (source) (`--source=true`)
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Вывод всех списков контроля доступа для исходного бакета:

```
$ ozone sh bucket getacl --source=true vol1/bucket2
```

Вывод зависит от присутствующих списков доступа, но должен быть похожим на следующий:

```
[ {
  "type" : "USER",
  "name" : "s_tikhomirov_krb1@AD.RANGER-TEST",
  "aclScope" : "ACCESS",
  "aclList" : [ "ALL" ]
}, {
  "type" : "USER",
  "name" : "sergei",
  "aclScope" : "ACCESS",
  "aclList" : [ "LIST", "ALL" ]
} ]
```

bucket info

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит информацию о бакете.

Использование

```
$ ozone sh bucket info [-hV] <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. Для получения информации о бакете вызовите следующую команду:

```
$ ozone sh bucket info vol1/bucket1
```

Вывод зависит от настроек бакета, но в целом должен быть похож на следующий:

```
{
  "metadata" : { },
  "volumeName" : "vol1",
  "name" : "bucket1",
  "storageType" : "DISK",
  "versioning" : false,
  "usedBytes" : 0,
  "usedNamespace" : 0,
  "creationTime" : "2025-01-22T22:06:04.800Z",
  "modificationTime" : "2025-01-22T22:06:55.834Z",
  "sourcePathExist" : true,
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "bucketLayout" : "FILE_SYSTEM_OPTIMIZED",
  "owner" : "s_tikhomirov_krb1",
  "replicationConfig" : {
    "replicationFactor" : "ONE",
    "requiredNodes" : 1,
    "replicationType" : "RATIS"
  },
  "link" : false
}
```

bucket link

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Создает символическую ссылку (symlink) между бакетами.

Использование

```
$ ozone sh bucket link [-hV] <source> <target>
```

Аргументы

Параметр	Описание
source	URI бакета-источника. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
target	URI целевого бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket2/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket2`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Создание символической ссылки между бакетами:

```
$ ozone sh bucket link vol1/bucket1 vol1/bucket2
```

bucket list

Sergei Tikhomirov

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит информацию о всех бакетах в томе.

Использование

```
$ ozone sh bucket list [-hV] [--has-snapshot]
                             [-p=<prefix>]
                             [-s=<startItem>]
                             [[-l=<limit>] | [-a]]
                             <value>
```

ПРИМЕЧАНИЕ

В данной команде `list` можно сократить до `ls`.

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
prefix	Префикс для фильтрации имен бакетов
startItem	Имя элемента, с которого начать вывод. Выбранный элемент не будет отображен в выдаче
limit	Максимальное число элементов для вывода. Значение по умолчанию — `100`
-a, --all	Определяет, что следует выводить все результаты (активно по умолчанию). Чтобы изменить поведение, используйте параметр `-l`
--has-snapshot	Определяет, что следует выводить только те бакеты, у которых есть активный снепшот
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создано два бакета (`bucket1` и `bucket2`). Для получения информации о всех бакетах вызовите следующую команду:

```
$ ozone sh bucket list vol1
```

Вывод зависит от настроек бакетов, но в целом должен быть похож на следующий:

```
[ {
  "metadata" : { },
  "volumeName" : "vol1",
  "name" : "bucket1",
  "storageType" : "DISK",
  "versioning" : false,
  "usedBytes" : 0,
  "usedNamespace" : 0,
  "creationTime" : "2025-01-22T22:06:04.800Z",
  "modificationTime" : "2025-01-22T22:06:55.834Z",
  "sourcePathExist" : true,
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "bucketLayout" : "FILE_SYSTEM_OPTIMIZED",
  "owner" : "s_tikhomirov_krb1",
  "replicationConfig" : {
    "replicationFactor" : "ONE",
    "requiredNodes" : 1,
    "replicationType" : "RATIS"
  },
  "link" : false
}, {
  "metadata" : { },
  "volumeName" : "vol1",
  "name" : "bucket2",
  "storageType" : "DISK",
  "versioning" : false,
  "usedBytes" : 0,
  "usedNamespace" : 0,
  "creationTime" : "2025-01-22T22:07:32.541Z",
  "modificationTime" : "2025-01-22T22:07:32.541Z",
  "sourcePathExist" : true,
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "bucketLayout" : "FILE_SYSTEM_OPTIMIZED",
  "owner" : "s_tikhomirov_krb1",
  "link" : false
} ]
```

Предположим, что в `vol1` добавлен новый бакет `newbucket`. Чтобы найти его среди многочисленных других бакетов командой `list`, используйте префиксный поиск:

```
$ ozone sh bucket list vol1 -p="new"
```

Соответствующий вывод:

```
[ {
  "metadata" : { },
  "volumeName" : "vol1",
  "name" : "newbucket",
  "storageType" : "DISK",
  "versioning" : false,
  "usedBytes" : 0,
  "usedNamespace" : 0,
  "creationTime" : "2025-02-12T22:51:39.876Z",
  "modificationTime" : "2025-02-12T22:51:39.876Z",
  "sourcePathExist" : true,
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "bucketLayout" : "FILE_SYSTEM_OPTIMIZED",
  "owner" : "s_tikhomirov_krb1",
  "link" : false
} ]
```

bucket removeacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Удаляет ACL из бакета.

Использование

```
$ ozone sh bucket removeacl [-hV] [-s=<store>]
                            -a=<list>
                            <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	▪ `<type>` — вид системной сущности: `user` или `group`.
	▪ `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	▪ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  ▪ `r` — чтение данных из ресурса.
	  ▪ `w` — запись данных в ресурс.
	  ▪ `c` — создание объектов в ресурсе.
	  ▪ `d` — удаление объектов из ресурса.
	  ▪ `l` — перечисление объектов, хранящихся в ресурсе.
	  ▪ `x` — чтение списка контроля доступа к ресурсу.
	  ▪ `y` — изменение списка контроля доступа к ресурсу.
	  ▪ `a` — все вышеперечисленное.
	  ▪ `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Удаление ACL из бакета:

```
$ ozone sh bucket removeacl vol1/newbucket -a=user:sergei:all
```

Подтверждение удаления:

```
ACL user:sergei:la[ACCESS] removed successfully.
```

bucket set-replication-config

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Устанавливает настройки репликации для бакета.

Использование

```
$ ozone sh bucket set-replication-config [-hV] [-r=<replication>]
                                               [-t=<type>]
                                               <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
replication	Определение репликации. Валидные значения зависят от типа репликации. Для `RATIS` подойдут `ONE` или `THREE`. В то время как для `EC` должен соблюдаться формат `CODEC-DATA-PARITY-CHUNK_SIZE`, например `rs-3-2-1024k`
type	Тип репликации. Поддерживаемые типы: `RATIS` и `EC`
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка новых настроек репликации для бакета:

```
$ ozone sh bucket set-replication-config -t=RATIS -r=ONE vol1/bucket2
```

bucket setacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Устанавливает один или несколько ACL, заменяя имеющиеся.

ВАЖНО

Все имеющиеся списки доступа будут удалены, если они не находятся в новых списках доступа.

Использование

```
$ ozone sh bucket setacl [-hV] [-s=<store>]
                              -a=<list>
                              <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	▪ `<type>` — вид системной сущности: `user` или `group`.
	▪ `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	▪ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  ▪ `r` — чтение данных из ресурса.
	  ▪ `w` — запись данных в ресурс.
	  ▪ `c` — создание объектов в ресурсе.
	  ▪ `d` — удаление объектов из ресурса.
	  ▪ `l` — перечисление объектов, хранящихся в ресурсе.
	  ▪ `x` — чтение списка контроля доступа к ресурсу.
	  ▪ `y` — изменение списка контроля доступа к ресурсу.
	  ▪ `a` — все вышеперечисленное.
	  ▪ `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка ACL для бакета:

```
$ ozone sh bucket setacl -s=OZONE -a=user:sergei:all vol1/bucket2
```

Подтверждающий вывод:

```
ACLs set successfully.
```

bucket setquota

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Управляет квотой на используемое бакетом пространство в хранилище.

Использование

```
$ ozone sh bucket setquota [-hV] [--namespace-quota=<quotaInNamespace>]
                              [--quota=<quotaInBytes>]
                              <value>
```

Аргументы

Параметр	Описание
quotaInNamespace	Максимальное количество ключей в бакете
quotaInBytes	Максимальная квота (в байтах)
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка квоты для бакета:

```
$ ozone sh bucket setquota --namespace-quota=5 --quota=1073741824 vol1/bucket3
```

bucket update

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Устанавливает нового владельца бакета.

Использование

```
$  ozone sh bucket update [-hV] [-u=<ownerName>] <value>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
ownerName	Имя нового владельца бакета
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка нового владельца бакета:

```
$ ozone sh bucket update -u=sergei vol1/bucket1
```

В качестве подтверждения команда выводит новую информацию о бакете:

```
{
  "metadata" : { },
  "volumeName" : "vol1",
  "name" : "bucket1",
  "storageType" : "DISK",
  "versioning" : false,
  "usedBytes" : 0,
  "usedNamespace" : 0,
  "creationTime" : "2025-01-22T22:06:04.800Z",
  "modificationTime" : "2025-02-12T23:58:47.979Z",
  "sourcePathExist" : true,
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "bucketLayout" : "FILE_SYSTEM_OPTIMIZED",
  "owner" : "sergei",
  "replicationConfig" : {
    "replicationFactor" : "ONE",
    "requiredNodes" : 1,
    "replicationType" : "RATIS"
  },
  "link" : false
}
```

key addacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Добавляет ACL для ключа.

Использование

```
$ ozone sh key addacl [-hV] [-s=<store>]
                            -a=<list>
                            <value>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	- `<type>` — вид системной сущности: `user` или `group`.
	- `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	- `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  - `r` — чтение данных из ресурса.
	  - `w` — запись данных в ресурс.
	  - `c` — создание объектов в ресурсе.
	  - `d` — удаление объектов из ресурса.
	  - `l` — перечисление объектов, хранящихся в ресурсе.
	  - `x` — чтение списка контроля доступа к ресурсу.
	  - `y` — изменение списка контроля доступа к ресурсу.
	  - `a` — все вышеперечисленное.
	  - `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Добавление ACL для ключа:

```
$ ozone sh key addacl vol1/bucket1/key1 -s=OZONE -a=user:sergei:all
```

Вывод, подтверждающий добавление:

```
ACL user:sergei:la[ACCESS] added successfully.
```

key cat

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит содержимое указанного ключа Ozone.

Использование

```
$ ozone sh key cat [-hV] <value>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. В этом бакете существует ключ `key1`. Для вывода содержимого этого ключа вызовите следующую команду:

```
$ ozone sh key cat vol1/bucket1/key1
```

key checksum

Сергей Тихомиров

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Возвращает информацию о контрольной сумме ключа.

Использование

```
$ ozone sh key checksum [-hV] [-c=<mode>]
                       <value>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
mode	Метод комбинирования контрольных сумм. Возможные значения: `COMPOSITE_CRC` (по умолчанию) и `MD5MD5CRC`
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. В этом бакете существует ключ `key1`. Для вывода информации о контрольной сумме этого ключа вызовите следующую команду:

```
$ ozone sh key checksum vol1/bucket1/key1
```

Вывод команды зависит от настроек ключа, но должен быть похож на следующий:

```
{
  "volumeName" : "vol1",
  "bucketName" : "bucket1",
  "name" : "key1",
  "dataSize" : 14,
  "algorithm" : "COMPOSITE-CRC32",
  "checksum" : "B4B9291A"
}
```

key cp

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Копирует существующий ключ в другой в рамках того же бакета.

Использование

```
$ ozone sh key cp [-hV] [-r=<replication>]
                        [-t=<type>]
                        <value>
                        <fromKey>
                        <toKey>
```

Аргументы

Параметр	Описание
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
replication	Определение репликации. Валидные значения зависят от типа репликации. Для `RATIS` подойдут `ONE` или `THREE`. В то время как для `EC` должен соблюдаться формат `CODEC-DATA-PARITY-CHUNK_SIZE`, например `rs-3-2-1024k`
type	Тип репликации. Поддерживаемые типы: `RATIS` и `EC`
fromKey	Существующий ключ, который следует скопировать
toKey	Имя для нового ключа
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. В этом бакете существует ключ `key1`. Для копирования этого ключа вызовите следующую команду:

```
$ ozone sh key cp vol1/bucket1 key1 key2
```

key delete

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Удаляет ключ.

Использование

```
$ ozone sh key delete [-hV] <value>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Удаление ключа:

```
$ ozone sh key delete vol1/bucket1/key1
```

key get

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Скачивает содержимое ключа с сервера.

Использование

```
$ ozone sh key get [-hV] <value>
                         <fileName>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
fileName	Путь к файлу, в который будет записано содержимое
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. В этом бакете существует ключ `key1`. Для скачивания содержимого этого ключа вызовите следующую команду:

```
$ ozone sh key get vol1/bucket1/key1 testFile1.txt
```

key getacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит все ACL для заданного ключа.

Использование

```
$ ozone sh key getacl [-hV] [-s=<store>] <value>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Вывод всех списков контроля доступа для ключа:

```
$ ozone sh key getacl vol1/bucket1/key1
```

Вывод зависит от присутствующих списков доступа, но должен быть похожим на следующий:

```
[ {
  "type" : "USER",
  "name" : "s_tikhomirov_krb1@AD.RANGER-TEST",
  "aclScope" : "ACCESS",
  "aclList" : [ "ALL" ]
}, {
  "type" : "USER",
  "name" : "sergei",
  "aclScope" : "ACCESS",
  "aclList" : [ "LIST", "ALL" ]
} ]
```

key info

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит информацию о ключе.

Использование

```
$ ozone sh key info [-hV] <value>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. В этом бакете существует ключ `key1`. Для получения информации об этом ключе вызовите следующую команду:

```
$ ozone sh key info vol1/bucket1/key1
```

Вывод зависит от настроек ключа, но в целом должен быть похож на следующий:

```
{
  "volumeName" : "vol1",
  "bucketName" : "bucket1",
  "name" : "key1",
  "dataSize" : 14,
  "creationTime" : "2025-02-23T21:14:28.098Z",
  "modificationTime" : "2025-02-23T21:17:20.170Z",
  "replicationConfig" : {
    "replicationFactor" : "ONE",
    "requiredNodes" : 1,
    "replicationType" : "RATIS"
  },
  "metadata" : { },
  "ozoneKeyLocations" : [ {
    "containerID" : 1,
    "localID" : 115816896921600001,
    "length" : 14,
    "offset" : 0,
    "keyOffset" : 0
  } ],
  "file" : true
}
```

key list

Sergei Tikhomirov

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Выводит информацию о всех ключах в бакете или томе.

Использование

```
$ ozone sh key list [-hV] [-p=<prefix>]
                          [-s=<startItem>]
                          [[-l=<limit>] | [-a]]
                          <value>
```

ПРИМЕЧАНИЕ

В данной команде `list` можно сократить до `ls`.

Аргументы

Параметр	Описание
value	URI тома или бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1`), так и коротким URI (`vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
prefix	Префикс для фильтрации имен ключей
startItem	Имя элемента, с которого начать вывод. Выбранный элемент не будет отображен в выдаче
limit	Максимальное число элементов для вывода. Значение по умолчанию — `100`
-a, --all	Определяет, что следует выводить все результаты (активно по умолчанию). Чтобы изменить поведение, используйте параметр `-l`
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. В этом бакете есть два ключа: `key1` и `key2`. Для получения информации о всех ключах вызовите следующую команду:

```
$ ozone sh key list vol1/bucket1
```

Вывод зависит от настроек ключей, но в целом должен быть похож на следующий:

```
[ {
  "volumeName" : "vol1",
  "bucketName" : "bucket1",
  "name" : "key1",
  "dataSize" : 14,
  "creationTime" : "2025-02-23T21:14:28.098Z",
  "modificationTime" : "2025-02-23T21:17:20.170Z",
  "replicationConfig" : {
    "replicationFactor" : "ONE",
    "requiredNodes" : 1,
    "replicationType" : "RATIS"
  },
  "metadata" : { },
  "file" : true
}, {
  "volumeName" : "vol1",
  "bucketName" : "bucket1",
  "name" : "key2",
  "dataSize" : 14,
  "creationTime" : "2025-02-23T22:16:35.235Z",
  "modificationTime" : "2025-02-23T22:16:36.399Z",
  "replicationConfig" : {
    "replicationFactor" : "ONE",
    "requiredNodes" : 1,
    "replicationType" : "RATIS"
  },
  "metadata" : { },
  "file" : true
} ]
```

Предположим, что теперь в бакет добавлен новый ключ `newkey`. Чтобы найти его среди многочисленных ключей командой `list`, используйте префиксный поиск:

```
$ ozone sh key list vol1/bucket1 -p="new"
```

Соответствующий вывод:

```
[ {
  "volumeName" : "vol1",
  "bucketName" : "bucket1",
  "name" : "newkey",
  "dataSize" : 14,
  "creationTime" : "2025-02-23T22:38:35.625Z",
  "modificationTime" : "2025-02-23T22:38:36.695Z",
  "replicationConfig" : {
    "replicationFactor" : "ONE",
    "requiredNodes" : 1,
    "replicationType" : "RATIS"
  },
  "metadata" : { },
  "file" : true
} ]
```

key put

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Добавляет ключ с содержимым файла в бакет.

Использование

```
$ ozone sh key put [-hV] [--stream]
                        [-r=<replication>]
                        [-t=<type>]
                        <value>
                        <fileName>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
replication	Определение репликации. Валидные значения зависят от типа репликации. Для `RATIS` подойдут `ONE` или `THREE`. В то время как для `EC` должен соблюдаться формат `CODEC-DATA-PARITY-CHUNK_SIZE`, например `rs-3-2-1024k`
type	Тип репликации. Поддерживаемые типы: `RATIS` и `EC`
fileName	Путь к файлу, содержимое которого следует загрузить в ключ
--stream	Указывает, что при выполнении следует использовать Streaming Write Pipeline
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. Чтобы добавить ключ в этот бакет, вызовите следующую команду:

```
$ ozone sh key put vol1/bucket1/key1 testFile.txt --stream
```

key removeacl

Сергей Тихомиров

Содержание

Назад к содержанию

[Определение](#)

[Использование](#)

[Примеры](#)

Определение

Удаляет ACL из ключа.

Использование

```
$ ozone sh key removeacl [-hV] [-s=<store>]
                               -a=<list>
                               <value>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	▪ `<type>` — вид системной сущности: `user` или `group`.
	▪ `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	▪ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  ▪ `r` — чтение данных из ресурса.
	  ▪ `w` — запись данных в ресурс.
	  ▪ `c` — создание объектов в ресурсе.
	  ▪ `d` — удаление объектов из ресурса.
	  ▪ `l` — перечисление объектов, хранящихся в ресурсе.
	  ▪ `x` — чтение списка контроля доступа к ресурсу.
	  ▪ `y` — изменение списка контроля доступа к ресурсу.
	  ▪ `a` — все вышеперечисленное.
	  ▪ `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Удаление ACL из ключа:

```
$ ozone sh key removeacl vol1/bucket1/key1 -a=user:sergei:all
```

Подтверждение удаления:

```
ACL user:sergei:la[ACCESS] removed successfully.
```

key rename

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Переименовывает существующий ключ.

Использование

```
$ ozone sh key rename [-hV] <value>
                        <fromKey>
                        <toKey>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
fromKey	Существующий ключ, который следует переименовать
toKey	Новое имя для ключа
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Пусть в томе `vol1` создан бакет `bucket1`. В этом бакете существует ключ `key1`. Для переименования этого ключа вызовите следующую команду:

```
$ ozone sh key rename vol1/bucket1 key1 key2
```

key setacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Устанавливает один или несколько ACL, заменяя имеющиеся.

ВАЖНО

Все имеющиеся списки доступа будут удалены, если они не находятся в новых списках доступа.

Использование

```
$ ozone sh key setacl [-hV] [-s=<store>]
                            -a=<list>
                            <value>
```

Аргументы

Параметр	Описание
value	URI ключа. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/key1`), так и коротким URI (`vol1/bucket1/key1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	▪ `<type>` — вид системной сущности: `user` или `group`.
	▪ `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	▪ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  ▪ `r` — чтение данных из ресурса.
	  ▪ `w` — запись данных в ресурс.
	  ▪ `c` — создание объектов в ресурсе.
	  ▪ `d` — удаление объектов из ресурса.
	  ▪ `l` — перечисление объектов, хранящихся в ресурсе.
	  ▪ `x` — чтение списка контроля доступа к ресурсу.
	  ▪ `y` — изменение списка контроля доступа к ресурсу.
	  ▪ `a` — все вышеперечисленное.
	  ▪ `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка ACL для ключа:

```
$ ozone sh key setacl -s=OZONE -a=user:sergei:all vol1/bucket1/key1
```

Подтверждающий вывод:

```
ACLs set successfully.
```

prefix addacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Добавляет ACL для префикса.

Использование

```
$ ozone sh prefix addacl [-hV] [-s=<store>]
                              -a=<list>
                              <value>
```

Аргументы

Параметр	Описание
value	URI префикса. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/prefix`), так и коротким URI (`vol1/bucket1/prefix`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	▪ `<type>` — вид системной сущности: `user` или `group`.
	▪ `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	▪ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  ▪ `r` — чтение данных из ресурса.
	  ▪ `w` — запись данных в ресурс.
	  ▪ `c` — создание объектов в ресурсе.
	  ▪ `d` — удаление объектов из ресурса.
	  ▪ `l` — перечисление объектов, хранящихся в ресурсе.
	  ▪ `x` — чтение списка контроля доступа к ресурсу.
	  ▪ `y` — изменение списка контроля доступа к ресурсу.
	  ▪ `a` — все вышеперечисленное.
	  ▪ `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Добавление ACL для префикса:

```
$ ozone sh prefix addacl vol1/bucket1/prefix -s=OZONE -a=user:sergei:all
```

Вывод, подтверждающий добавление:

```
ACL user:sergei:la[ACCESS] added successfully.
```

prefix getacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Выводит все ACL для заданного префикса.

Использование

```
$ ozone sh prefix getacl [-hV] [-s=<store>] <value>
```

Аргументы

Параметр	Описание
value	URI префикса. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/prefix`), так и коротким URI (`vol1/bucket1/prefix`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Вывод всех списков контроля доступа для префикса:

```
$ ozone sh prefix getacl vol1/bucket1/prefix
```

Вывод зависит от присутствующих списков доступа, но должен быть похожим на следующий:

```
[ {
  "type" : "USER",
  "name" : "s_tikhomirov_krb1@AD.RANGER-TEST",
  "aclScope" : "ACCESS",
  "aclList" : [ "ALL" ]
}, {
  "type" : "USER",
  "name" : "sergei",
  "aclScope" : "ACCESS",
  "aclList" : [ "LIST", "ALL" ]
} ]
```

prefix removeacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Удаляет ACL из префикса.

Использование

```
$ ozone sh prefix removeacl [-hV] [-s=<store>]
                                  -a=<list>
                                  <value>
```

Аргументы

Параметр	Описание
value	URI префикса. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/prefix`), так и коротким URI (`vol1/bucket1/prefix`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	▪ `<type>` — вид системной сущности: `user` или `group`.
	▪ `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	▪ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  ▪ `r` — чтение данных из ресурса.
	  ▪ `w` — запись данных в ресурс.
	  ▪ `c` — создание объектов в ресурсе.
	  ▪ `d` — удаление объектов из ресурса.
	  ▪ `l` — перечисление объектов, хранящихся в ресурсе.
	  ▪ `x` — чтение списка контроля доступа к ресурсу.
	  ▪ `y` — изменение списка контроля доступа к ресурсу.
	  ▪ `a` — все вышеперечисленное.
	  ▪ `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Удаление ACL из перфикса:

```
$ ozone sh prefix removeacl vol1/bucket1/prefix -a=user:sergei:all
```

Подтверждение удаления:

```
ACL user:sergei:la[ACCESS] removed successfully.
```

prefix setacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Устанавливает один или несколько ACL, заменяя имеющиеся.

ВАЖНО

Все имеющиеся списки доступа будут удалены, если они не находятся в новых списках доступа.

Использование

```
$ ozone sh prefix setacl [-hV] [-s=<store>]
                              -a=<list>
                              <value>
```

Аргументы

Параметр	Описание
value	URI префикса. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/prefix`), так и коротким URI (`vol1/bucket1/prefix`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`
list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где: ■ `<type>` — вид системной сущности: `user` или `group`. ■ `<entity>` — пользователь или группа, для которых устанавливаются права доступа. ■ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие: ▪ `r` — чтение данных из ресурса. ▪ `w` — запись данных в ресурс. ▪ `c` — создание объектов в ресурсе. ▪ `d` — удаление объектов из ресурса. ▪ `l` — перечисление объектов, хранящихся в ресурсе. ▪ `x` — чтение списка контроля доступа к ресурсу. ▪ `y` — изменение списка контроля доступа к ресурсу. ▪ `a` — все вышеперечисленное. ▪ `n` — никаких прав.
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка ACL для ключа:

```
$ ozone sh key setacl -s=OZONE -a=user:sergei:all vol1/bucket1/key1
```

Подтверждающий вывод:

```
ACLs set successfully.
```

snapshot create

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Создает снепшот бакета.

Использование

```
$ ozone sh snapshot create [-hV] [<snapshotName>]
                                 <value>
```

Аргументы

Параметр	Описание
snapshotName	Название снепшота. Если не указывать, то название будет сгенерировано автоматически
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Создание снепшота бакета:

```
$ ozone sh snapshot create vol1/bucket1 bucket1snapshot
```

snapshot delete

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Удаляет снепшот бакета.

Использование

```
$ ozone sh snapshot delete [-hV] <snapshotName>
                                <value>
```

Аргументы

Параметр	Описание
snapshotName	Название снепшота, который следует удалить
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Удаление снепшота бакета:

```
$ ozone sh snapshot delete vol1/bucket1 bucket1snapshot
```

snapshot diff

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Сравнивает два снепшота бакета.

Использование

```
$ ozone sh snapshot diff [-chV]  [-p=<pageSize>]
                                 [-t=<token>]
                                 <value>
                                 <fromSnapshot>
                                 <toSnapshot>
```

Аргументы

Параметр	Описание
pageSize	Количество изменений, которые следует вернуть
token	Токен продолжения для следующей страницы
fromSnapshot	Более старый снепшот бакета
toSnapshot	Более новый снепшот бакета
-c, --cancel	Запрос прекращения задачи SnapshotDiff. Если состояние задачи не `IN_PROGRESS`, то запрос завершится неудачно
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Создайте первый снепшот бакета:

```
$ ozone sh snapshot create vol1/bucket2 oldsnap
```

Добавьте ключ в бакет:

```
$ ozone sh key put vol1/bucket2/key42 file.txt
```

Создайте второй снепшот бакета:

```
$ ozone sh snapshot create vol1/bucket2 newsnap
```

Запросите сравнение:

```
$ ozone sh snapshot diff vol1/bucket1 oldsnap newsnap
```

Подтверждение, что выполнение задачи началось:

```
Snapshot diff job is IN_PROGRESS. Please retry after 60000 ms.
```

Через 60 секунд повторите прошлую команду. Результат должен быть следующим:

```
Difference between snapshot: oldsnap and snapshot: newsnap
+       ./key42
```

snapshot info

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Выводит информацию о снепшоте бакета.

Использование

```
$ ozone sh snapshot info [-hV] <snapshotName>
                               <value>
```

Аргументы

Параметр	Описание
snapshotName	Название снепшота
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Вывод информации о снепшоте:

```
$ ozone sh snapshot info vol1/bucket2 newsnap
```

Вывод зависит от снепшота, но должен быть похож на следующий:

```
{
  "volumeName" : "vol1",
  "bucketName" : "bucket2",
  "name" : "newsnap",
  "creationTime" : 1740354183187,
  "snapshotStatus" : "SNAPSHOT_ACTIVE",
  "snapshotId" : "958877db-3d6e-4ff6-9ff6-910f3176b8d7",
  "snapshotPath" : "vol1/bucket2",
  "checkpointDir" : "-958877db-3d6e-4ff6-9ff6-910f3176b8d7",
  "referencedSize" : 5,
  "referencedReplicatedSize" : 17,
  "exclusiveSize" : 0,
  "exclusiveReplicatedSize" : 0
}
```

snapshot list

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит информацию о всех снепшотах бакета.

Использование

```
$ ozone sh snapshot list [-hV] [-p=<prefix>]
                               [-s=<startItem>]
                               [[-l=<limit>] | [-a]]
                               <value>
```

Аргументы

Параметр	Описание
prefix	Префикс для фильтрации имен снепшотов
startItem	Имя элемента, с которого начать вывод. Выбранный элемент не будет отображен в выдаче
limit	Максимальное число элементов для вывода. Значение по умолчанию — `100`
-a, --all	Определяет, что следует выводить все результаты (активно по умолчанию). Чтобы изменить поведение, используйте параметр `-l`
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Вывод информации о всех снепшотах бакета:

```
$ ozone sh snapshot list vol1/bucket2
```

Вывод зависит от бакета, но должен быть похож на следующий:

```
 [ {
  "volumeName" : "vol1",
  "bucketName" : "bucket2",
  "name" : "oldsnap",
  "creationTime" : 1740353795458,
  "snapshotStatus" : "SNAPSHOT_ACTIVE",
  "snapshotId" : "e36a629c-b976-4cb1-b8aa-5b4f4beaf5ca",
  "snapshotPath" : "vol1/bucket2",
  "checkpointDir" : "-e36a629c-b976-4cb1-b8aa-5b4f4beaf5ca",
  "referencedSize" : 0,
  "referencedReplicatedSize" : 0,
  "exclusiveSize" : 0,
  "exclusiveReplicatedSize" : 0
}, {
  "volumeName" : "vol1",
  "bucketName" : "bucket2",
  "name" : "s20250223-232331.183",
  "creationTime" : 1740353011183,
  "snapshotStatus" : "SNAPSHOT_ACTIVE",
  "snapshotId" : "c10f2b45-5f40-4b25-ae0c-ce617e5909a0",
  "snapshotPath" : "vol1/bucket2",
  "checkpointDir" : "-c10f2b45-5f40-4b25-ae0c-ce617e5909a0",
  "referencedSize" : 0,
  "referencedReplicatedSize" : 0,
  "exclusiveSize" : 0,
  "exclusiveReplicatedSize" : 0
} ]
```

snapshot listDiff

Сергей Тихомиров

Содержание

Назад к содержанию

- Определение
- Использование
- Примеры

Определение

Перечисляет исполняющиеся задачи SnapshotDiff для бакета.

Использование

```
$ ozone sh snapshot listDiff [-ahV] [<jobStatus>]
                                    <value>
```

Аргументы

Параметр	Описание
jobStatus	Вывод задач, находящихся в указанном статусе. Возможные значения: `queued`, `in_progress`, `done`, `failed` и `rejected`. По умолчанию возвращает только задачи в статусе `in_progress`
-a, --all	Вывод всех задач вне зависимости от статуса
value	URI бакета. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/bucket1/`), так и коротким URI (начинается с наименования тома, например `vol1/bucket1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Вывод всех операций SnapshotDiff для бакета:

```
$ ozone sh snapshot listDiff vol1/bucket2 -a
```

Вывод должен быть похож на следующий:

```
[ {
  "volumeName" : "vol1",
  "bucketName" : "bucket2",
  "fromSnapshot" : "s20250223-232331.183",
  "toSnapshot" : "newsnap",
  "jobStatus" : "DONE"
}, {
  "volumeName" : "vol1",
  "bucketName" : "bucket2",
  "fromSnapshot" : "oldsnap",
  "toSnapshot" : "newsnap",
  "jobStatus" : "DONE"
} ]
```

token cancel

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Отменяет токен делегирования.

Использование

```
$ ozone sh token cancel [-hV] -t=<tokenFile>
```

Аргументы

Параметр	Описание
tokenFile	Путь к файлу, содержащему закодированный токен
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Отмена токена:

```
$ ozone sh token cancel -t=tokenFile
```

Подтверждение отмены выглядит следующим образом:

```
Token canceled successfully.
```

token get

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Создает токен делегирования и сохраняет его зашифрованную версию в файл.

Использование

```
$ ozone sh token get [-hV] [-r=<renewer>]
                           [<uri>]
                           -t=<tokenFile>
```

Аргументы

Параметр	Описание
renewer	Имя пользователя, который сможет обновлять токен
uri	Ресурс, для которого будет издан токен
tokenFile	Путь к файлу, который будет содержать закодированный токен
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Создание токена для бакета и установка пользователя в качестве renewer:

```
$ ozone sh token get -t=bucketKey vol1/bucket1 -r=sergei
```

Вывод должен быть похож на следующий:

```
Successfully get token for service 10.92.40.213:9874,10.92.40.239:9874,10.92.42.15:9874
Kind: OzoneToken, Service: 10.92.40.213:9874,10.92.40.239:9874,10.92.42.15:9874, Ident: (OzoneToken owner=s_tikhomirov_krb1@AD.
RANGER-TEST, renewer=sergei, realUser=, issueDate=2025-02-24T10:11:35.453Z, maxDate=2025-03-03T10:11:35.453Z, sequenceNumber=4,
masterKeyId=1, strToSign=null, signature=null, awsAccessKeyId=null, omServiceId=ozone1-mgr, omCertSerialId=7)
```

token print

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит информацию о токене делегирования.

Использование

```
$ ozone sh token print -t=<tokenFile>
```

Аргументы

Параметр	Описание
tokenFile	Путь к файлу, который будет содержать закодированный токен

Примеры

Вывод информации о токене делегирования:

```
$ ozone sh token print -t=tokenFile
```

Вывод должен быть похож на следующий:

```
Kind: OzoneToken, Service: 10.92.40.213:9874,10.92.40.239:9874,10.92.42.15:9874, Ident: (OzoneToken owner=s_tikhomirov_krb1@AD.
RANGER-TEST, renewer=sergei, realUser=, issueDate=2025-02-24T10:11:35.453Z, maxDate=2025-03-03T10:11:35.453Z, sequenceNumber=4,
masterKeyId=1, strToSign=null, signature=null, awsAccessKeyId=null, omServiceId=ozone1-mgr, omCertSerialId=7)
```

token renew

Сергей Тихомиров

Содержание

Назад к содержанию

[Определение]
[Использование]
[Примеры]

Определение

Обновляет токен делегирования.

Использование

```
$ ozone sh token renew [-hV] -t=<tokenFile>
```

Аргументы

Параметр	Описание
tokenFile	Путь к файлу, содержащему закодированный токен
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Обновление токена:

```
$ ozone sh token renew -t=tokenFile
```

Вывод должен быть похож на следующий:

```
Token renewed successfully, expiry time: 2025-02-25T10:51:37.082Z.
```

user assign-admin

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Определение

Выдает права администратора по access ID.

Использование

```
$ ozone sh user assign-admin [-dhV] [--om-service-id=<omServiceID>]
                                    [-t=<tenantId>]
                                    <accessId>
```

Аргументы

Параметр	Описание
-d, --delegated	Присвоение роли делегированного администратора. По умолчанию присваивается роль неделегированного администратора
omServiceID	ID сервиса в Ozone Manager. Необходимо, если кластер находится в режиме HA
tenantId	Имя тенанта (организации)
accessId	ID доступа
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

user assign

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование

Определение

Присваивает пользователю тенант (организацию).

Использование

```
$ ozone sh user assign [-hV] [--om-service-id=<omServiceID>]
                             -t=<tenantId>
                             <userPrincipal>
```

Аргументы

Параметр	Описание
omServiceID	ID сервиса в Ozone Manager. Необходимо, если кластер находится в режиме HA
tenantId	Имя тенанта (организации)
userPrincipal	Имя пользователя
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

user get-secret

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Определение

Возвращает secret, выданный на ID доступа. Эта команда отличается от `ozone s3 getsecret` тем, что она не станет генерировать секрет, если указанный ID доступа не существует.

Использование

```
$ ozone sh user get-secret [-hV] [--om-service-id=<omServiceID>]
                                <accessId>
```

Аргументы

Параметр	Описание
omServiceID	ID сервиса в Ozone Manager. Необходимо, если кластер находится в режиме HA
accessId	ID доступа, на который был выдан secret
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

user info

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Определение

Выводит информацию о пользователе.

Использование

```
$ ozone sh user info [-hjV] [--om-service-id=<omServiceID>]
                            <userPrincipal>
```

Аргументы

Параметр	Описание
omServiceID	ID сервиса в Ozone Manager. Необходимо, если кластер находится в режиме HA
userPrincipal	Имя пользователя
-j, --json	Вывод результата в формате JSON
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

user list

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Определение

Выводит список пользователей в тенанте (организации).

Использование

```
$ ozone sh user list [-hjV] [--om-service-id=<omServiceID>]
                            [-p=<prefix>]
                            <tenantId>
```

Аргументы

Параметр	Описание
omServiceID	ID сервиса в Ozone Manager. Необходимо, если кластер находится в режиме HA
prefix	Префикс для фильтрации имен пользователей
tenantId	Название организации (тенанта)
-j, --json	Вывод результата в формате JSON
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

user revoke-admin

Сергей Тихомиров

Содержание

Назад к содержанию

- Определение
- Использование

Определение

Отзывает у пользователя роль администратора.

Использование

```
$ ozone sh user revoke-admin [-hV] [--om-service-id=<omServiceID>]
                                  [-t=<tenantId>]
                                  <accessId>
```

Аргументы

Параметр	Описание
omServiceID	ID сервиса в Ozone Manager. Необходимо, если кластер находится в режиме HA
accessId	ID доступа
tenantId	Название организации (тенанта)
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

user revoke

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Определение

Удаляет пользователя из организации (тенанта).

Использование

```
$ ozone sh user revoke-admin [-hV] [--om-service-id=<omServiceID>]
                                  [-t=<tenantId>]
                                  <accessId>
```

Аргументы

Параметр	Описание
omServiceID	ID сервиса в Ozone Manager. Необходимо, если кластер находится в режиме HA
accessId	ID доступа
tenantId	Название организации (тенанта)
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

user set-secret

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Определение

Устанавливает secret для ID доступа.

Использование

```
$ ozone sh user set-secret [-hV] [--om-service-id=<omServiceID>]
                               -s=<secretKey>
                               <accessId>
```

Аргументы

Параметр	Описание
omServiceID	ID сервиса в Ozone Manager. Необходимо, если кластер находится в режиме HA
accessId	ID доступа
secretKey	Secret, который следует установить
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

volume addacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Добавляет ACL для тома.

Использование

```
$ ozone sh volume addacl [-hV] [-s=<store>]
                               -a=<list>
                               <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	- `<type>` — вид системной сущности: `user` или `group`.
	- `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	- `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  - `r` — чтение данных из ресурса.
	  - `w` — запись данных в ресурс.
	  - `c` — создание объектов в ресурсе.
	  - `d` — удаление объектов из ресурса.
	  - `l` — перечисление объектов, хранящихся в ресурсе.
	  - `x` — чтение списка контроля доступа к ресурсу.
	  - `y` — изменение списка контроля доступа к ресурсу.
	  - `a` — все вышеперечисленное.
	  - `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Добавление ACL для тома:

```
$ ozone sh volume addacl vol1 -s=OZONE -a=user:sergei:all
```

Вывод, подтверждающий добавление:

```
ACL user:sergei:la[ACCESS] added successfully.
```

volume clrquota

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Снимает ограничения квоты для тома.

Использование

```
$ ozone sh volume clrquota [-hV] [--namespace-quota]
                              [--space-quota]
                              <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
--namespace-quota	Определяет, следует ли снимать ограничения на максимальное число бакетов в томе. Новое значение будет равно `-1` по умолчанию
--space-quota	Определяет, следует ли снимать ограничения на максимальный объем занимаемой памяти. Новое значение будет равно `-1` по умолчанию
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка ACL для тома:

```
$ ozone sh volume clrquota --namespace-quota --space-quota vol1
```

volume create

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Создает новый том.

Использование

```
$ ozone sh volume create [-hV] [--namespace-quota=<quotaInNamespace>]
                              [--quota=<quotaInBytes>]
                              [-u=<ownerName>]
                              <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
quotaInNamespace	Максимальное количество бакетов в томе
quotaInBytes	Максимальная квота (в байтах)
ownerName	Владелец тома. Если не указан явно, то по умолчанию владельцем будет являться пользователь, создающий том
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Создание тома с ограниченной квотой:

```
$ ozone sh volume create --quota=1073741824 vol1
```

volume delete

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Удаляет том.

Использование

```
$ ozone sh volume delete [-hrVy] [-t=<threadNo>] <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
-r	Определяет, следует ли выполнять удаление рекурсивно
threadNo	Количество потоков, которое следует использовать для рекурсивного удаления
-y	Определяет, следует ли проводить удаление без интерактивного подтверждения от пользователя
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Удаление тома:

```
$ ozone sh volume delete vol1
```

volume getacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит все ACL для заданного тома.

Использование

```
$ ozone sh volume getacl [-hV] [-s=<store>] <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Вывод всех списков контроля доступа для тома:

```
$ ozone sh volume getacl vol1
```

Вывод зависит от присутствующих списков доступа, но должен быть похожим на следующий:

```
[ {
  "type" : "USER",
  "name" : "s_tikhomirov_krb1@AD.RANGER-TEST",
  "aclScope" : "ACCESS",
  "aclList" : [ "ALL" ]
}, {
  "type" : "USER",
  "name" : "sergei",
  "aclScope" : "ACCESS",
  "aclList" : [ "LIST", "ALL" ]
} ]
```

volume info

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит информацию о томе.

Использование

```
$ ozone sh volume info [-hV] <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов

Примеры

Пусть в томе `vol1` создано пять бакетов. Для получения информации о томе вызовите следующую команду:

```
$ ozone sh volume info vol1
```

Вывод зависит от настроек тома, но в целом должен быть похож на следующий:

```
{
  "metadata" : { },
  "name" : "vol1",
  "admin" : "s_tikhomirov_krb1",
  "owner" : "s_tikhomirov_krb1",
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "usedNamespace" : 5,
  "creationTime" : "2025-01-22T21:23:28.796Z",
  "modificationTime" : "2025-02-12T23:08:16.179Z",
  "acls" : [ {
    "type" : "USER",
    "name" : "s_tikhomirov_krb1",
    "aclScope" : "ACCESS",
    "aclList" : [ "ALL" ]
  }, {
    "type" : "USER",
    "name" : "sergei",
    "aclScope" : "ACCESS",
    "aclList" : [ "READ", "WRITE", "CREATE", "LIST", "DELETE", "READ_ACL", "WRITE_ACL", "ALL" ]
  } ],
  "refCount" : 0
}
```

Параметр `usedNamespace` здесь отвечает за количество бакетов в томе.

volume list

Sergei Tikhomirov

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Выводит информацию о всех томах.

Использование

```
$ ozone sh volume list [-hV] [-p=<prefix>]
                             [-s=<startItem>]
                             [-u=<userName>]
                             [[-l=<limit>] | [-a]]
                             <value>
```

ПРИМЕЧАНИЕ

В данной команде `list` можно сократить до `ls`.

Аргументы

Параметр	Описание
prefix	Префикс для фильтрации имен томов
startItem	Имя элемента, с которого начать вывод. Выбранный элемент не будет отображен в выдаче
limit	Максимальное число элементов для вывода. Значение по умолчанию — `100`
-a, --all	Определяет, что следует выводить все результаты (активно по умолчанию). Чтобы изменить поведение, используйте параметр `-l`
userName	Определяет, что следует выводить информацию только о тех томах, владельцем которых является указанный пользователь
value	URI. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/`), так и коротким URI (в данной команде может быть пустым). Вся недостающая информация считывается из конфигурационных файлов

Примеры

Пусть создан том `vol1`. Для получения информации о нем вызовите следующую команду:

```
$ ozone sh volume list
```

Вывод зависит от настроек томов, но в целом должен быть похож на следующий:

```
[ {
  "metadata" : { },
  "name" : "vol1",
  "admin" : "s_tikhomirov_krb1",
  "owner" : "s_tikhomirov_krb1",
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "usedNamespace" : 5,
  "creationTime" : "2025-01-22T21:23:28.796Z",
  "modificationTime" : "2025-02-12T23:08:16.179Z",
  "acls" : [ {
    "type" : "USER",
    "name" : "s_tikhomirov_krb1",
    "aclScope" : "ACCESS",
    "aclList" : [ "ALL" ]
  }, {
    "type" : "USER",
    "name" : "sergei",
    "aclScope" : "ACCESS",
    "aclList" : [ "READ", "WRITE", "CREATE", "LIST", "DELETE", "READ_ACL", "WRITE_ACL", "ALL" ]
  } ],
  "refCount" : 0
} ]
```

volume removeacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Удаляет ACL из тома.

Использование

```
$ ozone sh volume removeacl [-hV] [-s=<store>]
                             -a=<list>
                             <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`

list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где:
	▪ `<type>` — вид системной сущности: `user` или `group`.
	▪ `<entity>` — пользователь или группа, для которых устанавливаются права доступа.
	▪ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие:
	  ▪ `r` — чтение данных из ресурса.
	  ▪ `w` — запись данных в ресурс.
	  ▪ `c` — создание объектов в ресурсе.
	  ▪ `d` — удаление объектов из ресурса.
	  ▪ `l` — перечисление объектов, хранящихся в ресурсе.
	  ▪ `x` — чтение списка контроля доступа к ресурсу.
	  ▪ `y` — изменение списка контроля доступа к ресурсу.
	  ▪ `a` — все вышеперечисленное.
	  ▪ `n` — никаких прав.

-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Удаление ACL из тома:

```
$ ozone sh volume removeacl vol1 -a=user:sergei:all
```

Подтверждение удаления:

```
ACL user:sergei:la[ACCESS] removed successfully.
```

volume setacl

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Устанавливает один или несколько ACL, заменяя имеющиеся.

ВАЖНО

Все имеющиеся списки доступа будут удалены, если они не находятся в новых списках доступа.

Использование

```
$ ozone sh volume setacl [-hV] [-s=<store>]
                              -a=<list>
                              <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
store	Тип хранилища: `OZONE` или `S3`
list	Разделенный запятыми список ACL. ACL должен соответствовать следующему формату: `<type>:<entity>:<permissions>`, где: ■ `<type>` — вид системной сущности: `user` или `group`. ■ `<entity>` — пользователь или группа, для которых устанавливаются права доступа. ■ `<permissions>` — набор букв, определяющий уровни доступа сущности. Доступны следующие составляющие: ▪ `r` — чтение данных из ресурса. ▪ `w` — запись данных в ресурс. ▪ `c` — создание объектов в ресурсе. ▪ `d` — удаление объектов из ресурса. ▪ `l` — перечисление объектов, хранящихся в ресурсе. ▪ `x` — чтение списка контроля доступа к ресурсу. ▪ `y` — изменение списка контроля доступа к ресурсу. ▪ `a` — все вышеперечисленное. ▪ `n` — никаких прав.
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка ACL для тома:

```
$ ozone sh volume setacl -s=OZONE -a=user:sergei:all vol1
```

Подтверждающий вывод:

```
ACLs set successfully.
```

volume setquota

Сергей Тихомиров

Содержание

Назад к содержанию

Определение
Использование
Примеры

Определение

Управляет квотой на используемое томом пространство в хранилище.

Использование

```
$ ozone sh bucket setquota [-hV] [--namespace-quota=<quotaInNamespace>]
                                  [--quota=<quotaInBytes>]
                                  <value>
```

Аргументы

Параметр	Описание
quotaInNamespace	Максимальное количество бакетов в томе
quotaInBytes	Максимальная квота (в байтах)
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка квоты для тома:

```
$ ozone sh volume setquota --namespace-quota=5 --quota=1073741824 vol1
```

volume update

Сергей Тихомиров

Содержание

Назад к содержанию

Определение

Использование

Примеры

Определение

Устанавливает нового владельца тома.

Использование

```
$  ozone sh volume update [-hV] [--user=<ownerName>] <value>
```

Аргументы

Параметр	Описание
value	URI тома. Может быть как полным URI (начинается с `o3://`, например, `o3://hostname:9862/vol1/`), так и коротким URI (`vol1`). Вся недостающая информация считывается из конфигурационных файлов
ownerName	Имя нового владельца тома
-h, --help	Вывод справочного руководства для данной команды
-V, --version	Вывод информации о версии и завершение работы

Примеры

Установка нового владельца тома:

```
$ ozone sh volume update --user=sergei vol1
```

В качестве подтверждения команда выводит новую информацию о томе:

```
{
  "metadata" : { },
  "name" : "vol2",
  "admin" : "s_tikhomirov_krb1",
  "owner" : "sergei",
  "quotaInBytes" : -1,
  "quotaInNamespace" : -1,
  "usedNamespace" : 0,
  "creationTime" : "2025-02-23T20:59:54.218Z",
  "modificationTime" : "2025-02-23T21:00:38.275Z",
  "acls" : [ {
    "type" : "USER",
    "name" : "s_tikhomirov_krb1",
    "aclScope" : "ACCESS",
    "aclList" : [ "ALL" ]
  } ],
  "refCount" : 0
}
```

spark3-shell

Константин Алпашкин

Содержание

Назад к содержанию

Функции преобразования

Функции действия

Уровни персистентного хранения

Функции персистентности

Примеры

Spark shell предоставляет среду для использования функционала Spark. Доступны множество различных команд, которые можно использовать для обработки данных в интерактивной оболочке.

Основной концепцией Apache Spark является **устойчивый распределенный набор данных** (**Resilient Distributed Datasheet**, **RDD**). Это неизменяемая распределенная коллекция данных, разделенная по машинам в кластере.

Функции преобразования

Функции преобразования (transformation functions) — это операции над RDD, такие как `filter()`, `map()` или `union()`, которые порождают новый RDD.

Функция	Описание
map(function)	Возвращает новый RDD, применяя функцию к элементу
filter(function)	Возвращает новый dataset из тех исходных элементов, для которых функция возвращает значение `true`
filterByRange(lower, upper)	Возвращает RDD с элементами в указанном диапазоне
flatMap(function)	Действует подобно `map()`, однако возвращает последовательность вместо значения
reduceByKey(function,[num Tasks])	Агрегирует значения ключа с помощью функции
groupByKey([num Tasks])	Преобразует (K, V) в (K, <iterable V>)
distinct([num Tasks])	Устраняет дубликаты из RDD
mapPartitions(function)	Запускается отдельно на каждом разделе RDD
mapPartitionsWithIndex(function)	Применяет функцию к разделам, отмеченным числовым значением
sample(withReplacement, fraction, seed)	Используется для выборки части данных с использованием заданного случайного числа
union(anotherRDD)	Возвращает новый RDD, содержащий все элементы исходного и `anotherRDD` RDD
intersection(anotherRDD)	Возвращает новый RDD, содержащий пересечение элементов в исходном и `anotherRDD` RDD
cartesian()	Возвращает декартово произведение всех пар элементов
subtract(anotherRDD)	Создает новый RDD, удаляя элементы `anotherRDD` из исходного RDD
join(RDD,[numTasks])	Объединяет два элемента набора данных с общими аргументами. При объединении (A,B) и (A,C) создает новый RDD (A,(B,C))
cogroup(RDD,[numTasks])	Преобразует (A,B) в (A, <iterable B>)

Функции действия

Функции действия (Action functions) — это операции, которые запускают вычисления, такие как `count()`, `first()`, `take(n)` или `collect()`.

Функция	Описание
count()	Возвращает количество элементов в RDD
collect()	Возвращает количество элементов в RDD в виде массива
reduce(function)	Агрегирует элементы данных в RDD, принимая два аргумента и возвращая один
take(n)	Выбирает первые `n` элементов из RDD
foreach(function)	Выполняет функцию для каждого элемента данных в RDD
first()	Получает первый элемент данных RDD
saveastextfile(path)	Записывает содержимое RDD в один или несколько текстовых файлов в локальной файловой системе
takeordered(n, [ordering])	Возвращает первые `n` элементов RDD, используя либо естественный порядок, либо кастомный компаратор

Уровни персистентного хранения

Уровни персистентного хранения (Persistence storage levels) определяют, где и каким образом сохранять или кешировать RDD, DataFrame или Dataset.

Уровень	Описание
MEMORY_ONLY	Значение по умолчанию. Сохраняет RDD в памяти кластера в виде десериализованного Java-объекта
MEMORY_AND_DISK	Сохраняет в виде десериализованного Java-объекта. Если RDD не помещается в памяти кластера, он сохраняет разделы (partitions) на диске и считывает их
MEMORY_ONLY_SER	Сохраняет RDD в виде десериализованного Java-объекта. Эта операция более ЦПУ-затратна
MEMORY_ONLY_DISK_SER	Эта опция аналогична `MEMORY_ONLY_SER`, однако сохраняет данные на диск, если памяти недостаточно
DISC_ONLY	Сохраняет RDD только на диск
MEMORY_ONLY_2, MEMORY_AND_DISK_2 и так далее	Эта опция аналогична другим уровням, за исключением того, что разделы реплицируются на 2 slave-узла

Функции персистентности

Функции персистентности Spark (persistence functions) позволяют сохранять (или кешировать) dataset в памяти между операциями. При сохранении RDD каждый узел кластера сохраняет разделы RDD в памяти и использует их при выполнении операций над dataset (или производными этого dataset). Это увеличивает скорость выполнения последующих операций. Подобное кеширование является ключевым инструментом для итерационных алгоритмов и быстрого интерактивного взаимодействия.

Функция	Описание
cache()	Используется, чтобы избежать ненужных повторных вычислений. Эквивалентно `persist(MEMORY_ONLY)`
persist(<storage_level>)	Сохраняет RDD с заданным уровнем хранения
unpersist()	Помечает RDD как неперсистентный (non-persistent) и удаляет блок из памяти и с диска
checkpoint()	Сохраняет файл в директории checkpoint, так что все ссылки на его родительский RDD будут удалены

Примеры

Чтобы запустить Spark shell, выполните команду:

```
$ ./bin/spark3-shell
```

Ниже приведены примеры операций над RDD на языке Scala.

Чтение файла из локальной файловой системы:

```scala
val data = sc.textFile("dataset1.txt")
```

Создание RDD посредством распараллеливания:

```scala
val num = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)
val Dataset = sc.parallelize(num)
```

Подсчет всех элементов в RDD:

```scala
Dataset.count()
```

Запись выходных/обработанных данных в текстовый файл:

```scala
counts.saveAsTextFile("output")
```

Фильтрация RDD:

```scala
val DFData = data.filter(line => line.contains("one"))
```

spark3-submit

Константин Алпашкин

Содержание

Назад к содержанию

Master URL

Скрипт `spark3-submit` используется для запуска приложений в кластере. Скрипт расположен в директории *bin*.

Пример вызова скрипта `spark3-submit` представлен ниже:

```
./bin/spark3-submit \
  --class <main-class> \
  --master <master-url> \
  --deploy-mode <deploy-mode> \
  --conf <key>=<value> \
  ... # other options
  <application-jar> \
  [application-arguments]
```

Где:

- `--class` — точка входа вашего Spark-приложения (например, `org.apache.spark.examples.SparkPi`).
- `--master` — мастер URL кластера (например, *spark://23.195.26.187:7077*).
- `--deploy-mode` — определяет, следует ли развернуть драйвер на рабочих узлах (значение `cluster`) или локально в качестве внешнего клиента (`client`, значение по умолчанию).
- `--conf` — произвольные конфигурационные свойства Spark в формате `key=value`. Значения, содержащие пробелы, следует заключать в кавычки (`"key1"="value 1"`). Чтобы передать несколько пар конфигураций, их необходимо указать отдельными аргументами (например, `--conf <key>=<value> --conf <key2>=<value2>`).
- `<application-jar>` — путь к JAR-файлу, который содержит код вашего приложения и все зависимости. URL должен быть доступен внутри вашего кластера. К примеру, это могут быть пути *hdfs://** или *file://**, доступные на всех узлах.
- `[application-arguments]` — параметры, которые необходимо передать главному методу главного класса.

Master URL

Мастер URL, передаваемый в Spark, может быть в одном из следующих форматов:

URL	Описание
local	Запускает Spark локально с одним рабочим потоком (без параллелизма)
local[K]	Запускает Spark локально, используя K рабочих потоков (оптимальным значением K является значение, равное количеству ядер ЦПУ)
local[K,F]	Запускает Spark локально, используя K рабочих потоков и F `maxFailures` (больше информации доступно в описании spark.task.maxFailures)
local[*]	Запускает Spark локально с количеством рабочих потоков, равным количеству логических ядер на машине
local[*,F]	Запускает Spark локально с количеством рабочих потоков, равным количеству логических ядер на машине, и F `maxFailures`
local-cluster[N,C,M]	Режим **local-cluster** используется только для unit-тестирования. В этом режиме распределенный кластер эмулируется в JVM, где N — количество worker, C — количество ядер на один worker и M — память на один worker в МиБ
spark://HOST:PORT	Соединяется с указанным Spark-кластером в автономном режиме (standalone mode). Значение PORT должно соответствовать порту, который используется на вашем кластере. По умолчанию используется порт 7077
spark://HOST1:PORT1,HOST2:PORT2	URL для соединения с указанным Spark-кластером в автономном режиме (standalone mode) с использованием резервных мастер-узлов и ZooKeeper. В списке должны быть указаны все мастер-хосты, настроенные для работы с ZooKeeper в режиме высокой доступности. Значение PORT должно соответствовать порту, который используется на вашем кластере. По умолчанию используется порт 7077
mesos://HOST:PORT	URL для соединения с Mesos-кластером. Значение PORT должно соответствовать порту, который используется на вашем кластере. По умолчанию используется порт 5050 Также для соединения с Mesos-кластером, использующим ZooKeeper, используйте `mesos://zk://…`. При использовании `--deploy-mode`, пара HOST:PORT должна указывать на MesosClusterDispatcher
yarn	URL для соединения с YARN-кластером в режиме клиента или кластера, в зависимости от значения параметра `--deploy-mode`. Расположение кластера определяется с помощью переменных `HADOOP_CONF_DIR` или `YARN_CONF_DIR`
k8s://HOST:PORT	URL для соединения с Kubernetes-кластером в режиме клиента или кластера, в зависимости от значения параметра `--deploy-mode`. Значения HOST и PORT относятся к Kubernetes API Server. По умолчанию соединение использует TLS. Чтобы принудительно использовать незащищенное соединение, используйте URI типа `k8s://http://HOST:PORT`

Действия в SSM

Елена Костюченко

Действиями в SSM называются команды, выполняемые с данными при выполнении особых условий, установленных в правилах.

В таблице ниже перечислены все возможные действия в SSM.

Действия в SSM

Действие	Аргументы	Описание
addec	[-policyName $policyName] [-codec $codecName -dataNum $dataNum -parityNum $parityNum -cellSize $cellSize]	Устанавливает запрошенную политику данных или политику EC с заданными параметрами. Например, `RS-6-3-1024k`. Значение `cellSize` в байтах должно быть кратным 1024. SSM поддерживает значения для `cellSize` в формате: `1KB`, `1MB`
allssd	-file $file	Перемещает файл на SSD-диск
alldisk	-file $file	Перемещает файл на дисковое хранилище
append	-file $file [-length $length -bufSize $bufSize]	Добавляет файл с длиной `$length` и размером буфера `$bufSize`
archive	-file $file	Архивирует файл
cache	-file $file [-replica $replica_num]	Сохраняет файл в кеше HDFS. `$replica_num` — количество реплик, которые нужно кешировать. Если количество реплик не указано, SSM использует значение по умолчанию из HDFS или создаст одну реплику, если это директория
checkcompress	-file $file	Показывает состояние сжатия файла. Не применимо к директории
checkec	-file $file	Показывает политику EC для файла или директории
checkstorage	-file $file	Показывает тип хранения файла
checksum	-file $file	Показывает контрольную сумму файла
compact	-file ['$file1','$file2',..] -containerFile $containerFile	Сжимает запрошенные файлы в файл-контейнер. Сжатые файлы не должны иметь пробелов. Контейнеры не могут быть сжаты
compress	-file $file [-bufSize $bufSize] [-codec $codec]	Сжимает файл (не применимо к директории). Если опция `-bufSize` не указана, SSM использует значение, полученное путем деления размера файла на максимальное количество разделений, настроенное в *smartdefault.xml*. Наименьший размер буфера составляет 1 МБ. SSM поддерживает настройку удобочитаемого значения для `bufSize`, например, `1MB`. SSM также поддерживает различные реализации сжатия, включая Lz4, Bzip2, Zlib и snappy. Если опция `-codec` не указана, SSM будет использовать Zlib по умолчанию
concat	-file $src -dest $dest	Объединяет указанные файлы в целевом файле. `$src` принимает список из как минимум двух файлов, разделенных запятой
copy	-file $src -dest $dest [-offset $offset] [-length $length] [-bufSize $size]	Копирует запрошенный файл с дополнительными параметрами: смещение, длина и размер буфера
copy2s3	copy2s3 -file $file -dest $s3_dest	Копирует файл в хранилище S3. Значение `$s3_dest` должно быть URL, начинающимся с `s3a://`
decompress	-file $file [-bufSize $size]	Распаковывает файл (не применимо к директории)
delete	-file $file	Удаляет файл
disableec	-policy $policyName	Отключает указанную политику EC
ec	-file $file [-policy $policyName -bufSize $bufSize]	Изменяет политику EC файла на указанную. Если параметр `-policy` не указан, SSM будет использовать значение по умолчанию, настроенное в HDFS. Если параметр `-bufSize` не указан, SSM будет использовать значение по умолчанию в байтах, эквивалентное 1 МБ. SSM поддерживает значения для `bufSize` в формате: `1KB`, `1MB`
echo	-msg $message	Выводит сообщение
enableec	-policy $policyName	Включает заданную политику EC
list	-file $src	Выводит список всех файлов в директории
listec	—	Выводит список всех политик EC
merge	-file $src -dest $dest [-bufSize $size]	Объединяет файлы в целевой файл. Значение `$src` должно содержать не менее двух файлов, разделенных запятой
onedisk	-file $file	Перемещает одну реплику файла на диск. Не применимо к блокам EC
onessd	-file $file	Перемещает одну реплику файла на SSD. Не применимо к блокам EC
ramdisk	-file $file	Устанавливает политику хранения `LAZY_PERSIST` для директории с заранее настроенным параметром `RAM_DISK`
read	-file $file [-bufSize $size]	Читает файл с указанным размером буфера
removeec	-policy $policyName	Удаляет указанную политику EC
rename	-file $src -dest $dest	Переименовывает файл или директорию
truncate	-file $src -length $length	Обрезает файл до указанного размера
truncate0	-file $src	Обрезает файл до 0 МБ
uncache	-file $file	Удаляет файл из кеша HDFS
uncompact	-containerFile $containerFile	Распаковывает контейнер в исходные файлы
unec	-file $file [-bufSize $bufSize]	Изменяет политику EC для файла на репликацию. Если значение `-bufSize` не указано, SSM будет использовать значение по умолчанию в байтах, эквивалентное 1 МБ. SSM поддерживает значения для `bufSize` в формате: `1KB`, `1MB`
write	-file $file -length $length [-bufSize $size]	Записывает случайные данные в файл с указанным размером и длиной буфера
sleep	-m $millisecond	Приостанавливает работу на указанное количество миллисекунд
sync	-dest $dest -preserve $attributes	Синхронизирует файл в `$dest` с разделенным запятыми списком атрибутов файла: владелец, группа, разрешения, репликация, время изменения
distcp	-file $file -target $target [DistCp additional options]	Использует команду Hadoop distcp для указанных файлов
user defined actions	—	Позволяет реализовать пользовательские действия

Правила в SSM

Елена Костюченко

Назад к содержанию

Правила в SSM состоят из четырех частей:

```
<objects>: <trigger> | <conditions> | <commands>
```

В таблицах ниже описаны возможные значения для объектов, триггеров и условий. Команды описаны в разделе Действия в SSM.

Дополнительная информация о синтаксисе правил доступна в статье Определение правил в SSM, а более сложные примеры использования правил можно найти в статье Примеры использования правил в SSM.

Свойства объекта file

Свойство	Аббревиатура	Описание
accessCount(interval)	ac	Количество доступов к файлу за выбранный временной интервал
accessCountTop(interval,N)	acTop	Верхнее значение N для количества доступов за выбранный временной интервал
accessCountBottom(interval,N)	acBot	Нижнее значение N для количества доступов за выбранный временной интервал
accessCountTopOnStoragePolicy(interval, N, "$StoragePolicy")	acTopSp	Самое верхнее значение N для количества доступов с учетом политики хранения. Поддерживаемые политики хранения HDFS: COLD WARM HOT ONE_SSD ALL_SSD LAZY_PERSIST
accessCountBottomOnStoragePolicy(interval,N, "$StoragePolicy")	acBotSp	Самое нижнее N для количества доступов относительно политики хранения
age	—	Время с момента последнего изменения до настоящего момента
atime	—	Время последнего доступа
blocksize	—	Размер блока файла
ecPolicy	—	Политика erasure coding файла
inCache	—	Параметр, указывающий на нахождение файла в кеш-хранилище
isDir	—	Параметр, указывающий на то, является ли файл директорией
length	—	Размер файла в байтах. Поддерживается только чисто цифровое значение, которое указывает количество байт
mtime	—	Время последнего изменения файла
path	—	Путь к файлу в HDFS
storagePolicy	—	Политика хранения файла
unsynced	—	Параметр, указывающий на то, что файл не синхронизирован

Условие может включать одно из двух возможных значений времени:

- `<time>` — точное время в формате `"yyyy-MM-dd HH:mm:ss:ms"`, `now` (сейчас) или `now+<TimeInterval>` (сейчас+<TimeInterval>). Например, `"2017-07-29 23:00:00"`, `now+7day`.
- `<TimeInterval>` — временной интервал, описанный как число и единица измерения времени (`5sec`, `5min`, `5day`) или в формате `<time> - <time>`, например, `now - "2016-03-19 23:00:00"`.

Условия

Значение	Описание	Пример
at <time>	Выполнять правило в указанное время	at "2017-07-29 23:00:00" at now
every <TimeInterval>	Выполнять правило с указанной частотой	every 1min
from <Time> [To <Time>]	Используется вместе с условием `every` для указания временного диапазона, когда нужно выполнять правило	every 1day from now every 1min from now to now+7day

Условные операторы

Оператор	Описание	Пример
Логический оператор	and, or, not (и, или, не)	accessCount(1min) > 0 and length = 5KB
Цифровой оператор	+, -, *, /, %	acTop - acBot > 0
Оператор сравнения	>, >=, <, <=, ==, !=	accessCount(1min) > 0

YARN CLI

Константин Алпашкин

Содержание

Назад к содержанию

Пользовательские команды
Административные команды

Данный раздел содержит справочную документацию по YARN shell.

Все YARN-команды вызываются скриптом *bin/yarn*. Запуск скрипта *bin/yarn* без аргументов выводит описание для всех команд.

Использование показано ниже:

```
$ yarn [SHELL_OPTIONS] COMMAND [GENERIC_OPTIONS] [SUB_COMMAND] [COMMAND_OPTIONS]
```

В YARN есть инфраструктура парсинга параметров, в которой используются общие параметры парсинга, а также запуск классов.

Базовая структура YARN shell

Параметры команды	Описание
SHELL_OPTIONS	Общие shell-параметры
GENERIC_OPTIONS	Общие shell-параметры, поддерживаемые несколькими командами
COMMAND_OPTIONS	Различные команды с параметрами

Пользовательские команды

Команда	Описание
application	Используется для вывода отчетов приложений, закрытия приложений и управления долго работающими приложениями
applicationattempt	Выводит applicationattempt-отчет
classpath	Выводит classpath
container	Выводит container-отчеты
jar	Запускает JAR-файл. С помощью этой команды пользователи могут упаковать YARN-код в JAR-файл и выполнить его
logs	Создает дамп логов контейнера
node	Выводит node-отчет
queue	Выводит информацию об очередях
top	Отображает YARN-приложения и используемые ими ресурсы
version	Выводит текущую версию Hadoop
envvars	Выводит вычисленные переменные окружения Hadoop

Административные команды

Команда	Описание
daemonlog	Позволяет динамически возвращать/устанавливать уровень логирования в демоне
nodemanager	Запускает NodeManager
proxyserver	Запускает прокси веб-сервер
resourcemanager	Запускает ResourceManager
rmadmin	Запускает ResourceManager admin client
schedulerconf	Обновляет конфигурацию планировщика
scmadmin	Запускает Shared Cache Manager admin client
sharedcachemanager	Запускает Shared Cache Manager
timelineserver	Запускает TimeLineServer
registrydns	Запускает RegistryDNS server

application или app

Константин Алпашкин

Назад к содержанию

Команда используется для печати отчетов о приложениях, остановки приложений (kill) и управления приложениями, работающими в течение длительного времени.

Использование показано ниже:

```
$ yarn application [options]
$ yarn app [options]
```

Аргументы

-appId <applicationId>	Указывает идентификатор приложения для работы
-appStates <states>	Работает с флагом `-list` и позволяет фильтровать приложения на основе списка состояний приложений, разделенных запятыми. Допустимые значения: `ALL`, `NEW`, `NEW_SAVING`, `SUBMITTED`, `ACCEPTED`, `RUNNING`, `FINISHED`, `FAILED`, `KILLED`
-appTags <tags>	Работает с флагом `-list` и позволяет фильтровать приложения на основе списка тегов приложений, разделенных запятыми
-appTags <types>	Работает с флагом `-list` и позволяет фильтровать приложения на основе типов приложений, разделенных запятыми
-changeQueue <queue_name>	Переводит приложение в новую очередь. ApplicationId можно передать с помощью флага `-appId`. Флаг `-movetoqueue` устарел, новый флаг `-changeQueue` выполняет те же функции
-component <component_name> <count>	Работает с флагом `-flex` и позволяет изменить количество компонентов/контейнеров, работающих для приложения или долго работающего сервиса. Поддерживает абсолютные или относительные значения, такие как `+1`, `2` или `-3`
-components <components>	Работает с флагом `-upgrade` и позволяет обновлять указанные компоненты приложения. Если компонентов указано несколько, они должны быть разделены запятыми
-decomission <application_name>	Выводит из эксплуатации экземпляры компонентов для приложения или долго работающего сервиса. Требуется опция `-instances`. Поддерживает флаг `-appTypes` для указания используемой реализации клиента
-destroy <application_name>	Уничтожает сохраненную спецификацию приложения и безвозвратно удаляет все данные приложения. Поддерживает параметр `-appTypes` для указания реализации клиента
-enableFastLaunch	Загружает зависимости ApplicationMaster в HDFS, чтобы ускорить будущие запуски. Поддерживает параметр `-appTypes` для указания реализации клиента
-flex <application_name/ID>	Изменяет количество запущенных контейнеров для компонента приложения или долго работающего сервиса. Требуется опция `-component`. Если указано имя, необходимо указать `-appType`, если только это не `yarn-service` по умолчанию. Если указан идентификатор, будет выполняться поиск `appType`. Поддерживает опцию `-appTypes` для указания реализации клиента
-help	Отображает справочную информацию по всем командам
-instances <component_instances>	Работает с параметром `-upgrade` для запуска обновления указанных экземпляров компонентов приложения. Также работает с параметром `-decomission` для вывода из эксплуатации указанных экземпляров компонентов. Несколько экземпляров должны быть разделены запятыми
-kill <application_ID>	Завершает работу приложения (kill). Можно указать несколько приложений, разделенных запятой
-launch <application_name> <file_name>	Запускает приложение из файла спецификации (сохраняет спецификацию и запускает приложение). Можно указать параметры `-updateLifetime` и `-changeQueue`, чтобы изменить значения, предоставленные в файле. Поддерживает параметр `-appTypes` для указания реализации клиента
-list	Выводит список приложений. Поддерживает опциональный параметр `-appTypes` для фильтрации приложений по типу, `-appStates` для фильтрации приложений по состоянию и `-appTags` для фильтрации приложений по тегам
-movetoqueue <application_ID>	Флаг устарел. Перемещает приложение в другую очередь. Вместо этого используйте флаг `-changeQueue`
-queue <queue_name>	Работает с флагом `-movetoqueue` и позволяет указать очередь для перемещения приложения
-save <application_name> <file_name>	Сохраняет файл спецификации для приложения. Можно указать параметры `-updateLifetime` и `-changeQueue`, чтобы изменить значения, предоставленные в файле. Поддерживает опцию `-appTypes` для указания реализации клиента
-start <application_name>	Запускает ранее сохраненное приложение. Поддерживает опцию `-appTypes` для указания реализации клиента
-status <application_name/ID>	Выводит статус приложения. Если указан идентификатор, выводит общий статус приложения YARN. Если указано имя, выводит статус приложения на основе собственной реализации приложения; необходимо указать параметр `-appTypes`, если не используется тип `yarn-service` по умолчанию
-stop <application_name/ID>	Останавливает приложение (позже приложение можно снова запустить). Если указано имя, необходимо указать `-appType`, если не используется `yarn-service` по умолчанию. Если указан идентификатор, будет выполняться поиск `appType`. Поддерживает опцию `-appTypes` для указания реализации клиента
-updateLifetime <timeout>	Обновляет тайм-аут ожидания приложения, используя текущее время. Идентификатор приложения можно передать с помощью параметра `-appId`. Значение тайм-аута в секундах
-updatePriority <priority>	Обновляет приоритет приложения. Идентификатор приложения можно передать с помощью параметра `-appId`

applicationattempt

Константин Алпашкин

Назад к содержанию

Выводит applicationattempt-отчет.

Использование показано ниже:

```
$ yarn applicationattempt [options]
```

Аргументы

-help	Отображает справочную информацию
-list <application_ID>	Выводит список попыток запуска приложения (application attempts) для указанного приложения
-status <application_attempt_ID>	Выводит статус попытки запуска приложения

classpath

Константин Алпашкин

Назад к содержанию

Выводит classpath, который содержит JAR-файлы Hadoop и необходимые библиотеки. При вызове без аргументов выводит classpath, установленный командными скриптами, который, вероятно, будет содержать подстановочные знаки (wildcards) в classpath-записях. Дополнительные параметры выводят classpath после резолвинга подстановочных знаков или записывают classpath в манифест JAR-файла. Последнее полезно в средах, где нельзя использовать подстановочные знаки, а развернутый classpath превышает максимально поддерживаемую длину командной строки.

Использование показано ниже:

```
$ yarn classpath [--glob |--jar <path> |-h |--help]
```

Аргументы

--glob	Разворачивает подстановочные знаки
--jar <path/to/.jar>	Записывает classpath-строку в манифест JAR-файла с именем `<path/to/.jar>`
--help	Выводит информацию об использовании

container

Константин Алпашкин

Назад к содержанию

Выводит container-отчет.

Использование показано ниже:

```
$ yarn container [options]
```

Аргументы

-help	Отображает справочную информацию
-list <application_attempt_ID>	Выводит список контейнеров для попытки запуска приложения (application attempt)
-status <container_ID>	Выводит статус контейнера

envvars

Константин Алпашкин

Назад к содержанию

Команда выводит вычисленные переменные среды Hadoop.

Использование показано ниже:

```
$ yarn envvars
```

jar

Константин Алпашкин

Назад к содержанию

Запускает JAR-файл. Вы можете упаковать свой YARN-код в JAR-файл и выполнить его с помощью этой команды.

Использование показано ниже:

```
$ yarn jar <jar> [mainClass]
```

Аргументы

<jar>	Путь к JAR-файлу
mainClass	Имя main-класса. Если не указано, main-класс считывается из манифеста JAR

logs

Константин Алпашкин

Назад к содержанию

Создает дамп логов контейнера.

Использование показано ниже:

```
$ yarn logs -applicationId <application_ID> [options]
```

Аргументы

-applicationId <application_ID>	Указывает ID приложения
-appOwner <appOwner>	Указывает владельца приложения (если не указано, используется текущий пользователь)
-containerId <container_ID>	Указывает ID контейнера (обязателен, если указан `-nodeAddress`)
-help	Выводит справочную информацию
-nodeAddress <nodeAddress>	Адрес узла в формате `nodeName:port` (обязателен, если указан `-containerId`)

node

Константин Алпашкин

Назад к содержанию

Выводит node-отчеты.

Использование показано ниже:

```
$ yarn node [options]
```

Аргументы

-all	Работает с `-list` и выводит список всех узлов
-list	Выводит список всех запущенных узлов. Поддерживает необязательный флаг `-states` для фильтрации узлов по статусу, а также флаг `-all` для отображения всех узлов
-states <states>	Работает с `-list` и позволяет фильтровать узлы по состояниям, используя файл, в котором состояния разделены запятой
-status <nodeId>	Выводит status-отчет для определенного узла

queue

Константин Алпашкин

Назад к содержанию

Выводит информацию об очереди.

Использование показано ниже:

```
$ yarn queue [options]
```

Аргументы

-help	Выводит справочную информацию
-status <queueName>	Выводит статус очереди

top

Константин Алпашкин

Назад к содержанию

Отображает состояние приложений, использование памяти и ядра, а также общий процент завершения приложения в терминале. Эта команда аналогична команде `top` Unix, которая отображает процессы Linux.

Использование показано ниже:

```
$ yarn top
```

version

Константин Алпашкин

Назад к содержанию

Команда выводит текущую версию Hadoop.

Использование показано ниже:

```
$ yarn version
```

daemonlog

Константин Алпашкин

Назад к содержанию

Динамически возвращает/устанавливает уровень для лога, который определяется полным именем класса в демоне. По умолчанию команда отправляет HTTP-запрос, но это можно переопределить, используя аргумент `-protocol https` для отправки HTTPS-запросов.

Обратите внимание, что настройка не является постоянной и сбрасывается при перезапуске демона. При выполнении команды отправляется HTTP/HTTPS-запрос внутреннему сервлету Jetty-демона. Поддерживаются следующие демоны:

- Common
    - key management server
- HDFS
    - name node
    - secondary name node
    - data node
    - journal node
    - HttpFS server
- YARN
    - resource manager
    - node manager
    - Timeline server

Использование показано ниже:

```
$ hadoop daemonlog -getlevel <host:port> <classname> [-protocol (http|https)]
$ hadoop daemonlog -setlevel <host:port> <classname> <level> [-protocol (http|https)]
```

Аргументы

-getlevel <host:port> <class_name> [-protocol (httphttps)]	Для демона с адресом `<host:port>` выводит уровень лога, который определяется полным именем класса `<class_name>`. Флаг `-protocol` указывает протокол для соединения
-setlevel <host:port> <class_name> level [-protocol (httphttps)]	Для демона с адресом `<host:port>` задает уровень лога, который определяется полным именем класса `<class_name>`. Флаг `-protocol` указывает протокол для соединения

Пример:

```
$ bin/hadoop daemonlog -setlevel 127.0.0.1:9870 org.apache.hadoop.hdfs.server.namenode.NameNode DEBUG
$ bin/hadoop daemonlog -getlevel 127.0.0.1:9871 org.apache.hadoop.hdfs.server.namenode.NameNode DEBUG -protocol https
```

nodemanager

Константин Алпашкин

Назад к содержанию

Команда запускает NodeManager.

Использование показано ниже:

```
$ yarn nodemanager
```

proxyserver

Константин Алпашкин

Назад к содержанию

Коменда запускает прокси веб-сервер.

Использование показано ниже:

```
$ yarn proxyserver
```

registrydns

Константин Алпашкин

Назад к содержанию

Запускает RegistryDNS-сервер.

Использование показано ниже:

```
$ yarn registrydns
```

resourcemanager

Константин Алпашкин

Назад к содержанию

Команда запускает ResourceManager.

Использование показано ниже:

```
$ yarn resourcemanager [options]
```

Аргументы

-format-state-store	Форматирует RMStateStore. Если флаг задан, очищает RMStateStore и может быть полезен в случаях, когда прошлые приложения больше не нужны. Следует устанавливать только тогда, когда ResourceManager не работает
-remove-application-from-state-store <appId>	Удаляет приложение из RMStateStore. Следует устанавливать только тогда, когда ResourceManager не работает
-format-conf-store	Форматирует YarnConfigurationStore. Следует устанавливать только тогда, когда ResourceManager не работает
-convert-fs-configuration [-y|yarnsiteconfig] [-f|fsconfig] [-r|rulesconfig] [-o|output-directory] [-p|print] [-c|cluster-resource]	Это экспериментальный функционал, не предназначенный для использования в production-среде. Опция находится в разработке, поэтому конвертер не считается законченным. Преобразует конфигурацию Fair Scheduler в конфигурацию Capacity Scheduler. Требуется два обязательных входных файла: ▪ *yarn-site.xml* в формате: `[-y|yarnsiteconfig <path/to/yarn-site.xml>]`. ▪ *fair-scheduler.xml* в формате: `[-f|fsconfig <path/to/fair-scheduler.xml>]`. Эта конфигурация не является обязательной, если в *yarn-site.xml* есть ссылка на *fair-scheduler.xml* через свойство `yarn.scheduler.fair.allocation.file`. Если указаны оба файла, опция `-f` имеет приоритет. Выходной каталог файлов конфигурации также должен быть указан с помощью `[-o|output-directory <path/to/dir>]`. Необязательный файл конфигурации правил также может быть указан в следующем формате: `[-r|rulesconfig <path/to/rulesconfig_file>]`. Формат файла конфигурации правил — *.properties*. Существует дополнительный необязательный параметр `[-p|print]`. Если он задан, конфигурация будет выведена в консоль. При нормальной работе выходные файлы (*yarn-site.xml* и *capacity-scheduler.xml*) сохраняются в указанном выходном каталоге. Параметр ресурса кластера `[-c|cluster-resource <resource>]` необходимо указать, если какая-либо очередь имеет настройку `maxResources` со значением в процентах. Формат строки ресурса такой же, как в *fair-scheduler.xml*

rmadmin

Константин Алпашкин

Назад к содержанию

Команда запускает ResourceManager admin client.

Использование показано ниже:

```
$ yarn rmadmin [options]
```

Аргументы

-refreshQueues	Перезагружает списки управления доступом, состояния и свойства планировщика для очередей. ResourceManager перезагрузит файл конфигурации mapred-очередей
-refreshNodes [-g|graceful <timeout in seconds> -client|server]	Обновляет информацию о хостах в ResourceManager. Здесь `[-g|graceful <timeout_in_seconds> -client|server]` является необязательным; если вы укажете время ожидания, тогда ResourceManager будет ожидать истечения времени ожидания, прежде чем отметить NodeManager как выведенный из эксплуатации (decommissioned). `[-client|server]` указывает, должно ли отслеживание тайм-аута обрабатываться клиентом или ResourceManager. Отслеживание на стороне клиента блокирующее, а отслеживание на стороне сервера — нет. Отсутствие тайм-аута или установка значения `-1` указывает на бесконечный тайм-аут. Известная проблема: отслеживание на стороне сервера будет немедленно прекращено, если произойдет отработка отказа ResourceManager в режиме высокой доступности (High Availability, HA)
-refreshNodesResources	Обновляет ресурсы NodeManagers в ResourceManager
-refreshSuperUsrGroupsConfiguration	Обновляет сопоставления прокси-групп суперпользователя
-refreshUserToGroupsMappings	Обновляет сопоставления пользователей и групп
-refreshAdminAcls	Обновляет списки управления доступом (ACL) для администраторов ResourceManager
-refreshServiceAcl	Перезагружает файл политики авторизации на уровне сервиса
-getGroups <username>	Возвращает группы, к которым принадлежит указанный пользователь
-addToCLusterNodeLabels <"label1(exclusive=true),label2(exclusive=false),label3">	Добавляет метки узлов в кластер. Эксклюзивность по умолчанию `true`
-removeFromClusterNodeLabels <label1,label2,label3> (label splitted by ",")	Удаляет метки узлов кластера
-replaceLabelsOnNode <"node1[:port]=label1,label2 node2[:port]=label1,label2"> [-failOnUnknownNodes]	Заменяет метки на узлах (указание нескольких меток на одном хосте пока не поддерживается). `-failOnUnknownNodes` не является обязательным. Если флаг задан, это приведет к ошибке в случае, если указанные узлы неизвестны
-directlyAccessNodeLabelStore	Флаг устарел и будет удален в последующих версиях. Прямой доступ к хранилищу меток узлов. С этим параметром все операции, связанные с метками узлов, не будут подключаться к ResourceManager. Вместо этого они будут напрямую обращаться/изменять сохраненные метки узлов. По умолчанию установлено значение `false` (доступ через Resource Manager) Если вы настроите `yarn.node-labels.fs-store.root-dir` так, чтобы он указывал на локальный каталог (вместо NFS или HDFS), этот параметр будет работать только тогда, когда команда выполняется на машине, на котором запущен Resource Manager
-refreshClusterMaxPriority	Обновляет максимальный приоритет кластера
-updateNodeResource [NodeID] [MemSize] [vCores] ([OvercommitTimeout])	Обновляет типы ресурсов на определенных узлах. Типы ресурсов — это разделенные запятыми пары ключ/значение любых ресурсов, доступных в Resource Manager. Например, `memory-mb=1024Mi,vcores=1,resource1=2G,resource2=4m`
-transitionToActive [-forceactive] [-forcemanual] <serviceId>	Переводит сервис в активное состояние. Если используется опция `-forcemanual`, рекомендовано иметь target-узел активным, не выполняя проверку, что нет активного узла, Эту команду нельзя использовать, если включен автоматический переход на другой сервис. И хотя возможно переопределить это с помощью параметра `-forcemanual`, используйте его с осторожностью
-transitionToStandby [-forcemanual] <serviceId>	Переводит сервис в Standby-состояние. Эту команду нельзя использовать, если включена автоматическая отработка отказа. И хотя возможно переопределить это с помощью параметра `-forcemanual`, используйте его с осторожностью
-getServiceState <serviceId>	Возвращает состояние указанного сервиса
-getAllServiceState	Возвращает состояние всех сервисов
-checkHealth <serviceId>	Запрашивает у сервиса проверку работоспособности. Инструмент RMAdmin завершит работу с ненулевым кодом выхода, если проверка не пройдена
-help [cmd]	Отображает справочную информацию для указанной команды или для всех команд, если команда не указана

schedulerconf

Константин Алпашкин

Назад к содержанию

Обновляет конфигурацию планировщика. Этот функционал находится в стадии альфа-тестирования и может быть изменен.

Использование показано ниже:

```
$ yarn shedulerconf [options]
```

Аргументы

--add <"queuePath1:key1=val1,key2=val2; queuePath2:key3=val3">	Разделенные точкой с запятой значения добавляемых очередей и их конфигурации. В этом примере добавляется очередь `queuePath1` (полное имя пути) с конфигурационными параметрами `key1=val1` и `key2=val2`. Также добавляется очередь `queuePath2` с конфигурацией `key3=val3`
-remove <"queuePath1;queuePath2">	Разделенные точкой с запятой очереди для удаления. В этом примере удаляются очереди `queuePath1` и `queuePath2` (полные имена путей). Перед удалением очереди должны быть переведены в состояние STOPPED
-update <"queuePath1:key1=val1,key2=val2; queuePath2:key3=val3">	Разделенные точкой с запятой значения очередей, конфигурации которых необходимо обновить. В этом примере задаются параметры конфигурации `key1=val1` и `key2=val2` для очереди `queuePath1` (полное имя пути), а также `key3=val3` для очереди `queuePath2`
-global <key1=val1,key2=val2>	Обновляет глобальные конфигурации планировщика. В этом примере задаются параметры `key1=val1` и `key2=val2` для глобальной конфигурации планировщика

scmadmin

Константин Алпашкин

Назад к содержанию

Запускает Shared Cache Manager admin client.

Использование показано ниже:

```
$ yarn scmadmin [options]
```

Аргументы

-help	Выводит справочную информацию
-runCleanerTask	Запускает задачу очистки

sharedcachemanager

Константин Алпашкин

Запускает Shared Cache Manager.

Использование показано ниже:

```
$ yarn sharedcachemanager
```

timelineserver

Константин Алпашкин

Назад к содержанию

Запускает TimeLineServer.

Использование показано ниже:

```
$ yarn timelineserver
```

Конфигурационные параметры

Константин Алпашкин

Содержание

Назад к содержанию

- Airflow
- Core configuration
- Flink
- Flink2
- HBase
- HDFS
- Hive
- HUE
- Impala
- Kyuubi
- MySQL
- Ozone
- Solr
- Spark
- Spark3
- SSM
- Trino
- YARN
- Zeppelin
- ZooKeeper
- Monitoring

Данный раздел содержит описание параметров сервисов ADH, которые вы можете настроить с помощью ADCM. Больше информации о процессе настройки сервисов доступно в соответствующих разделах: Online-установка, Offline-установка.

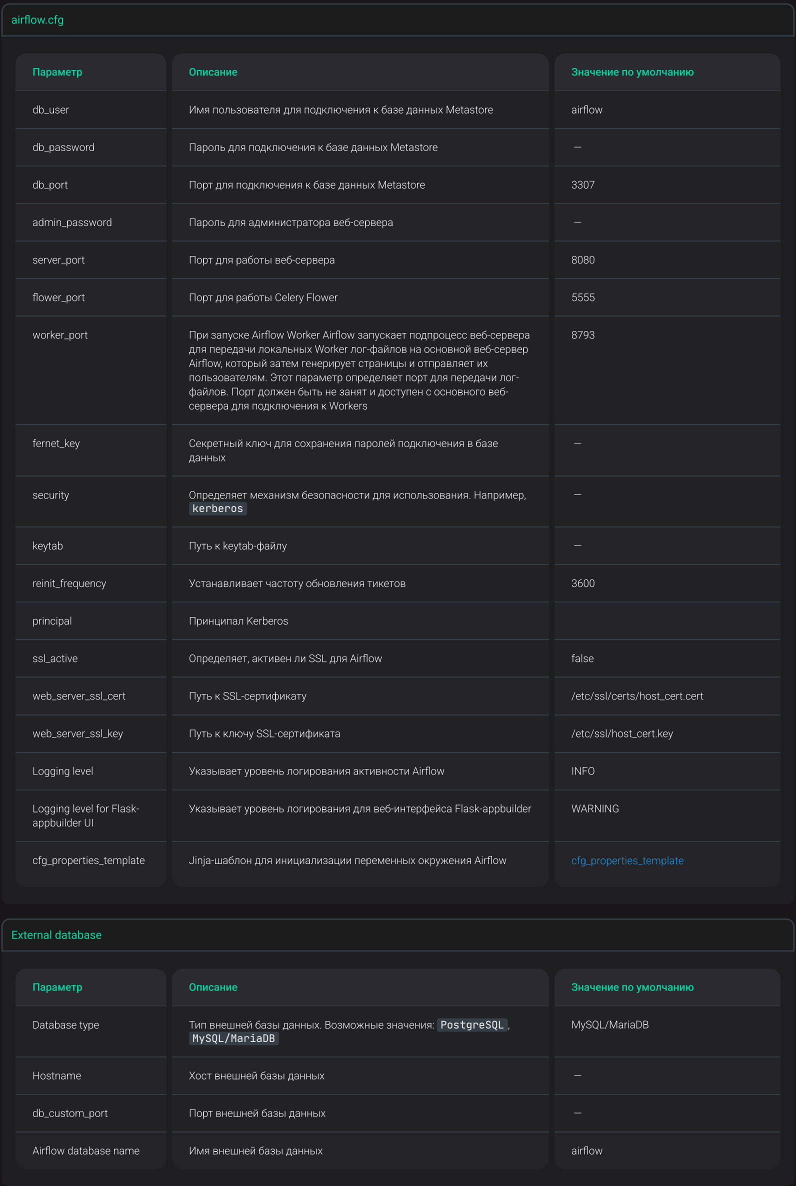
ПРИМЕЧАНИЕ

- Некоторые из перечисленных параметров становятся доступны в UI ADCM после установки флага ***Advanced***.
- Значения, установленные в ***Custom***-параметрах, перепишут текущие значения параметров, даже если они read-only.

Airflow

Redis configuration

Параметр	Описание	Значение по умолчанию
redis.conf	Файл конфигурации Redis	—
sentinel.conf	Файл конфигурации Sentinel	—
redis_port	Порт брокера Redis	6379
sentinel_port	Порт Sentinel	26379



airflow.cfg

Параметр	Описание	Значение по умолчанию
db_user	Имя пользователя для подключения к базе данных Metastore	airflow
db_password	Пароль для подключения к базе данных Metastore	—
db_port	Порт для подключения к базе данных Metastore	3307
admin_password	Пароль для администратора веб-сервера	—
server_port	Порт для работы веб-сервера	8080
flower_port	Порт для работы Celery Flower	5555
worker_port	При запуске Airflow Worker Airflow запускает подпроцесс веб-сервера для передачи локальных Worker лог-файлов на основной веб-сервер Airflow, который затем генерирует страницы и отправляет их пользователям. Этот параметр определяет порт для передачи лог-файлов. Порт должен быть не занят и доступен с основного веб-сервера для подключения к Workers	8793
fernet_key	Секретный ключ для сохранения паролей подключения в базе данных	—
security	Определяет механизм безопасности для использования. Например, `kerberos`	—
keytab	Путь к keytab-файлу	—
reinit_frequency	Устанавливает частоту обновления тикетов	3600
principal	Принципал Kerberos	
ssl_active	Определяет, активен ли SSL для Airflow	false
web_server_ssl_cert	Путь к SSL-сертификату	/etc/ssl/certs/host_cert.cert
web_server_ssl_key	Путь к ключу SSL-сертификата	/etc/ssl/host_cert.key
Logging level	Указывает уровень логирования активности Airflow	INFO
Logging level for Flask-appbuilder UI	Указывает уровень логирования для веб-интерфейса Flask-appbuilder	WARNING
cfg_properties_template	Jinja-шаблон для инициализации переменных окружения Airflow	cfg_properties_template

External database

Параметр	Описание	Значение по умолчанию
Database type	Тип внешней базы данных. Возможные значения: `PostgreSQL`, `MySQL/MariaDB`	MySQL/MariaDB
Hostname	Хост внешней базы данных	—
db_custom_port	Порт внешней базы данных	—
Airflow database name	Имя внешней базы данных	airflow

External Broker

Параметр	Описание	Значение по умолчанию
Broker URL	URL внешнего брокера	—

LDAP Security manager

Параметр	Описание	Значение по умолчанию
AUTH_LDAP_SERVER	URI сервера LDAP	—
AUTH_LDAP_BIND_USER	Путь к прокси-пользователю LDAP для привязки к верхнему уровню. Например: `cn=airflow`, `ou=users`, `dc=example`, `dc=com`	—
AUTH_LDAP_BIND_PASSWORD	Пароль привязанного пользователя	—
AUTH_LDAP_SEARCH	Укажите в значении параметра новый путь, под которым пользователи должны получать доступ к Airflow. Например: `dc=example`, `dc=com`	—
AUTH_LDAP_UID_FIELD	Поле UID (уникальный идентификатор) в LDAP	—
AUTH_ROLES_MAPPING	Параметр для сопоставления внутренних ролей группам LDAP Active Directory	—
AUTH_LDAP_GROUP_FIELD	Атрибут пользователя LDAP, имеющий свои DN	—
AUTH_ROLES_SYNC_AT_LOGIN	Флаг, указывающий, следует ли заменять все роли пользователя при каждом входе в систему или только при регистрации	true
PERMANENT_SESSION_LIFETIME	Устанавливает тайм-аут бездействия, по истечении которого пользователи должны повторно пройти аутентификацию (для синхронизации ролей)	1800
AUTH_LDAP_USE_TLS	Использование TLS	false
AUTH_LDAP_ALLOW_SELF_SIGNED	Использование самоподписанных сертификатов	true
AUTH_LDAP_TLS_CACERTFILE	Местонахождение сертификата	—

Core configuration

core-site.xml

Параметр	Описание	Значение по умолчанию
fs.defaultFS	Название файловой системы по умолчанию в виде URI, на основании которого будет выбрана имплементация файловой системы	hdfs://hdfs

fs.trash.checkpoint.interval	Интервал между trash-чекпойнтами в минутах. Должно быть меньше или равно `fs.trash.interval`. Если значение равно `0`, то используется значение параметра `fs.trash.interval`. При каждом запуске чекпойнтера создается новый чекпойнт и удаляются все чекпойнты, созданные больше `fs.trash.interval` минут назад	60
fs.trash.interval	Длительность хранения чекпойнта в минутах. Если значение равно `0`, то функция удаления старых чекпойнтов отключена. Эта опция может быть настроена как на сервере, так и на клиенте. Если на сервере функция отключена, то проверяется значение на клиенте. Если функция включена на сервере, то используется серверное значение, а клиентское игнорируется	1440
hadoop.tmp.dir	Корневая директория для создания временных директорий	/tmp/hadoop-${user.name}
hadoop.zk.address	Хост и порт сервера ZooKeeper	—
io.file.buffer.size	Размер буфера для файлов последовательностей. Размер этого буфера должен быть кратен размеру страницы в соответстувующей системе (например, 4096 для Intel x86). Значение определяет объем данных, находящихся в буфере во время операций чтения и записи	131072
net.topology.script.file.name	Название скрипта, который следует запустить для разрешения имен DNS и имен `NetworkTopology`	—
hadoop.proxyuser.hbase.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в HBase	*
hadoop.proxyuser.hbase.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в HBase	*
hadoop.proxyuser.hue.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в HUE	*
hadoop.proxyuser.hue.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в HUE	*
hadoop.proxyuser.hbase-phoenix_queryserver.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в Phoenix Query Server	*
hadoop.proxyuser.hbase-phoenix_queryserver.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в Phoenix Query Server	*
hadoop.proxyuser.hive.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в Hive	*
hadoop.proxyuser.hive.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в Hive	*
hadoop.proxyuser.httpfs.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в HttpFS	*
hadoop.proxyuser.httpfs.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в HttpFS	*
hadoop.proxyuser.HTTP.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в HTTP keytab services	*
hadoop.proxyuser.HTTP.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в HTTP keytab services	*

hadoop.proxyuser.knox.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в Knox	*
hadoop.proxyuser.knox.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в Knox	*
hadoop.proxyuser.kyuubi.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в Kyuubi	*
hadoop.proxyuser.kyuubi.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в Kyuubi	*
hadoop.proxyuser.livy.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в Livy	*
hadoop.proxyuser.livy.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в Livy	*
hadoop.proxyuser.yarn.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в YARN	*
hadoop.proxyuser.yarn.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в YARN	*
hadoop.proxyuser.zeppelin.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в Zeppelin	*
hadoop.proxyuser.zeppelin.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в Zeppelin	*
hadoop.proxyuser.trino.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в Trino	*
hadoop.proxyuser.trino.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в Trino	*
fs.s3a.endpoint	URL эндпойнта AWS S3	—
fs.s3a.access.key	Ключ доступа AWS S3	—
fs.s3a.secret.key	Секретный ключ AWS S3	—
fs.s3a.impl	URI класса файловой системы AWS S3	org.apache.hadoop.fs.s3a.S3AFileSystem
fs.s3a.fast.upload	Определяет, используется ли функция Fast Upload	true
fs.s3a.connection.ssl.enabled	Определяет, используется ли SSL для подключения к сервисам AWS	false
fs.s3a.path.style.access	Определяет, используется ли path-style доступ к S3	true
hadoop.proxyuser.om.groups	Список групп, перечисленных через запятую, для пользователей которых разрешен механизм имперсонации в Ozone Manager	*
hadoop.proxyuser.om.hosts	Список хостов, перечисленных через запятую. Для пользователей, выполняющих подключение с этих хостов, будет разрешен механизм имперсонации в Ozone Manager	*
ha.zookeeper.quorum	Разделенный запятыми список серверов ZooKeeper, которые будут использованы `ZKFailoverController` для отказоустойчивости	—

ipc.client.fallback-to-simple-auth-allowed	Определяет, сможет ли клиент принять инструкцию сервера по переходу на аутентификацию SASL SIMPLE (незащищенную). При значении `false` клиент не разрешит переход на SIMPLE-аутентификацию и оборвет соединение	true
hadoop.security.authentication	Тип аутентификации. Возможные значения: - `simple` — аутентификация отключена; - `kerberos` — Kerberos-аутентификация.	simple
hadoop.security.authorization	Определяет, разрешена ли RPC-авторизация на уровне сервиса	false
hadoop.rpc.protection	Список активных функций защиты. Возможные значения: - `authentication` — только аутентификация; - `integrity` — проверка целостности в дополнение к аутентификации; - `privacy` — шифрование данных в дополнение к проверке целостности.	authentication
hadoop.security.auth_to_local	Правила сопоставления имен принципалов с именами локальных пользователей	—
User managed hadoop.security.auth_to_local	Определяет, включена ли автоматическая генерация значения параметра `hadoop.security.auth_to_local`	false
hadoop.http.authentication.type	Определяет тип аутентификации для HTTP веб-консолей. Возможные значения: - `simple` — simple-аутентификация; - `kerberos` — Kerberos-аутентификация; - `[AUTHENTICATION_HANDLER_CLASSNAME]` — пользовательская имплементация аутентификации.	simple
hadoop.http.authentication.kerberos.principal	Имя Kerberos-принципала для `kerberos`-аутентификации. Короткое имя принципала должно быть HTTP исходя из спецификации Kerberos HTTP SPNEGO. `_HOST` (если имеется) заменится на адрес связанного HTTP-сервера	HTTP/_HOST@$LOCALHOST
hadoop.http.authentication.kerberos.keytab	Местоположение keytab-файла с данными Kerberos-принципала	/etc/security/keytabs/HTTP.service.keytab
ha.zookeeper.acl	ACL для всех znode	—
hadoop.http.filter.initializers	Значение параметра должно содержать initializer-класс `org.apache.hadoop.security.AuthenticationFilterInitializer`	org.apache.hadoop.security.AuthenticationFilterInitializer
hadoop.http.authentication.signature.secret.file	Секретный файл подписи для подписи токенов аутентификации. Если параметр не задан, во время запуска генерируется случайный секретный ключ. Один и тот же секретный ключ следует использовать для всех узлов кластера, JobTracker, NameNode, DataNode и TastTracker. Этот файл должен быть доступен для чтения только пользователю Unix, запускающему демоны	/etc/security/http_secret
hadoop.http.authentication.cookie.domain	Домен, используемый для HTTP cookie, в котором хранится токен аутентификации. Для корректной аутентификации на всех узлах кластера домен должен быть установлен правильно. Значение по умолчанию отсутствует, и если параметр не задан, HTTP cookie не будет иметь домена и будет работать только с именем хоста, который выпустил cookie	—
hadoop.ssl.require.client.cert	Определяет, необходимы ли пользовательские сертификаты	false
hadoop.ssl.hostname.verifier	Верификатор имени хоста для `HttpsURLConnections`. Возможные значения:	DEFAULT

	- `DEFAULT` - `STIRCT` - `STRICT_IE6` - `DEFAULT_AND_LOCALHOST` - `ALLOW_ALL`	
hadoop.ssl.keystores.factory.class	Имя класса, имплементирующего `KeyStoresFactory`	org.apache.hadoop.security.ssl.FileBasedKeyStoresFactory
hadoop.ssl.server.conf	Имя файла, из которого будет извлечена информация об SSL-сервере. Поиск файла происходит в CLASSPATH. Обычно он находится в директории *conf/* сервиса Hadoop	ssl-server.xml
hadoop.ssl.client.conf	Имя файла, из которого будет извлечена информация об SSL-клиенте. Поиск файла происходит в CLASSPATH. Обычно он находится в директории *conf/* сервиса Hadoop	ssl-client.xml
hadoop.ssl.enabled.protocols	Поддерживаемые протоколы SSL	TLSv1.2
fs.AbstractFileSystem.ofs.impl	URI класса AbstractFileSystem для файловой системы Rooted Ozone (ofs)	org.apache.hadoop.fs.ozone.RootedOzFs
fs.ofs.impl	Класс имплементации файловой системы ofs	org.apache.hadoop.fs.ozone.RootedOzoneFileSystem

ssl-server.xml

Параметр	**Описание**	**Значение по умолчанию**
ssl.server.truststore.location	Расположение truststore-файла, используемого NameNode и DataNode	—
ssl.server.truststore.password	Пароль к truststore-файлу	—
ssl.server.truststore.type	Формат truststore-файла	jks
ssl.server.truststore.reload.interval	Интервал между проверками обновления truststore-файла в миллисекундах	10000
ssl.server.keystore.location	Расположение keystore-файла, используемого NameNode и DataNode	—
ssl.server.keystore.password	Пароль к keystore-файлу	—
ssl.server.keystore.keypassword	Пароль к ключу в keystore-файле	—
ssl.server.keystore.type	Формат keystore-файла	jks

ssl-client.xml

Параметр	**Описание**	**Значение по умолчанию**
ssl.client.truststore.location	Расположение truststore-файла, используемого NameNode и DataNode	—
ssl.client.truststore.password	Пароль к truststore-файлу	—

ssl.client.truststore.type	Формат truststore-файла	jks
ssl.client.truststore.reload.interval	Интервал между проверками обновления truststore-файла в миллисекундах	10000
ssl.client.keystore.location	Расположение keystore-файла, используемого NameNode и DataNode	—
ssl.client.keystore.password	Пароль к keystore-файлу	—
ssl.client.keystore.keypassword	Пароль к ключу в keystore-файле	—
ssl.client.keystore.type	Формат keystore-файла	jks

Прочее

Параметр	Описание	Значение по умолчанию
Custom core-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *core-site.xml*	—
Custom hadoop-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hadoop-env.sh*	—
Custom ssl-server.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ssl-server.xml*	—
Custom ssl-client.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ssl-client.xml*	—
Custom log4j.properties	Пользовательские настройки логирования	log.conf

Компонент Configuration server

nginx.conf

Параметр	Описание	Значение по умолчанию
root_config_dir	Корневое расположение для хранения конфигураций	/srv/config
nginx_http_port	Порт HTTP для Nginx	9998
nginx_https_port	Порт HTTPS для Nginx	9998
ssl_certificate	Путь к сертификату SSL для Nginx	/etc/ssl/certs/host_cert.cert
ssl_certificate_key	Путь к ключу сертификата SSL для Nginx	/etc/ssl/host_cert.key
ssl_protocols	Версия протокола SSL, необходимая для использования SSL-транспорта	TLSv1.2

Flink

flink-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые будут записаны в *flink-env.sh*	/usr/lib/bigtop-utils/bigtop-detect-javahome
HADOOP_CLASSPATH	Список файлов и директорий, которые должны быть добавлены в classpath	$(hadoop classpath)
HADOOP_HOME	Домашняя директория Hadoop	/usr/lib/hadoop
HADOOP_CONF_DIR	Директория, в которой хранятся конфигурации Hadoop	/etc/hadoop/conf
FLINK_LOG_DIR	Директория, в которой хранятся логи Flink	/var/log/flink
FLINK_CONF_DIR	Директория, в которой хранятся конфигурации Flink	/etc/flink/conf
FLINK_HOME	Домашняя директория Flink	/usr/lib/flink
$HADOOP_OPTS	Список параметров, с которыми стартует процесс Flink	$HADOOP_OPTS -Dsecurity.kerberos.token.provider.hadoopfs.renewer=yarn

flink-conf.yaml

Параметр	Описание	Значение по умолчанию
historyserver.archive.fs.dir	Разделенный запятыми список директорий с архивами задач Flink, которые сканирует Flink History Server	hdfs:///apps/flink/completed-jobs
jobmanager.archive.fs.dir	Директория, в которую JobManager сохраняет архивы завершенных задач	hdfs:///apps/flink/completed-jobs
archive.fs.refresh-interval	Интервал сканирования директорий с архивами задач в миллисекундах	10000
historyserver.web.port	Номер порта веб-интерфейса Flink History Server	8082
historyserver.web.ssl.enabled	Включает/отключает SSL для доступа к веб-интерфейсу Flink History Server	false
rest.port	Порт для подключения клиента. Используется только если режим высокой доступности находится в состоянии `NONE`	8081
jobmanager.rpc.port	RPC-порт, по которому доступен JobManager. Этот параметр не используется в режиме высокой доступности (High Availability, HA), поскольку при HA порт для подключения к JobManager генерирует ZooKeeper	6123
sql-gateway.endpoint.rest.port	Номер порта для соединения с сервисом SQL Gateway	8083
taskmanager.network.bind-policy	Политика автоматического биндинга адресов, используемая TaskManager	name
parallelism.default	Уровень параллелизма по умолчанию для всех сред выполнения	1
taskmanager.numberOfTaskSlots	Количество слотов задач, которые предлагает каждый TaskManager. Каждый слот запускает один параллельный пайплайн	1
taskmanager.cpu.cores	Количество ядер CPU, используемое TaskManager. По умолчанию значение равно количеству слотов на один TaskManager	1

taskmanager.memory.flink.size	Общий объем памяти, выделяемый для TaskExecutors	—
taskmanager.memory.process.size	Общий объем памяти процесса, выделяемый для TaskExecutors. Сюда входит вся память, которую потребляет TaskExecutor, включая общую память Flink, JVM Metaspace и JVM Overhead. В контейнеризованных средах этот параметр следует устанавливать равным памяти контейнера	2048m
jobmanager.memory.flink.size	Общий объем памяти, выделяемый для JobManager	—
jobmanager.memory.process.size	Общий объем памяти процесса, выделяемый для JobManager. Сюда входит вся память, которую потребляет JVM JobManager, включая общую память Flink, JVM Metaspace и JVM Overhead. В контейнеризованных средах этот параметр следует устанавливать равным памяти контейнера	2048m
taskmanager.heap.size	Размер Java heap для JVM TaskManager	1024m
jobmanager.memory.heap.size	Размер Java heap для JVM JobManager	—
flink.yarn.appmaster.vcores	Количество виртуальных ядер (vcores), используемых YARN application master	1
taskmanager.host	Внешний адрес сетевого интерфейса, где запущен TaskManager	—
taskmanager.memory.task.heap.size	Размер Java heap, зарезервированной для задач	256m
taskmanager.memory.task.off-heap.size	Размер памяти off-heap, зарезервированной для задач	256m
taskmanager.memory.managed.size	Размер managed-памяти для TaskExecutors. Включает память off-heap, управляемую менеджером памяти, память, зарезервированную для сортировки, хеш-таблиц, кеширования промежуточных результатов, а также бэкенд RocksDB	256m
taskmanager.memory.framework.heap.size	Размер памяти Java heap, зарезервированной для фреймворка TaskExecutor, которая не будет выделяться под слоты задач	256m
taskmanager.memory.framework.off-heap.size	Размер памяти off-heap, зарезервированной для фреймворка TaskExecutor, которая не будет выделяться под слоты задач	256m
taskmanager.memory.network.min	Минимальный размер сетевой памяти (network memory) для TaskExecutors. Сетевая память — это область памяти off-heap, зарезервированная для ShuffleEnvironment (например, сетевые буферы)	256m
taskmanager.memory.network.max	Максимальный размер сетевой памяти (network memory) для TaskExecutors. Сетевая память — это область памяти off-heap, зарезервированная для ShuffleEnvironment (например, сетевые буферы)	256m
taskmanager.memory.jvm-overhead.max	Максимальный размер JVM overhead для TaskExecutors. Это память типа off-heap, зарезервированная для JVM overhead, например пространство стека потоков, кеш компиляции и так далее	256m
taskmanager.memory.jvm-metaspace.size	Размер метапространства JVM для TaskExecutors	256m
yarn.provided.lib.dirs	Список директорий с provided-библиотеками, разделенный точкой с запятой. Flink использует эти библиотеки, чтобы не загружать локальные JAR-файлы для ускорения запуска задачи	hdfs:///apps/flink/
flink.yarn.resourcemanager.scheduler.address	Адрес интерфейса планировщика	—

flink.yarn.containers.vcores	Устанавливает количество виртуальных ядер для контейнеров Flink YARN	1
flink.yarn.application.classpath	Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	▪ /etc/hadoop/conf/* ▪ /usr/lib/hadoop/* ▪ /usr/lib/hadoop/lib/* ▪ /usr/lib/hadoop-hdfs/* ▪ /usr/lib/hadoop-hdfs/lib/* ▪ /usr/lib/hadoop-yarn/* ▪ /usr/lib/hadoop-yarn/lib/* ▪ /usr/lib/hadoop-mapreduce/* ▪ /usr/lib/hadoop-mapreduce/lib/*
high-availability.cluster-id	ID Flink-кластера для разделения нескольких Flink-кластеров друг от друга	default
high-availability.storageDir	Путь в файловой системе (URI), где Flink хранит метаданные в режиме высокой доступности	—
high-availability	Определяет режим высокой доступности (High Availability, HA) для кластера	ZOOKEEPER
high-availability.zookeeper.quorum	ZooKeeper-кворум при работе Flink в режиме высокой доступности с ZooKeeper	—
high-availability.zookeeper.path.root	Рутовый путь Flink ZNode в Zookeeper	/flink
sql-gateway.session.check-interval	Интервал для проверки сессий на бездействие. Значение меньшее или равное `0` отключает проверки	1 min
sql-gateway.session.idle-timeout	Время, по истечении которого сессия автоматически закрывается, если не было установлено ни единого соединения. При значении меньшем или равном `0` сессии не закрываются	10 min
sql-gateway.session.max-num	Максимально возможное количество сессий на сервере SQL Gateway	1000000
sql-gateway.worker.keepalive-time	Время гарантированной жизни worker-потоков при бездействии. Когда количество worker-потоков превышает значение `sql-gateway.worker.threads.min`, потоки удаляются после данного интервала	5 min
sql-gateway.worker.threads.max	Максимально возможное количество worker-потоков	500
sql-gateway.worker.threads.min	Минимальное количество worker-потоков на сервере SQL Gateway. Если количество потоков меньше данного значения, потоки автоматически не удаляются	5
env.java.opts.all	Параметры Java, используемые для запуска всех JVM Flink	-Djava.io.tmpdir={{ cluster.config.java_tmpdir | d('/tmp') }}
security.kerberos.login.use-ticket-cache	Определяет, использовать ли кеш тикетов Kerberos	false
security.delegation.token.provider.HiveServer2.enabled	Определяет, должен ли сервис получать учетные данные из HiveServer2 при включенной безопасности	false

security.kerberos.login.keytab	Абсолютный путь к Kerberos keytab-файлу, который хранит учетные данные пользователя	—
security.kerberos.login.principal	Принципал Kerberos для Flink	—
security.delegation.tokens.hive.renewer	Принципал Flink Kerberos для Hive	—
security.kerberos.login.contexts	Разделенный запятыми список логин-контекстов, которым предоставляются учетные данные Kerberos	—
security.ssl.internal.enabled	Включает SSL для внутренних каналов связи между компонентами Flink. Сюда входит общение между TaskManagers, транспортировка данных из JobManager в TaskManager, установка RPC-соединений и так далее	false
security.ssl.internal.keystore	Путь к keystore-файлу, который используется внутренними эндпойнтами Flink	—
security.ssl.internal.truststore	Путь к truststore-файлу, который используется внутренними эндпойнтами Flink	—
security.ssl.internal.keystore-password	Пароль к keystore-файлу, который используется внутренними эндпойнтами Flink	—
security.ssl.internal.truststore-password	Пароль к truststore-файлу, который используется внутренними эндпойнтами Flink	—
security.ssl.internal.key-password	Пароль для дешифрования ключа в keystore-файле, который используется внутренними эндпойнтами Flink	—
security.ssl.rest.enabled	Включает SSL для коммуникации через внешние REST-эндпойнты	false
security.ssl.rest.keystore	Java keystore-файл, который содержит SSL-ключ и сертификат для использования внешними REST-эндпойнтами Flink	—
security.ssl.rest.truststore	Truststore-файл, который содержит сертификаты ЦС для проверки внешних REST-эндпойнтов Flink	—
security.ssl.rest.keystore-password	Секретный ключ для дешифрования keystore-файла, используемого для внешних REST-эндпойнтов Flink	—
security.ssl.rest.truststore-password	Пароль для доступа к truststore-файлу, который используется для внешних REST-эндпойнтов Flink	—
security.ssl.rest.key-password	Пароль для расшифровки ключа в keystore-файле для использования внешними REST-эндпойнтами Flink	—
security.ssl.protocol	Версия протокола TLS, которая будет использоваться для SSL. Принимает только одно значение, но не список	TLSv1.2
zookeeper.sasl.disable	Определяет использование SASL-аутентификации в Zookeeper	false
Logging level	Устанавливает уровень логирования Flink	INFO
yarn.classpath.include-user-jar	Указывает, добавлять ли пользовательские JAR-файлы в classpath, а также указывает позицию JAR в classpath-строке. Возможные значения: ▪ `DISABLED` — пользовательские JAR-файлы не добавляются в classpath. ▪ `FIRST` — пользовательские JAR-файлы добавляются в начало classpath перед системными библиотеками. Это позволяет переопределить системные классы/библиотеки. ▪ `LAST` — пользовательские JAR-файлы добавляются в конец classpath-строки. ▪ `ORDER` — пользовательские JAR-файлы добавляются в classpath в алфавитном порядке.	ORDER

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom flink-conf.yaml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *flink-conf.yaml*	—
log4j.properties	Содержимое конфигурационного файла *log4j.properties*	log4j.properties
log4j-cli.properties	Содержимое конфигурационного файла *log4j-cli.properties*	log4j-cli.properties

Flink2

flink-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые будут записаны в *flink-env.sh*	/usr/lib/bigtop-utils/bigtop-detect-javahome
HADOOP_CLASSPATH	Список файлов и директорий, которые должны быть добавлены в classpath	$(hadoop classpath)
HADOOP_HOME	Домашняя директория Hadoop	/usr/lib/hadoop
HADOOP_CONF_DIR	Директория, в которой хранятся конфигурации Hadoop	/etc/hadoop/conf
FLINK_LOG_DIR	Директория, в которой хранятся логи Flink	/var/log/flink2
FLINK_CONF_DIR	Директория, в которой хранятся конфигурации Flink	/etc/flink2/conf
FLINK_HOME	Домашняя директория Flink	/usr/lib/flink2
$HADOOP_OPTS	Список дополнительных параметров для запуска процесса Flink	$HADOOP_OPTS -Dsecurity.kerberos.token.provider.hadoopfs.renewer=yarn

config.yaml

Параметр	Описание	Значение по умолчанию
historyserver.archive.fs.dir	Разделенный запятыми список директорий с архивами задач Flink, которые сканирует Flink History Server	hdfs:///apps/flink2/completed-jobs
jobmanager.archive.fs.dir	Директория, в которую JobManager сохраняет архивы завершенных задач	hdfs:///apps/flink2/completed-jobs
archive.fs.refresh-interval	Интервал сканирования директорий с архивами задач в миллисекундах	10000
historyserver.web.port	Номер порта веб-интерфейса Flink History Server	8092
historyserver.web.ssl.enabled	Включает/отключает SSL для доступа к веб-интерфейсу Flink History Server	false

jobmanager.rpc.port	RPC-порт, по которому доступен JobManager. Этот параметр не используется в режиме высокой доступности (High Availability, HA), поскольку при HA порт для подключения к JobManager генерирует ZooKeeper	6123
sql-gateway.endpoint.rest.port	Номер порта для соединения с сервисом SQL Gateway	8093
taskmanager.network.bind-policy	Политика автоматического биндинга адресов, используемая TaskManager	name
parallelism.default	Уровень параллелизма по умолчанию для всех сред выполнения	1
taskmanager.numberOfTaskSlots	Количество слотов задач, которые предлагает каждый TaskManager. Каждый слот запускает один параллельный пайплайн	1
taskmanager.cpu.cores	Количество ядер CPU, используемое TaskManager. По умолчанию значение равно количеству слотов на один TaskManager	1
taskmanager.memory.process.size	Общий объем памяти процесса, выделяемый для TaskExecutors. Сюда входит вся память, которую потребляет TaskExecutor, включая общую память Flink, JVM Metaspace и JVM Overhead. В контейнеризованных средах этот параметр следует устанавливать равным памяти контейнера	2048m
jobmanager.memory.process.size	Общий объем памяти процесса, выделяемый для JobManager. Сюда входит вся память, которую потребляет JVM JobManager, включая общую память Flink, JVM Metaspace и JVM Overhead. В контейнеризованных средах этот параметр следует устанавливать равным памяти контейнера	2048m
flink.yarn.appmaster.vcores	Количество виртуальных ядер (vcores), используемых YARN application master	1
taskmanager.host	Внешний адрес сетевого интерфейса, где запущен TaskManager	—
taskmanager.memory.task.heap.size	Размер Java heap, зарезервированной для задач	256m
taskmanager.memory.task.off-heap.size	Размер памяти off-heap, зарезервированной для задач	256m
taskmanager.memory.managed.size	Размер managed-памяти для TaskExecutors. Включает память off-heap, управляемую менеджером памяти, память, зарезервированную для сортировки, хеш-таблиц, кеширования промежуточных результатов, а также бэкенд RocksDB	256m
taskmanager.memory.framework.heap.size	Размер памяти Java heap, зарезервированной для фреймворка TaskExecutor, которая не будет выделяться под слоты задач	256m
taskmanager.memory.framework.off-heap.size	Размер памяти off-heap, зарезервированной для фреймворка TaskExecutor, которая не будет выделяться под слоты задач	256m
taskmanager.memory.network.min	Минимальный размер сетевой памяти (network memory) для TaskExecutors. Сетевая память — это область памяти off-heap, зарезервированная для ShuffleEnvironment (например, сетевые буферы)	256m
taskmanager.memory.network.max	Максимальный размер сетевой памяти (network memory) для TaskExecutors. Сетевая память — это область памяти off-heap, зарезервированная для ShuffleEnvironment (например, сетевые буферы)	256m
taskmanager.memory.jvm-overhead.max	Максимальный размер JVM overhead для TaskExecutors. Это память типа off-heap, зарезервированная для JVM overhead, например пространство стека потоков, кеш компиляции и так далее	256m
taskmanager.memory.jvm-metaspace.size	Размер метапространства JVM для TaskExecutors	256m

yarn.provided.lib.dirs	Список директорий с provided-библиотеками, разделенный точкой с запятой. Flink использует эти библиотеки, чтобы не загружать локальные JAR-файлы для ускорения запуска задачи	hdfs:///apps/flink2/
flink.yarn.resourcemanager.scheduler.address	Адрес интерфейса планировщика	—
flink.yarn.containers.vcores	Устанавливает количество виртуальных ядер для контейнеров Flink YARN	1
flink.yarn.application.classpath	Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	■ /etc/hadoop/conf/* ■ /usr/lib/hadoop/* ■ /usr/lib/hadoop/lib/* ■ /usr/lib/hadoop-hdfs/* ■ /usr/lib/hadoop-hdfs/lib/* ■ /usr/lib/hadoop-yarn/* ■ /usr/lib/hadoop-yarn/lib/* ■ /usr/lib/hadoop-mapreduce/* ■ /usr/lib/hadoop-mapreduce/lib/*
sql-gateway.session.check-interval	Интервал для проверки сессий на бездействие. Значение меньшее или равное `0` отключает проверки	1 min
sql-gateway.session.idle-timeout	Время, по истечении которого сессия автоматически закрывается, если не было установлено ни единого соединения. При значении меньшем или равном `0` сессии не закрываются	10 min
sql-gateway.session.max-num	Максимально возможное количество сессий на сервере SQL Gateway	1000000
sql-gateway.worker.keepalive-time	Время гарантированной жизни worker-потоков при бездействии. Когда количество worker-потоков превышает значение `sql-gateway.worker.threads.min`, потоки удаляются после данного интервала	5 min
sql-gateway.worker.threads.max	Максимально возможное количество worker-потоков	500
sql-gateway.worker.threads.min	Минимальное количество worker-потоков на сервере SQL Gateway. Если количество потоков меньше данного значения, потоки автоматически не удаляются	5
env.java.opts.all	Параметры Java, используемые для запуска всех JVM Flink	-Djava.io.tmpdir={{ cluster.config.java_tmpdir | d('/tmp') }}
security.kerberos.login.use-ticket-cache	Определяет, использовать ли кеш тикетов Kerberos	false
security.delegation.token.provider.HiveServer2.enabled	Определяет, должен ли сервис получать учетные данные из HiveServer2 при включенной безопасности	false
security.kerberos.login.keytab	Абсолютный путь к Kerberos keytab-файлу, который хранит учетные данные пользователя	—
security.kerberos.login.principal	Принципал Kerberos для Flink	—
security.delegation.tokens.hive.renewer	Принципал Flink Kerberos для Hive	—
security.kerberos.login.contexts	Разделенный запятыми список логин-контекстов, которым предоставляются учетные данные Kerberos	—

security.ssl.internal.enabled	Включает SSL для внутренних каналов связи между компонентами Flink. Сюда входит общение между TaskManagers, транспортировка данных из JobManager в TaskManager, установка RPC-соединений и так далее	false
security.ssl.internal.keystore	Путь к keystore-файлу, который используется внутренними эндпойнтами Flink	—
security.ssl.internal.truststore	Путь к truststore-файлу, который используется внутренними эндпойнтами Flink	—
security.ssl.internal.keystore-password	Пароль к keystore-файлу, который используется внутренними эндпойнтами Flink	—
security.ssl.internal.truststore-password	Пароль к truststore-файлу, который используется внутренними эндпойнтами Flink	—
security.ssl.internal.key-password	Пароль для дешифрования ключа в keystore-файле, который используется внутренними эндпойнтами Flink	—
security.ssl.rest.enabled	Включает SSL для коммуникации через внешние REST-эндпойнты	false
security.ssl.rest.keystore	Java keystore-файл, который содержит SSL-ключ и сертификат для использования внешними REST-эндпойнтами Flink	—
security.ssl.rest.truststore	Truststore-файл, который содержит сертификаты ЦС для проверки внешних REST-эндпойнтов Flink	—
security.ssl.rest.keystore-password	Секретный ключ для дешифрования keystore-файла, используемого для внешних REST-эндпойнтов Flink	—
security.ssl.rest.truststore-password	Пароль для доступа к truststore-файлу, который используется для внешних REST-эндпойнтов Flink	—
security.ssl.rest.key-password	Пароль для расшифровки ключа в keystore-файле для использования внешними REST-эндпойнтами Flink	—
security.ssl.protocol	Версия протокола TLS, которая будет использоваться для SSL. Принимает только одно значение, но не список	TLSv1.2
Logging level	Устанавливает уровень логирования Flink	INFO
yarn.classpath.include-user-jar	Указывает, добавлять ли пользовательские JAR-файлы в classpath, а также указывает позицию JAR в classpath-строке. Возможные значения: ▪ `DISABLED` — пользовательские JAR-файлы не добавляются в classpath. ▪ `FIRST` — пользовательские JAR-файлы добавляются в начало classpath перед системными библиотеками. Это позволяет переопределить системные классы/библиотеки. ▪ `LAST` — пользовательские JAR-файлы добавляются в конец classpath-строки. ▪ `ORDER` — пользовательские JAR-файлы добавляются в classpath в алфавитном порядке.	ORDER

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom flink_env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *flink_env.sh*	—

Custom config.yaml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *config.yaml*	—
Custom sql-gateway-runner flink-conf.yaml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле sql-gateway-runner *flink-conf.yaml*	—
log4j.properties	Содержимое конфигурационного файла *log4j.properties*	log4j.properties
log4j-cli.properties	Содержимое конфигурационного файла *log4j-cli.properties*	log4j-cli.properties
log4j-console.properties	Содержимое конфигурационного файла *log4j-console.properties*	log4j-console.properties
log4j-session.properties	Содержимое конфигурационного файла *log4j-session.properties*	log4j-session.properties

HBase

hbase-site.xml

Параметр	Описание	Значение по умолчанию
hbase.balancer.period	Интервал для запуска балансировщика регионов в мастере	300000
hbase.client.pause	Общее значение паузы клиента. Используется для указания паузы перед повторной попыткой после неудачного получения, поиска региона и так далее. Для получения более подробной информации об использовании пауз с повторными попытками смотрите описание параметра `hbase.client.retries.number`	100
hbase.client.max.perregion.tasks	Максимальное количество одновременных задач мутации, которые клиент может поддерживать в одном регионе. Иными словами, если для определенного региона уже выполняется `hbase.client.max.perregion.tasks` процессов записи, новые операции put не будут отправляться в этот регион, пока процесс записи не завершится	1
hbase.client.max.perserver.tasks	Максимальное количество одновременных задач мутации, которые один экземпляр HTable будет отправлять на один Region-сервер	2
hbase.client.max.total.tasks	Максимальное количество одновременных задач мутации, которые один экземпляр HTable будет отправлять в кластер	100
hbase.client.retries.number	Максимальное количество повторных попыток. Заданное значение используется для всех операций повтора, таких как: получение значения ячейки, запуск обновления строки и так далее. Интервал повтора вычисляется функцией, основанной на `hbase.client.pause`. Чтобы узнать, как увеличивается размер резервных копий при повторной обработке, смотрите значение константы `RETRY_BACKOFF`. Измените этот параметр и `hbase.client.pause` в соответствии с вашей рабочей нагрузкой	15
hbase.client.scanner.timeout.period	Период аренды клиентского сканера (в миллисекундах)	60000
hbase.cluster.distributed	Режим кластера. Возможные значения: `false` — для автономного режима и псевдораспределенных систем с управляемым ZooKeeper; `true` — для полностью распределенного режима с неуправляемым ZooKeeper Quorum. Если указано `false`, все демоны HBase и ZooKeeper запускаются в одной JVM, если `true` — каждый демон запускается в отдельной JVM	true
hbase.hregion.majorcompaction	Интервал между выполнением Major compaction (в миллисекундах). Установите значение `0`, чтобы отключить автоматические	604800000

	выполнение Major compaction по времени. Major compaction, запрошенные пользователем и основанные на размере, будут по-прежнему выполняться. Значение параметра умножается на `hbase.hregion.majorcompaction.jitter`, чтобы операция сжатия выполнялась в случайное время в течение определенного временного промежутка	
hbase.hregion.max.filesize	Максимальный размер файла. Если общий размер некоторых HFile в регионе превышает это значение, регион разделяется на две части. Существует два варианта работы этого параметра: первый, когда размер любого хранилища превышает пороговое значение, затем разделяется, а другой — если общий размер региона превышает пороговое значение, затем разделяется. Его можно настроить с помощью `hbase.hregion.split.overallfiles`	10737418240
hbase.hstore.blockingStoreFiles	Если количество StoreFile в хранилище превышает значение данного параметра (один StoreFile записывается на каждый сброс MemStore), обновления для этого региона блокируются до тех пор, пока не будет завершено сжатие (compaction) или пока не будет превышено значение `hbase.hstore.blockingWaitTime`	16
hbase.hstore.blockingWaitTime	Время, в течение которого регион будет блокировать обновления после достижения предела StoreFile. Лимит определяется параметром `hbase.hstore.blockingStoreFiles`. По истечении этого времени регион перестанет блокировать обновления, даже если процесс сжатия не был завершен	90000
hbase.hstore.compaction.max	Максимальное количество StoreFiles, которое будет выбрано для одной процедуры сжатия (Minor compaction), независимо от количества StoreFiles, готовых к сжатию. Значение `hbase.hstore.compaction.max` контролирует время, необходимое для завершения одного сжатия. Установка большего значения означает, что в одно сжатие включается больше файлов StoreFile. В большинстве случаев подходит значение по умолчанию	10
hbase.hstore.compaction.min	Минимальное количество StoreFiles, которые должны быть пригодны для сжатия, прежде чем выполнять операцию сжатия. Назначение параметра состоит в том, чтобы избежать ситуации, когда выполняется сжатие для большого количества маленьких StoreFiles. Установка значения `2` приводит к выполнению незначительного сжатия каждый раз, когда у вас есть два StoreFile в хранилище, и это, вероятно, неподходящее поведение. Если вы установите слишком большое значение, все остальные значения необходимо будет соответствующим образом отрегулировать. В большинстве случаев подходит значение по умолчанию. В предыдущих версиях HBase параметр `hbase.hstore.compaction.min` назывался `hbase.hstore.compactionThreshold`	3
hbase.hstore.compaction.min.size	Все StoreFile, размер которых меньше, чем указано параметром, могут быть подвержены процедуре Minor compaction. StoreFiles, размер которых равен или больше значения данного параметра, оцениваются с помощью `hbase.hstore.compaction.ratio`, чтобы определить, подходят ли они для сжатия. Поскольку это ограничение автоматически включает все StoreFile, размер которых меньше значения параметра, значение параметра может потребоваться уменьшить в средах с высокой интенсивностью записи, когда выполняется запись большого количества файлов размером 1-2 МБ. В противном случае, каждый StoreFile будет номинирован на сжатие, а размер результирующего файла будет все так же мал, что повлечет очередное сжатие. Если значение параметра небольшое, проверка соотношения срабатывает быстрее. Это устранило некоторые проблемы, замеченные в более ранних версиях HBase, но в большинстве ситуаций изменение этого параметра больше не требуется	134217728
hbase.hstore.compaction.ratio	Для операции Minor compaction данный параметр используется для определения того, подходит ли для сжатия StoreFile, размер которого превышает `hbase.hstore.compaction.min.size`. Назначение параметра заключается в ограничении сжатия больших StoreFile. Значение `hbase.hstore.compaction.ratio` выражается в виде десятичного числа с плавающей запятой	1.2F

hbase.hstore.compaction.ratio.offpeak	Коэффициент сжатия, используемый при сжатиях вне пиковой нагрузки, если также настроены часы непиковой нагрузки. Значение параметра — десятичное число с плавающей запятой. Позволяет выполнять более агрессивное (или менее агрессивное, если вы установите его ниже, чем `hbase.hstore.compaction.ratio`) сжатие в течение заданного периода времени. Значение игнорируется, если непиковое время отключено (по умолчанию). Параметр работает аналогично параметру `hbase.hstore.compaction.ratio`	5.0F
hbase.hstore.compactionThreshold	Если количество StoreFiles в хранилище превышает значение параметра (один StoreFile записывается при каждом сбросе MemStore), выполняется процедура сжатия для перезаписи всех StoreFiles в один StoreFile. Чем больше значение, тем реже выполняется сжатие, однако в таком случае для завершения сжатия требуется больше времени	3
hbase.hstore.flusher.count	Количество потоков, выполняющих сбросы (flush). При меньшем количестве потоков сбросы в MemStore будут поставлены в очередь. При большем количестве потоков сбросы будут выполняться параллельно, увеличивая нагрузку на HDFS и потенциально вызывая больше операций сжатия	2
hbase.hstore.time.to.purge.deletes	Время задержки до выполнения очистки маркеров удаления, включая маркеры с отметками времени в будущем. Если значение не указано или используется `0`, все маркеры удаления, в том числе с отметками времени в будущем, удаляются во время следующего Major compaction. В противном случае маркер удаления сохраняется до этапа основного сжатия (Major compaction), которое выполняется после отметки времени маркера плюс значение этого параметра (в миллисекундах)	0
hbase.master.ipc.address	HMaster RPC	0.0.0.0
hbase.normalizer.period	Период, в течение которого нормализатор региона работает на Мастере (в миллисекундах)	300000
hbase.regionserver.compaction.enabled	Включает/отключает сжатие, используя значения `true`/`false`. Вы также можете динамически переключать операции сжатия с помощью команды оболочки `compaction_switch`	true
hbase.regionserver.ipc.address	IP-адрес Region Server	0.0.0.0
hbase.regionserver.regionSplitLimit	Максимальное количество регионов, при достижении которого деление на регионы (split) больше не выполняется. Данный параметр не является жестким ограничением количества регионов, но служит ориентиром для Region Server, чтобы прекратить разделение после определенного предела	1000
hbase.rootdir	Каталог, совместно используемый Region Servers и в который HBase записывает данные. URL-адрес должен быть полностью определенным и должен включать схему файловой системы. Например, чтобы указать каталог HDFS */hbase*, где экземпляр NameNode HDFS имеет адрес *namenode.example.org* и порт *9000*, установите это значение следующим образом: `hdfs://namenode.example.org:9000/hbase`	—
hbase.zookeeper.quorum	Список серверов в ZooKeeper ensemble, разделенных запятыми. Например, `host1.mydomain.com,host2.mydomain.com,host3.mydomain.com`. По умолчанию используется значение `localhost` для локальных и псевдораспределенных режимов работы. Для полностью распределенных систем значение параметра должно содержать полный список серверов ZooKeeper ensemble. Если `HBASE_MANAGES_ZK` установлено в *hbase-env.sh*, HBase будет использовать список серверов для запуска ZooKeeper при запуске/остановке кластера. На стороне клиента к списку ensemble добавляется значение `hbase.zookeeper.property.clientPort` и передается конструктору ZooKeeper в качестве параметра строки подключения	—

zookeeper.session.timeout	Время ожидания сеанса ZooKeeper в миллисекундах. Используется двумя способами. Первый способ — когда значение обрабатывается клиентом ZooKeeper, который HBase использует для подключения к ensemble. Параметр также используется HBase, когда последний запускает сервер ZooKeeper (в этом случае тайм-аут передается как `maxSessionTimeout`). Более деальная информация доступна на странице документации ZooKeeper. Например, если Region Server HBase подключается к ZooKeeper ensemble, который также управляется HBase, время ожидания сеанса будет задано этим параметром. Но Region Server, который подключается к ensemble, управляемому другой конфигурацией, будет использовать `maxSessionTimeout` этого ensemble. Таким образом, даже если HBase-кофигурация использует 90 секунд, максимальный тайм-аут ensemble может быть меньшее, и он будет иметь приоритет. Текущее значение по умолчанию `maxSessionTimeout` в ZooKeeper составляет `40` секунд, что меньше, чем в HBase	90000
zookeeper.znode.parent	Корневой znode для HBase в ZooKeeper. Все файлы HBase ZooKeeper, которые используют относительные пути, будут попадать в этот узел. По умолчанию все пути к файлам HBase ZooKeeper используют относительные пути, поэтому все файлы будут попадать в данный каталог, если не будут изменены	/hbase
hbase.rest.port	Порт, используемый HBase Rest Server	60080
hbase.zookeeper.property.authProvider.1	Указывает класс провайдера аутентификации для ZooKeeper	—
hbase.security.authentication	Установите значение `true`, чтобы использовать строгую аутентификацию (strong authentication) для HBase RPC	false
hbase.security.authorization	Установите значение `true`, чтобы использовать строгую авторизацию (strong authorization) для HBase RPC	false
hbase.master.kerberos.principal	Принципал Kerberos, используемый для запуска процесса HMaster	—
hbase.master.keytab.file	Абсолютный путь к Kerberos keytab-файлу, который используется для входа в систему принципала HMaster server	—
hbase.regionserver.kerberos.principal	Принципал Kerberos, который используется для запуска процесса HRegionServer	—
hbase.regionserver.keytab.file	Абсолютный путь к Kerberos keytab-файлу, который используется для входа в систему принципала HRegionServer	—
hbase.rest.authentication.type	Тип аутентификации Kerberos для REST Gateway	—
hbase.rest.authentication.kerberos.principal	Принципал Kerberos для REST Gateway	—
hbase.rest.support.proxyuser	Позволяет использовать REST-сервер с поддержкой режима прокси-пользователей	false
hbase.rest.authentication.kerberos.keytab	Kerberos keytab-файл для REST gateway	—
hbase.thrift.keytab.file	Kerberos keytab-файл для Thrift	—
hbase.rest.keytab.file	Kerberos keytab-файл для HBase REST gateway	—
hbase.rest.kerberos.principal	HBase REST gateway Kerberos principal	—
hbase.thrift.kerberos.principal	Thrift Kerberos principal	—
hbase.thrift.security.qop	Устанавливает проверку аутентификации, целостности и приватности. Поддерживаемые значения: ▪ `auth-conf` — проверка аутентификации, целостности и приватности; ▪ `auth-int` — проверка аутентификации и целостности; ▪ `auth` — только проверка аутентификации.	—

phoenix.queryserver.keytab.file	Путь к Kerberos keytab-файлу	—
phoenix.queryserver.kerberos.principal	Принципал Kerberos для использования при аутентификации. Если `phoenix.queryserver.kerberos.http.principal` не задан, указанный принципал также будет использоваться как для аутентификации соединений SPNEGO, так и для подключения к HBase	—
phoenix.queryserver.kerberos.keytab	Полный путь к Kerberos keytab-файлу, который будет использоваться для входа в систему принципала HMaster server	—
phoenix.queryserver.http.keytab.file	Keytab-файл, используемый для аутентификации соединений SPNEGO. Этот параметр должен быть указан, если задан `phoenix.queryserver.kerberos.http.principal`. `phoenix.queryserver.keytab.file` будет использоваться, если данный параметр не указан	—
phoenix.queryserver.http.kerberos.principal	Принципал Kerberos для использования при аутентификации подключений SPNEGO. `phoenix.queryserver.kerberos.principal` будет использоваться, если этот параметр не задан	—
phoenix.queryserver.kerberos.http.principal	Параметр устарел. Используйте `phoenix.queryserver.http.kerberos.principal`	—
hbase.security.authentication.ui	Включает Kerberos-аутентификацию для веб-интерфейса HBase, используя SPNEGO	—
hbase.security.authentication.spnego.kerberos.principal	Принципал Kerberos для SPNEGO-аутентификации	—
hbase.security.authentication.spnego.kerberos.keytab	Путь к Kerberos keytab-файлу с принципалами для SPNEGO-аутентификации	—
hbase.ssl.enabled	Определяет, включен ли SSL для веб-интерфейсов	false
hadoop.ssl.enabled	Определяет, включен ли SSL для Hadoop RPC	false
ssl.server.keystore.location	Путь к используемому keystore-файлу	—
ssl.server.keystore.password	Пароль для доступа к keystore-файлу	—
ssl.server.truststore.location	Путь к используемому truststore-файлу	—
ssl.server.truststore.password	Пароль для доступа к truststore-файлу	—
ssl.server.keystore.keypassword	Пароль для доступа к ключу в keystore-файле	—
hbase.rest.ssl.enabled	Определяет, включен ли SSL для HBase REST server	false
hbase.rest.ssl.keystore.store	Путь к используемому keystore-файлу	—
hbase.rest.ssl.keystore.password	Пароль для доступа к keystore-файлу	—
hbase.rest.ssl.keystore.keypassword	Пароль для доступа к ключу в keystore-файле	—
hadoop.security.credential.provider.path	Путь к провайдеру учетных данных (jceks), содержащему пароли ко всем сервисам	—

Credential encryption

Параметр	Описание	Значение по умолчанию
Encryption enable	Определяет, включено ли шифрование учетных данных	false

Credential provider path	Путь к провайдеру учетных данных для создания файлов *.jceks*, содержащих закрытые ключи	jceks://file/etc/hbase/conf/hbase.jceks
Ranger plugin credential provider path	Путь к провайдеру учетных данных плагина Ranger	jceks://file/etc/hbase/conf/ranger-hbase.jceks
Custom jceks	Определяет, используются ли кастомные файлы *.jceks*, расположенные по пути провайдера учетных данных (`true`), или автосгенерированные (`false`)	false
Password file name	Имя файла паролей в classpath сервиса, если использование файла паролей выбрано в опциях credstore	hbase_credstore_pass

hbase-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые должны быть добавлены в *hbase-env.sh*	—
HBASE_MASTER_OPTS	Дополнительные параметры для `HBASE_MASTER_OPTS`	-Xms700m -Xmx9G
PHOENIX_QUERYSERVER_OPTS	Дополнительные параметры для `PHOENIX_QUERYSERVER_OPTS`	-Xms700m -Xmx8G
HBASE_THRIFT_OPTS	Дополнительные параметры для `HBASE_THRIFT_OPTS`	-Xms700m -Xmx8G
HBASE_REST_OPTS	Дополнительные параметры для `HBASE_REST_OPTS`	-Xms200m -Xmx8G
HBASE_OPTS	Дополнительные параметры для `HBASE_OPTS`	-XX:+UseConcMarkSweepGC
HBASE_CLASSPATH	classpath для HBase. Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	/usr/lib/phoenix/phoenix-server-hbase.jar
Final HBASE_OPTS	Итоговое значение параметра `HBASE_OPTS` в файле *hbase-env.sh*	—

hbase-regionserver-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые должны быть добавлены в *hbase-regionserver-env.sh*	—
HBASE_REGIONSERVER_OPTS	Начальный (-Xms) и максимальный (-Xmx) размер Java heap для HBase Region server	-Xms700m -Xmx9G
Final HBASE_REGIONSERVER_OPTS	Итоговое значение параметра `HBASE_REGIONSERVER_OPTS` в файле *hbase-regionserver-env.sh*	—

ranger-hbase-audit.xml

Параметр	Описание	Значение по умолчанию
xasecure.audit.destination.solr.batch.filespool.dir	Путь к Spool-каталогу	/srv/ranger/hdfs_plugin/audit_solr_spool

xasecure.audit.destination.solr.urls	URL сервера Solr для сохранения аудит-событий. Оставьте значение пустым или установите значение `NONE` при использовании ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.zookeepers	Указывает строку подключения ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.force.use.inmemory.jaas.config	Использует in-memory JAAS-файл для соединения с Solr	—
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.jaas.Client.loginModuleControlFlag	Определяет требование к успешному завершению работы модуля. Значения могут быть `required`, `requisite`, `sufficient` или `optional`	—
xasecure.audit.jaas.Client.loginModuleName	Имя класса-аутентификатора	—
xasecure.audit.jaas.Client.option.keyTab	Имя keytab-файла, из которого необходимо получить секретный ключ принципала	—
xasecure.audit.jaas.Client.option.principal	Имя используемого принципала	—
xasecure.audit.jaas.Client.option.serviceName	Имя пользователя или сервиса, для которого выполнятся логин	—
xasecure.audit.jaas.Client.option.storeKey	Установите значение `true`, чтобы keytab-файл или ключ принципала сохранялись в учетных данных субъекта	false
xasecure.audit.jaas.Client.option.useKeyTab	Установите значение `true`, чтобы модуль получал ключ принципала из keytab-файла	false

ranger-hbase-security.xml

Параметр	Описание	Значение по умолчанию
ranger.plugin.hbase.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.hbase.service.name	Имя Ranger-сервиса, который содержит политики для этого экземпляра	—
ranger.plugin.hbase.policy.cache.dir	Каталог, в котором кешируются политики Ranger после успешного извлечения из источника	/srv/ranger/hbase/policycache
ranger.plugin.hbase.policy.pollIntervalMs	Определяет, как часто проверять политики на предмет изменения	30000
ranger.plugin.hbase.policy.rest.client.connection.timeoutMs	Время ожидания подключения HBase-плагина RangerRestClient (в миллисекундах)	120000
ranger.plugin.hbase.policy.rest.client.read.timeoutMs	Тайм-аут на чтение для HBase-плагина RangerRestClient (в миллисекундах)	30000
ranger.plugin.hbase.policy.rest.ssl.config.file	Путь к конфигурационному файлу RangerRestClient SSL плагина HBase	/etc/hbase/conf/ranger-hbase-policymgr-ssl.xml

ranger-hbase-policymgr-ssl.xml

Параметр	Описание	Значение по умолчанию
xasecure.policymgr.clientssl.keystore	Путь к keystore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.credential.file	Путь к файлу с паролями для keystore-файла	/etc/hbase/conf/ranger-hbase.jceks
xasecure.policymgr.clientssl.truststore.credential.file	Путь к truststore-файлу с учетными данными	/etc/hbase/conf/ranger-hbase.jceks
xasecure.policymgr.clientssl.truststore	Путь к truststore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.password	Пароль для доступа к keystore-файлу	—
xasecure.policymgr.clientssl.truststore.password	Пароль для доступа к truststore-файлу	—

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom hbase-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hbase-site.xml*	—
Custom hbase-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hbase-env.sh*	—
Custom hbase-regionserver-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hbase-regionserver-env.sh*	—
Ranger plugin enabled	Определяет, активен ли Ranger-плагин	false
Custom ranger-hbase-audit.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hbase-audit.xml*	—
Custom ranger-hbase-security.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hbase-security.xml*	—
Custom ranger-hbase-policymgr-ssl.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hbase-policymgr-ssl.xml*	—
Custom log4j.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *log4j.properties*	log4j.properties

Custom hadoop-metrics2-hbase.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hadoop-metrics2-hbase.properties*	hadoop-metrics2-hbase.properties

HDFS

Credential Encryption

Параметр	**Описание**	**Значение по умолчанию**
Encryption enable	Включает или выключает функцию шифрования учетных данных. Если опция включена, HDFS хранит учетные данные (пароли настроек, учетные данные для взаимодействия с другими сервисами) в зашифрованном виде	false
Credential provider path	Путь к хранилищу ключей JCEKS	jceks://file/etc/hadoop/conf/hadoop.jceks
Ranger plugin credential provider path	Путь к хранилищу ключей JCEKS, используемому Ranger	jceks://file/etc/hadoop/conf/ranger-hdfs.jceks
Custom jceks	Укажите значение `true` при использовании кастомного файла JCEKS-хранилища. Укажите значение `false` для использования дефолтного файла хранилища, который генерируется автоматически	false
Password file name	Имя файла в classpath сервиса, в котором хранятся пароли	hadoop_credstore_pass

Enable CORS

Параметр	**Описание**	**Значение по умолчанию**
hadoop.http.cross-origin.enabled	Обеспечивает поддержку кросс-источников для всех веб-сервисов	true
hadoop.http.cross-origin.allowed-origins	Список разрешенных источников, разделенных запятыми. Значения с префиксом `regex` интерпретируются как регулярные выражения. Значения, содержащие подстановочные знаки (`*`), также возможны, в этом случае генерируется регулярное выражение, использование не рекомендуется, и поддержка доступна только для обратной совместимости	*
hadoop.http.cross-origin.allowed-headers	Список разрешенных HTTP-заголовков, разделенных запятыми	X-Requested-With,Content-Type,Accept,Origin,WWW-Authenticate,Accept-Encoding,Transfer-Encoding
hadoop.http.cross-origin.allowed-methods	Список разрешенных методов, разделенных запятыми	GET,PUT,POST,OPTIONS,HEAD,DELETE
hadoop.http.cross-origin.max-age	Количество секунд, в течение которых может кешироваться предварительный запрос	1800
core_site.enable_cors.active	Включение CORS (Cross-Origin Resource Sharing)	true

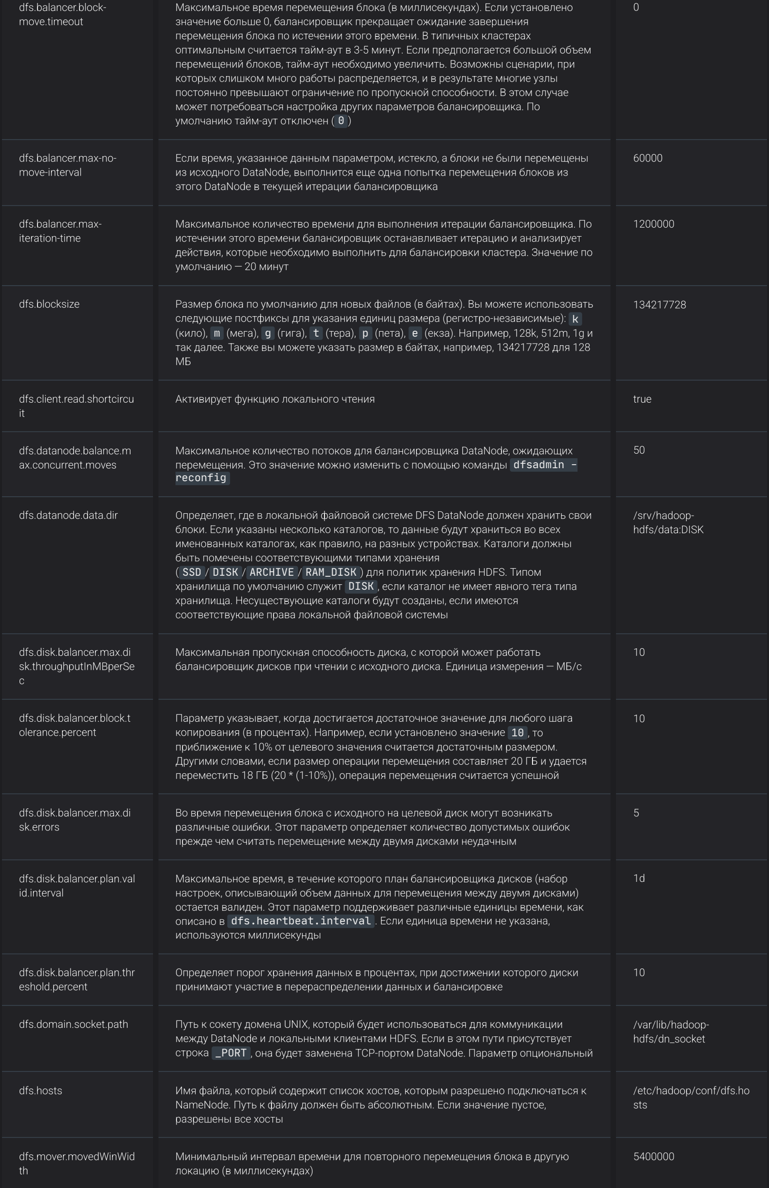
hdfs-site.xml

Параметр	**Описание**	**Значение по умолчанию**

dfs.client.block.write.replace-datanode-on-failure.enable	Если в процессе записи возникает сбой DataNode или ошибка сети, DFSClient попытается удалить отказавший DataNode из пайплайна, а затем продолжит запись с оставшимися DataNodes. Это приводит к уменьшению DataNodes в пайплайне. Решением является добавление новых DataNode в пайплайн. Данный параметр включает/отключает функцию добавления DataNode на глобальном уровне. Если кластер очень маленький, например, 3 узла или меньше, администраторы такого кластера могут установить политику `NEVER` в файле конфигурации по умолчанию или отключить эту функцию. В противном случае пользователи могут столкнуться с необычно высоким уровнем отказов пайплайна, из-за невозможности добавления новых узлов данных для замены отказавших. Чтобы получить больше информации, смотрите параметр `dfs.client.block.write.replace-datanode-on-failure.policy`	true

dfs.client.block.write.replace-datanode-on-failure.policy	Этот параметр используется, только если значение `dfs.client.block.write.replace-datanode-on-failure.enable` равно `true`.
	Допустимые значения:
	■ `ALWAYS`. Всегда добавляет новый DataNode после того как существующий DataNode удален.
	■ `NEVER`. Никогда не добавляет новый DataNode.
	■ `DEFAULT`. Пусть `r` — фактор репликации, а `n` — количество существующих DataNodes. Добавляет новый DataNode только если `r` больше или равно `3` и выполняются следующие условия:
	  ■ `floor(r/2)` больше или равно `n`;
	  ■ `r` больше `n` и блок имеет статус hflushed/appended.
		DEFAULT

dfs.client.block.write.replace-datanode-on-failure.best-effort	Используется, только если значение `dfs.client.block.write.replace-datanode-on-failure.enable` равно `true`. Клиент попытается заменить отказавший DataNode в пайплайне записи (при условии, что политика позволяет это сделать), однако продолжит операцию записи в случае, если замена DataNode также не удалась. Рассмотрим случай, когда замена DataNode не была выполнена. В таком случае значение `false` приводит к исключению и запись прерывается; значение `true` означает, что запись должна быть возобновлена с оставшимися DataNodes. Обратите внимание, что установка для этого свойства значения `true` позволяет выполнять запись в пайплайн с меньшим количеством DataNodes. В результате увеличивается вероятность потери данных	false
dfs.client.block.write.replace-datanode-on-failure.min-replication	Минимальное количество репликаций, необходимых для предотвращения сбоя пайплайна записи в случае, если новые DataNodes не могут быть найдены для замены отказавших DataNodes (например, из-за сбоя сети). Если количество оставшихся DataNodes больше или равно значению этого параметра, запись продолжается в оставшиеся узлы. В противном случае будет брошено исключение. Если используется значение `0`, будет брошено исключение в момент, когда замена не может быть найдена. Чтобы получить больше информации, смотрите параметр `dfs.client.block.write.replace-datanode-on-failure.policy`	0
dfs.balancer.dispatcherThreads	Размер пула потоков для механизма перемещения блоков балансировщика (balancer block mover) HDFS — **dispatchExecutor**	200
dfs.balancer.movedWinWidth	Временной интервал для отслеживания балансировщиком блоков и их местоположений (в миллисекундах)	5400000
dfs.balancer.moverThreads	Размер пула потоков для выполнения перемещений блоков — **moverThreadAllocator**	1000
dfs.balancer.max-size-to-move	Максимальное количество байт, которое балансировщик может переместить в одном потоке	10737418240
dfs.balancer.getBlocks.min-block-size	Минимальный размер блока (в байтах), который следует игнорировать при получении списка блоков	10485760
dfs.balancer.getBlocks.size	Общий размер блоков DataNode (в байтах), которые необходимо получить при получении исходного списка блоков	2147483648



dfs.balancer.block-move.timeout	Максимальное время перемещения блока (в миллисекундах). Если установлено значение больше 0, балансировщик прекращает ожидание завершения перемещения блока по истечении этого времени. В типичных кластерах оптимальным считается тайм-аут в 3-5 минут. Если предполагается большой объем перемещений блоков, тайм-аут необходимо увеличить. Возможны сценарии, при которых слишком много работы распределяется, и в результате многие узлы постоянно превышают ограничение по пропускной способности. В этом случае может потребоваться настройка других параметров балансировщика. По умолчанию тайм-аут отключен (`0`)	0
dfs.balancer.max-no-move-interval	Если время, указанное данным параметром, истекло, а блоки не были перемещены из исходного DataNode, выполнится еще одна попытка перемещения блоков из этого DataNode в текущей итерации балансировщика	60000
dfs.balancer.max-iteration-time	Максимальное количество времени для выполнения итерации балансировщика. По истечении этого времени балансировщик останавливает итерацию и анализирует действия, которые необходимо выполнить для балансировки кластера. Значение по умолчанию — 20 минут	1200000
dfs.blocksize	Размер блока по умолчанию для новых файлов (в байтах). Вы можете использовать следующие постфиксы для указания единиц размера (регистро-независимые): `k` (кило), `m` (мега), `g` (гига), `t` (тера), `p` (пета), `e` (екза). Например, 128k, 512m, 1g и так далее. Также вы можете указать размер в байтах, например, 134217728 для 128 МБ	134217728
dfs.client.read.shortcircuit	Активирует функцию локального чтения	true
dfs.datanode.balance.max.concurrent.moves	Максимальное количество потоков для балансировщика DataNode, ожидающих перемещения. Это значение можно изменить с помощью команды `dfsadmin -reconfig`	50
dfs.datanode.data.dir	Определяет, где в локальной файловой системе DFS DataNode должен хранить свои блоки. Если указаны несколько каталогов, то данные будут храниться во всех именованных каталогах, как правило, на разных устройствах. Каталоги должны быть помечены соответствующими типами хранения (`SSD`/`DISK`/`ARCHIVE`/`RAM_DISK`) для политик хранения HDFS. Типом хранилища по умолчанию служит `DISK`, если каталог не имеет явного тега типа хранилища. Несуществующие каталоги будут созданы, если имеются соответствующие права локальной файловой системы	/srv/hadoop-hdfs/data:DISK
dfs.disk.balancer.max.disk.throughputInMBperSec	Максимальная пропускная способность диска, с которой может работать балансировщик дисков при чтении с исходного диска. Единица измерения — МБ/с	10
dfs.disk.balancer.block.tolerance.percent	Параметр указывает, когда достигается достаточное значение для любого шага копирования (в процентах). Например, если установлено значение `10`, то приближение к 10% от целевого значения считается достаточным размером. Другими словами, если размер операции перемещения составляет 20 ГБ и удается переместить 18 ГБ (20 * (1-10%)), операция перемещения считается успешной	10
dfs.disk.balancer.max.disk.errors	Во время перемещения блока с исходного на целевой диск могут возникать различные ошибки. Этот параметр определяет количество допустимых ошибок прежде чем считать перемещение между двумя дисками неудачным	5
dfs.disk.balancer.plan.valid.interval	Максимальное время, в течение которого план балансировщика дисков (набор настроек, описывающий объем данных для перемещения между двумя дисками) остается валиден. Этот параметр поддерживает различные единицы времени, как описано в `dfs.heartbeat.interval`. Если единица времени не указана, используются миллисекунды	1d
dfs.disk.balancer.plan.threshold.percent	Определяет порог хранения данных в процентах, при достижении которого диски принимают участие в перераспределении данных и балансировке	10
dfs.domain.socket.path	Путь к сокету домена UNIX, который будет использоваться для коммуникации между DataNode и локальными клиентами HDFS. Если в этом пути присутствует строка `_PORT`, она будет заменена TCP-портом DataNode. Параметр опциональный	/var/lib/hadoop-hdfs/dn_socket
dfs.hosts	Имя файла, который содержит список хостов, которым разрешено подключаться к NameNode. Путь к файлу должен быть абсолютным. Если значение пустое, разрешены все хосты	/etc/hadoop/conf/dfs.hosts
dfs.mover.movedWinWidth	Минимальный интервал времени для повторного перемещения блока в другую локацию (в миллисекундах)	5400000

dfs.mover.moverThreads	Устанавливает размер пула потоков балансировщика	1000
dfs.mover.retry.max.attempts	Максимальное количество повторных попыток, прежде чем mover посчитает перемещение неудачным	10
dfs.mover.max-no-move-interval	Указывает временной отрезок. Если время истекло, и ни один блок не был перемещен из исходного DataNode, будет предпринята еще одна попытка переместить блоки из этого DataNode в текущей итерации перемещения	60000
dfs.namenode.name.dir	Определяет, где в локальной файловой системе DFS NameNode должен хранить таблицу имен (fsimage). Если указаны несколько каталогов, то таблица имен реплицируется во все каталоги для избыточности	/srv/hadoop-hdfs/name
dfs.namenode.checkpoint.dir	Определяет, где в локальной файловой системе Secondary DFS NameNode должен хранить временные образы для слияния. Если указаны несколько каталогов, то образ реплицируется во все каталоги для избыточности	/srv/hadoop-hdfs/checkpoint
dfs.namenode.hosts.provider.classname	Имя класса, предоставляющего доступ к хост-файлам. По умолчанию используется `org.apache.hadoop.hdfs.server.blockmanagement.HostFileManager`, который загружает файлы, указанные в `dfs.hosts` и `dfs.hosts.exclude`. Если используется `org.apache.hadoop.hdfs.server.blockmanagement.CombinedHostFileManager`, этот класс загружает файл JSON, определенный в `dfs.hosts`. Чтобы изменить имя класса, требуется перезапуск NameNode. Команда `dfsadmin -refreshNodes` обновляет только файлы конфигурации, используемые классом	org.apache.hadoop.hdfs.server.blockmanagement.CombinedHostFileManager
dfs.namenode.rpc-bind-host	Фактический адрес, к которому будет привязан сервер RPC. Если этот необязательный адрес установлен, он переопределяет только часть имени хоста `dfs.namenode.rpc-address`. Его также можно указать для NameNode или службы имен для режимов высокой доступности или федерации. При установке значения `0.0.0.0` NameNode слушает все интерфейсы	0.0.0.0
dfs.permissions.superusergroup	Имя группы суперпользователей. В значении должно быть указано имя одной группы	hadoop
dfs.replication	Репликация блока по умолчанию. Фактическое количество репликаций можно указать при создании файла. Используется значение по умолчанию, если репликация не указана во время создания файла	3
dfs.journalnode.http-address	HTTP-адрес веб-интерфейса JournalNode	0.0.0.0:8480
dfs.journalnode.https-address	HTTPS-адрес веб-интерфейса JournalNode	0.0.0.0:8481
dfs.journalnode.rpc-address	RPC-адрес веб-интерфейса JournalNode	0.0.0.0:8485
dfs.datanode.http.address	Адрес HTTP-сервера DataNode	0.0.0.0:9864
dfs.datanode.https.address	Адрес HTTPS-сервера DataNode	0.0.0.0:9865
dfs.datanode.address	Адрес DataNode для передачи данных	0.0.0.0:9866
dfs.datanode.ipc.address	IPC-адоес DataNode	0.0.0.0:9867
dfs.namenode.http-address	Адрес веб-интерфейса dfs NameNode	0.0.0.0:9870
dfs.namenode.https-address	HTTPS-адрес NameNode	0.0.0.0:9871
dfs.ha.automatic-failover.enabled	Определяет, включен ли механизм автоматической обработки отказа	true
dfs.ha.fencing.methods	Список скриптов или Java-классов, которые будут использоваться для восстановления Active NameNode во время обработки отказа	shell(/bin/true)

dfs.journalnode.edits.dir	Каталог, в котором хранятся edit-файлы журнала	/srv/hadoop-hdfs/journalnode
dfs.namenode.shared.edits.dir	Каталог в хранилище, которое расшарено между несколькими NameNode в кластере, работающем в режиме высокой доступности. В этот каталог данные будут записываться активными узлами и считываться резервными, чтобы поддерживать синхронизацию пространств имен. Этот каталог не нужно указывать в `dfs.namenode.edits.dir`. В кластере без высокой доступности данный параметр следует оставить пустым	---
dfs.internal.nameservices	Уникальный идентификатор служб имен для кластера или федераций. Для одиночного кластера — укажите имя, которое будет использоваться как алиас. Для федерации HDFS — укажите через запятую все пространства имен (namespace), связанные с этим кластером. Этот параметр позволяет использовать алиас вместо IP-адреса или FQDN для некоторых команд, например: `hdfs dfs -ls hdfs://<dfs.internal.nameservices>`. Может содержать только латинские буквы и цифры	—
dfs.block.access.token.enable	Если установлено значение `true`, токены доступа используются для доступа к узлам данных. Если установлено значение `false`, токены доступа не проверяются при доступе к узлам данных	false
dfs.namenode.kerberos.principal	Имя принципала NameNode-сервиса. Обычно используется значение `nn/_HOST@REALM.TLD`. Каждый NameNode заменяет `_HOST` своим собственным полным именем хоста во время запуска. Токен `_HOST` позволяет использовать один и тот же параметр конфигурации на обоих NameNodes в режиме высокой доступности	nn/_HOST@REALM
dfs.namenode.keytab.file	Файл keytab, используемый каждым демоном NameNode для входа в качестве принципала. Имя принципала настраивается с помощью `dfs.namenode.kerberos.principal`	/etc/security/keytabs/nn.service.keytab
dfs.namenode.kerberos.internal.spnego.principal	Имя принципала HTTP Kerberos для NameNode	HTTP/_HOST@REALM
dfs.web.authentication.kerberos.principal	Имя принципала Kerberos для WebHDFS	HTTP/_HOST@REALM
dfs.web.authentication.kerberos.keytab	Kerberos keytab-файл для WebHDFS	/etc/security/keytabs/HTTP.service.keytab
dfs.journalnode.kerberos.principal	Имя принципала JournalNode. Обычно используется значение `jn/_HOST@REALM.TLD`. Каждый JournalNode заменяет `_HOST` своим собственным полным именем хоста при запуске. Токен `_HOST` позволяет использовать один и тот же параметр конфигурации на всех JournalNodes	jn/_HOST@REALM
dfs.journalnode.keytab.file	Файл keytab, используемый каждым демоном JournalNode для входа в качестве сервисного принципала. Имя принципала задается с помощью `dfs.journalnode.kerberos.principal`	/etc/security/keytabs/jn.service.keytab
dfs.journalnode.kerberos.internal.spnego.principal	Принципал сервера, используемый HTTP-сервером JournalNode для проверки подлинности SPNEGO, когда включена защита Kerberos. Обычно используется значение `HTTP/_HOST@REALM.TLD`. Принципал сервера SPNEGO по соглашению начинается с префикса `HTTP/`. Если используется значение `*`, веб-сервер попытается войти в систему с каждым принципалом, указанным в keytab-файле `dfs.web.authentication.kerberos.keytab`. Для большинства систем можно установить значение `${dfs.web.authentication.kerberos.principal}`, то есть использовать значение `dfs.web.authentication.kerberos.principal`	HTTP/_HOST@REALM
dfs.datanode.data.dir.perm	Разрешения для каталогов в локальной файловой системе, где DFS DataNode хранит свои блоки. Разрешения могут быть восьмеричными или символьными	700
dfs.datanode.kerberos.principal	Принципал DataNode. Обычно используется значение `dn/_HOST@REALM.TLD`. Каждый DataNode заменяет `_HOST` своим собственным полным именем хоста при запуске. Токен `_HOST` позволяет использовать один и тот же параметр конфигурации на всех DataNodes	dn/_HOST@REALM.TLD
dfs.datanode.keytab.file	Файл keytab, используемый каждым демоном DataNode для входа в качестве принципала. Имя принципала задается с помощью `dfs.datanode.kerberos.principal`	/etc/security/keytabs/dn.service.keytab

dfs.http.policy	Определяет, поддерживается ли HTTPS (SSL) в HDFS. Параметр определяет HTTP-endpoint для демонов HDFS. Поддерживаются следующие значения: `HTTP_ONLY` — доступен только протокол http; `HTTPS_ONLY` — доступен только https; `HTTP_AND_HTTPS` — доступны http и https	HTTP_ONLY
dfs.data.transfer.protection	Разделенный запятыми список значений защиты SASL, используемых для защищенных соединений с DataNode при чтении или записи блоков данных. Возможные значения: - `authentication` — обеспечивает только аутентификацию; целостность или конфиденциальность не поддерживаются; - `integrity` — аутентификация и целостность включены; - `privacy` — включены аутентификация, целостность и конфиденциальность. Если `dfs.encrypt.data.transfer=true`, то он заменяет параметр `dfs.data.transfer.protection` и требует, чтобы все соединения использовали специальный зашифрованный SASL-handshake. Это свойство игнорируется для соединений с DataNode, которые слушают привилегированный порт. В этом случае предполагается, что использование привилегированного порта обеспечивает достаточное доверие	—
dfs.encrypt.data.transfer	Определяет, должны ли фактические данные блока, которые считываются/записываются из/в HDFS, шифроваться в сети. Этот параметр нужно устанавливать только для NameNodes и DataNodes, клиенты сделают это автоматически. Можно переопределить этот параметр для каждого соединения, указав пользовательскую логику через `dfs.trustedchannel.resolver.class`	false
dfs.encrypt.data.transfer.algorithm	Это значение может быть установлено как `3des`, так и `rc4`. Если значение не установлено, используется настроенный по умолчанию JCE в системе (обычно 3DES). Широко распространено мнение, что 3DES более безопасен, но RC4 значительно быстрее. Обратите внимание, что если AES поддерживается и клиентом, и сервером, то этот алгоритм шифрования будет использоваться только для первоначального обмена ключами AES	3des
dfs.encrypt.data.transfer.cipher.suites	Значение параметра может быть либо пустым, либо `AES`/`CTR`/`NoPadding`. Если значение задано, то `dfs.encrypt.data.transfer` использует указанный набор шифров для шифрования данных. Если не задано, используется только алгоритм, указанный в `dfs.encrypt.data.transfer.algorithm`	—
dfs.encrypt.data.transfer.cipher.key.bitlength	Длина ключа в битах, согласованная dfsclient и datanode для шифрования. Это значение может быть установлено на `128`, `192` или `256`	128
ignore.secure.ports.for.testing	Позволяет игнорировать HTTPS-требования в SASL-режиме	false
dfs.client.https.need-auth	Определяет, требуется ли аутентификация сертификата клиента SSL	false

httpfs-site.xml

Параметр	**Описание**	**Значение по умолчанию**
httpfs.http.administrators	Список управления доступом для администраторов. Используется для управления доступом к сервлетам по умолчанию для сервера HttpFS. Значение должно содержать список пользователей и групп, разделенных запятыми. Список пользователей идет первым и отделяется пробелом, за которым следует список групп, например: `user1,user2 group1,group2`. Указание пользователей и групп являются опциональным, поэтому вы можете определить как пользователей, так и группы, или и то, и другое. Обратите внимание, что во всех этих случаях вы всегда должны использовать начальный пробел в списке групп. Использование символа `*` предоставляет доступ всем пользователям и группам	*
hadoop.http.temp.dir	Каталог HttpFS для временных файлов	${hadoop.tmp.dir}/httpfs
httpfs.ssl.enabled	Определяет, включен ли SSL. По умолчанию установлено значение `false`, то есть SSL отключен	false

httpfs.hadoop.config.dir	Расположение каталога конфигурации Hadoop	/etc/hadoop/conf
httpfs.hadoop.authentication.type	Определяет механизм аутентификации, используемый httpfs для HTTP-клиентов. Допустимые значения: `simple` и `kerberos`. Если используется `simple`, клиенты должны указать имя пользователя с параметром строки запроса `user.name`. Если используется `kerberos`, клиенты должны использовать HTTP SPNEGO или токены делегирования	simple
httpfs.hadoop.authentication.kerberos.keytab	Kerberos keytab-файл с учетными данными для принципала HTTP Kerberos, используемого httpfs в HTTP-endpoint. Параметр `httpfs.authentication.kerberos.keytab` устарел. Вместо этого используйте `hadoop.http.authentication.kerberos.keytab`	/etc/security/keytabs/httpfs.service.keytab
httpfs.hadoop.authentication.kerberos.principal	Принципал HTTP Kerberos, используемый HttpFS в HTTP-endpoint. Имя принципала HTTP Kerberos должно начинаться с `HTTP/` в соответствии со спецификацией Kerberos HTTP SPNEGO. Параметр `httpfs.authentication.kerberos.principal` устарел. Вместо этого используйте `hadoop.http.authentication.kerberos.principal`	HTTP/${httpfs.hostname}@${kerberos.realm}

ranger-hdfs-audit.xml

Параметр	Описание	Значение по умолчанию
xasecure.audit.destination.solr.batch.filespool.dir	Путь к Spool-каталогу	/srv/ranger/hdfs_plugin/audit_solr_spool
xasecure.audit.destination.solr.urls	URL сервера Solr для сохранения аудит-событий. Оставьте значение пустым или установите значение `NONE` при использовании ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.zookeepers	Указывает строку подключения ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.force.use.inmemory.jaas.config	Использует in-memory JAAS-файл для соединения с Solr	—
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.jaas.Client.loginModuleControlFlag	Определяет требование к успешному завершению работы модуля. Значения могут быть `required`, `requisite`, `sufficient` или `optional`	—
xasecure.audit.jaas.Client.loginModuleName	Имя класса-аутентификатора	—
xasecure.audit.jaas.Client.option.keyTab	Имя keytab-файла, из которого необходимо получить секретный ключ принципала	—
xasecure.audit.jaas.Client.option.principal	Имя используемого принципала	—
xasecure.audit.jaas.Client.option.serviceName	Имя пользователя или сервиса, для которого выполнятся логин	—
xasecure.audit.jaas.Client.option.storeKey	Установите значение `true`, чтобы keytab-файл или ключ принципала сохранялись в учетных данных субъекта	false
xasecure.audit.jaas.Client.option.useKeyTab	Установите значение `true`, чтобы модуль получал ключ принципала из keytab-файла	false

ranger-hdfs-security.xml

Параметр	Описание	Значение по умолчанию
ranger.plugin.hdfs.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.hdfs.service.name	Имя Ranger-сервиса, который содержит политики для этого экземпляра	—
ranger.plugin.hdfs.policy.cache.dir	Каталог, в котором кешируются политики Ranger после успешного извлечения из источника	/srv/ranger/hdfs/policycache
ranger.plugin.hdfs.policy.pollIntervalMs	Определяет, как часто проверять политики на предмет изменения	30000
ranger.plugin.hdfs.policy.rest.client.connection.timeoutMs	Время ожидания подключения HDFS-плагина RangerRestClient (в миллисекундах)	120000
ranger.plugin.hdfs.policy.rest.client.read.timeoutMs	Тайм-аут на чтение для HDFS-плагина RangerRestClient (в миллисекундах)	30000
ranger.plugin.hdfs.policy.rest.ssl.config.file	Путь к конфигурационному файлу RangerRestClient для HDFS-плагина	/etc/hadoop/conf/ranger-hdfs-policymgr-ssl.xml

httpfs-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые должны быть добавлены в *httpfs-env.sh*	—
HADOOP_CONF_DIR	Каталог конфигурации Hadoop	/etc/hadoop/conf
HADOOP_LOG_DIR	Расположение каталога с логами	${HTTPFS_LOG}
HADOOP_PID_DIR	Каталог с PID-файлами Hadoop	${HTTPFS_TEMP}
HTTPFS_SSL_ENABLED	Определяет, включен ли SSL для httpfs	false
HTTPFS_SSL_KEYSTORE_FILE	Путь к keystore-файлу	admin
HTTPFS_SSL_KEYSTORE_PASS	Пароль для доступа к keystore-файлу	admin
Final HTTPFS_ENV_OPTS	Итоговое значение параметра `HTTPFS_ENV_OPTS` в *httpfs-env.sh*	—

hadoop-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые должны быть добавлены в *hadoop-env.sh*	—
HDFS_NAMENODE_OPTS	NameNode Heap Memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap и параметры среды для NameNode	-Xms1G -Xmx8G
HDFS_DATANODE_OPTS	DataNode Heap Memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap и параметры среды для DataNode	-Xms700m -Xmx8G

HDFS_HTTPFS_OPTS	HttpFS Heap Memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap и параметры среды для httpfs	-Xms700m -Xmx8G
HDFS_JOURNALNODE_OPTS	JournalNode Heap Memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap и параметры среды для JournalNode	-Xms700m -Xmx8G
HDFS_ZKFC_OPTS	ZKFC Heap Memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap и параметры среды для ZKFC	-Xms500m -Xmx8G
Final HADOOP_ENV_OPTS	Итоговое значение параметра `HADOOP_ENV_OPTS` в *hadoop-env.sh*	—

ssl-server.xml

Параметр	Описание	Значение по умолчанию
ssl.server.truststore.location	Truststore-файл, используемый NameNodes и DataNodes	—
ssl.server.truststore.password	Пароль к truststore-файлу	—
ssl.server.truststore.type	Формат truststore-файла	jks
ssl.server.truststore.reload.interval	Интервал между проверками на обновление truststore-файла (в миллисекундах)	10000
ssl.server.keystore.location	Путь к keystore-файлу, используемому NameNodes и DataNodes	—
ssl.server.keystore.password	Пароль к keystore-файлу	—
ssl.server.keystore.keypassword	Пароль к ключу в keystore-файле	—
ssl.server.keystore.type	Формат keystore-файла	—

Lists of decommissioned and in maintenance hosts

Параметр	Описание	Значение по умолчанию
DECOMMISSIONED	Когда администратор выводит DataNode из эксплуатации, DataNode сначала переходит в состояние `DECOMMISSION_INPROGRESS`. После того, как все блоки, принадлежащие этому DataNode, полностью реплицированы на другие хосты с учетом коэффициента репликации каждого блока, DataNode переходит в состояние `DECOMMISSIONED`. После этого администратор может отключить узел для выполнения долгосрочного ремонта и обслуживания, которое может занять несколько дней или недель. После обслуживания машина может быть снова введена в эксплуатацию в кластере	—
IN_MAINTENANCE	Иногда администраторам нужно отключить DataNodes всего на несколько минут/часов для выполнения краткосрочного ремонта или обслуживания. В таких сценариях накладные расходы на репликацию блоков HDFS, связанные с выводом из эксплуатации, могут не понадобиться, и желателен облегченный процесс. Для этого используется состояние обслуживания (`IN_MAINTENANCE`). Когда администратор переводит DataNode в состояние обслуживания, DataNode сначала переходит в состояние `ENTERING_MAINTENANCE`. После того, как все блоки,	—

	принадлежащие этому DataNode, минимально реплицируются на других хостах, DataNode будет немедленно переведен в состояние `IN_MAINTENANCE`. После завершения обслуживания администратор может вывести DataNode из состояния обслуживания. Кроме того, состояние обслуживания поддерживает тайм-аут, который позволяет администраторам настраивать максимальную продолжительность, в течение которой DataNode может оставаться в состоянии обслуживания. По истечении тайм-аута DataNode будет автоматически переведен из состояния обслуживания HDFS без вмешательства человека	

Другие параметры

Параметр	Описание	Значение по умолчанию
Additional nameservices	Дополнительные (внутренние) имена для кластера HDFS, позволяющие запрашивать другой кластер HDFS из текущего	—
Custom core-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *core-site.xml*	—
Custom hdfs-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hdfs-site.xml*	—
Custom httpfs-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *httpfs-site.xml*	—
Ranger plugin enabled	Определяет, активен ли Ranger-плагин	—
Custom ranger-hdfs-audit.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hdfs-audit.xml*	—
Custom ranger-hdfs-security.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hdfs-security.xml*	—
Custom ranger-hdfs-policymgr-ssl.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hdfs-policymgr-ssl.xml*	—
Custom httpfs-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *httpfs-env.sh*	—
Custom hadoop-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hadoop-env.sh*	—
Custom ssl-server.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ssl-server.xml*	—
Custom ssl-client.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ssl-client.xml*	—

Topology script	Скрипт топологии, используемый в HDFS	—
Topology data	Опциональный параметр. Текстовый файл, используемый для сопоставления имен хостов и количества стоек (rack) для скрипта топологии. Сохраняется в */etc/hadoop/conf/topology.data*	—
Custom log4j.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *log4j.properties*	log4j.properties
Custom httpfs-log4j.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *httpfs-log4j.properties*	httpfs-log4j.properties

Компонент HDFS DataNode

Monitoring

Параметр	Описание	Значение по умолчанию
Java agent path	Путь к Java-агенту JMX Prometheus	/usr/lib/adh-utils/jmx/jmx_prometheus_javaagent.jar
Prometheus metrics port	Порт, на котором будут доступны метрики HDFS DataNode в формате Prometheus	9202
Mapping config path	Путь к файлу конфигурации мэппинга метрик	/etc/hadoop/conf/jmx_hdfs_datanode_metric_config.yml
Mapping config	Файл конфигурации мэппинга метрик	hdfs-mapping-config.yml

Компонент HDFS JournalNode

Monitoring

Параметр	Описание	Значение по умолчанию
Java agent path	Путь к Java-агенту JMX Prometheus	/usr/lib/adh-utils/jmx/jmx_prometheus_javaagent.jar
Prometheus metrics port	Порт, на котором будут доступны метрики HDFS JournalNode в формате Prometheus	9203
Mapping config path	Путь к файлу конфигурации мэппинга метрик	/etc/hadoop/conf/jmx_hdfs_journalnode_metric_config.yml
Mapping config	Файл конфигурации мэппинга метрик	hdfs-mapping-config.yml

Компонент HDFS NameNode

Monitoring

Параметр	Описание	Значение по умолчанию
Java agent path	Путь к Java-агенту JMX Prometheus	/usr/lib/adh-utils/jmx/jmx_prometheus_javaagent.jar
Prometheus metrics port	Порт, на котором будут доступны метрики HDFS NameNode в формате Prometheus	9201

Mapping config path	Путь к файлу конфигурации мэппинга метрик	/etc/hadoop/conf/jmx_hdfs_namenode_metric_config.yml
Mapping config	Файл конфигурации мэппинга метрик	hdfs-mapping-config.yml

Hive

hive-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые должны быть добавлены в *hive-env.sh*	—
HADOOP_CLASSPATH	Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	▪ /etc/tez/conf/ ▪ /usr/lib/tez/* ▪ /usr/lib/tez/lib/*
HIVE_HOME	Домашний каталог Hive	/usr/lib/hive
METASTORE_PORT	Порт Hive Metastore	9083
HADOOP_CLIENT_OPTS	Клиентские параметры Hadoop. Например, параметры запуска JVM	$HADOOP_CLIENT_OPTS -Djava.io.tmpdir={{ cluster.config.java_tmpdir | d('/tmp') }}

hive-server2-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые должны быть добавлены в *hive-server2-env.sh*	—
HADOOP_CLIENT_OPTS	Параметры клиента для HiveServer2	-Xms256m -Xmx256m
HIVE_AUX_JARS_PATH	Позволяет добавить дополнительные JAR-файлы в classpath Hive. Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	—
Final HIVE_SERVER2_ENV_OPTS	Итоговое значение параметра `HIVE_SERVER2_ENV_OPTS` в *hive-server2-env.sh*	—

hive-metastore-env.sh

Параметр	Описание	Значение по умолчанию
Sources	Список источников, которые должны быть добавлены в *hive-metastore-env.sh*	—
HADOOP_CLIENT_OPTS	Параметры клиента для Hive Metastore	-Xms256m -Xmx256m
Final HIVE_METASTORE_ENV_OPTS	Итоговое значение параметра `HIVE_METASTORE_ENV_OPTS` в *hive-metastore-env.sh*	—

Credential Encryption

Параметр	Описание	Значение по умолчанию
Encryption enable	Включает или выключает функцию шифрования учетных данных. Если опция включена, Hive хранит учетные данные (пароли настроек, учетные данные для взаимодействия с другими сервисами) в зашифрованном виде	false
Credential provider path	Путь к хранилищу ключей JCEKS	jceks://file/etc/hive/conf/hive.jceks
Ranger plugin credential provider path	Путь к хранилищу ключей JCEKS, используемому Ranger	jceks://file/etc/hive/conf/ranger-hive.jceks
Custom jceks	Укажите значение `true` при использовании кастомного файла JCEKS-хранилища. Укажите значение `false` для использования дефолтного файла хранилища, который генерируется автоматически	false
Password file name	Имя файла в classpath сервиса, в котором хранятся пароли	hive_credstore_pass

hive-site.xml

Параметр	Описание	Значение по умолчанию
hive.cbo.enable	Если установлено значение `true`, включает оптимизатор (cost-based optimizer), использующий фреймворк Calcite	true
hive.compute.query.using.stats	Если установлено значение `true`, Hive будет отвечать на несколько запросов, таких как `min`, `max` и `count (1)`, используя статистику, хранящуюся в Metastore. Для сбора базовой статистики установите `hive.stats.autogather=true`. Для более расширенного сбора статистики выполните запросы `ANALYZE TABLE`	false
hive.execution.engine	Задает тип используемого движка. Допустимые значения: `mr` (MapReduce, значение по умолчанию), `tez` (движок Tez, только для Hadoop 2) или `spark` (движок Spark, для Hive 1.1.0 и более поздних версий)	Tez
hive.log.explain.output	Когда этот параметр задан, выходные данные команды `EXPLAIN EXTENDED` добавляются в log4j-лог с уровнем INFO, а также отображаются в веб-интерфейсе HiveServer2 (***Drilldown*** → ***Query Plan***). Начиная с Hive 3.1.0, этот параметр логируется только как log4j INFO. Чтобы сделать вывод команды `EXPLAIN EXTENDED` доступным в веб-интерфейсе в Hive 3.1.0 и более поздних версиях, используйте `hive.server2.webui.explain.output`	true
hive.metastore.event.db.notification.api.auth	Определяет, должен ли Metastore выполнять авторизацию с использованием database notification API, таких как `get_next_notification`. Если установлено значение `true`, то только суперпользователи в настройках прокси имеют разрешение	false
hive.metastore.uris	URI, используемый для доступа к метаданным в удаленном Metastore. Для удаленного хранилища метаданных необходимо указать URI сервера хранилища метаданных Thrift: *thrift://<hostname>:<port>*, где *<hostname>* — имя или IP-адрес сервера хранилища метаданных Thrift, *<port>* — порт, который слушает сервер Thrift	—
hive.metastore.warehouse.dir	Абсолютный путь к файлу HDFS базы данных по умолчанию для хранилища, который является локальным для кластера	/apps/hive/warehouse
hive.server2.enable.doAs	Активирует соединение от имени другого пользователя (Impersonation)	false

hive.stats.fetch.column.stats	Для аннотации дерева операторов со статистической информацией нужна статистика по столбцам. Статистика столбцов извлекается из хранилища метаданных. Извлечение статистики для каждого столбца может быть дорогостоящей операцией при большом количестве столбцов. Этот параметр можно использовать для отключения получения статистики столбцов из хранилища метаданных	false
hive.tez.container.size	По умолчанию Tez будет создавать контейнеры такого размера, как и Mapper. Этот параметр можно использовать для перезаписи значения по умолчанию	1024
hive.support.concurrency	Определяет, должен ли Hive поддерживать параллелизм. Экземпляр ZooKeeper должен быть запущен, чтобы Hive Lock Manager по умолчанию поддерживал блокировки чтения/записи	false
hive.txn.manager	Установите значение для этого параметра `org.apache.hadoop.hive.ql.lockmgr.DbTxnManager`, что является одним из шагов включения транзакций Hive. DummyTxnManage по умолчанию имеет поведение версий Hive до 0.13 и не предоставляет транзакций	org.apache.hadoop.hive.ql.lockmgr.DummyTxnManager
hive.users.in.admin.role	Список пользователей, которым будет присвоена роль администратора при старте Metastore	—
javax.jdo.option.ConnectionUserName	Имя пользователя базы данных metastore	APP
javax.jdo.option.ConnectionPassword	Пароль пользователя metastore	—
javax.jdo.option.ConnectionURL	JDBC URI для соединения, используемый для доступа к данным, которые хранятся в локальном metastore. Используйте следующий URI подключения: *jdbc:<datastore type>://<node name>:<port>/<database name>*, где: - *<node name>* — имя хоста или IP-адрес data store; - *<data store type>* — тип data store; - *<port>* — номер порта, который слушает data store для удаленного вызова процедур; - *<database name>* — имя базы данных. Например, следующий URI указывает на локальный Metastore, который использует MySQL в качестве хранилища данных: `jdbc:mysql://hostname23:3306/metastore`	jdbc:postgresql://{{ groups['adpg.adpg'][0] | d(omit) }}:5432/hive
javax.jdo.option.ConnectionDriverName	Имя класса JDBC-драйвера для доступа к Hive Metastore	org.postgresql.Driver
hive.server2.transport.mode	Устанавливает режим передачи данных	binary
hive.server2.thrift.port	Номер порта, используемого для бинарного (binary) соединения с Thrift Server2	10000
hive.server2.thrift.http.port	Номер порта, используемого для HTTP-соединения с Thrift Server2	10001
hive.server2.thrift.http.path	HTTP-endpoint сервиса Thrift Server2	cliservice
hive.metastore.transactional.event.listeners	Listener-класс, который сохраняет события в базе данных	org.apache.hive.hcatalog.listener.DbNotificationListener
hive.metastore.dml.events	Включает или отключает отслеживание DML-событий	true
hive.server2.authentication.kerberos.principal	Принципал Kerberos для HiveServer2	—

hive.server2.authentication.kerberos.keytab	Путь к keytab-файлу Kerberos, который содержит принципал для HiveServer2	—
hive.server2.authentication.spnego.principal	Kerberos-принципал SPNEGO	—
hive.server2.webui.spnego.principal	Kerberos-принципал SPNEGO для доступа к веб-интерфейсу	—
hive.server2.webui.spnego.keytab	Keytab-файл SPNEGO для доступа к веб-интерфейсу	—
hive.server2.webui.use.spnego	Определяет, использовать ли Kerberos SPNEGO для доступа к веб-интерфейсу	false
hive.server2.authentication.spnego.keytab	Путь к принципалу SPNEGO	—
hive.server2.authentication	Устанавливает режим аутентификации	NONE
hive.metastore.sasl.enabled	Если значение `true`, интерфейс Metastore Thrift защищен с помощью SASL. Клиенты должны аутентифицироваться с помощью Kerberos	false
hive.metastore.kerberos.principal	Сервисный Kerberos-принципал для хранилища метаданных Thrift-сервера. Строка `_HOST` автоматически заменяется актуальным именем хоста	—
hive.metastore.kerberos.keytab.file	Путь к Kerberos keytab-файлу, который содержит сервисный принципал для хранилища метаданных сервера Thrift	—
hive.server2.use.SSL	Определяет, использовать ли SSL для HiveServer2	false
hive.server2.keystore.path	Keystore-файл, используемый HiveServer2	—
hive.server2.keystore.password	Пароль к keystore-файлу, который использует HiveServer2	—
hive.server2.truststore.path	Truststore-файл, используемый HiveServer2	—
hive.server2.truststore.password	Пароль для доступа к truststore-файлу HiveServer2	—
hive.server2.webui.use.ssl	Определяет, использовать ли SSL для веб-интерфейса Hive	false
hive.server2.webui.keystore.path	Путь к keystore-файлу, который используется для доступа к веб-интерфейсу Hive	—
hive.server2.webui.keystore.password	Пароль к keystore-файлу, который используется для доступа к веб-интерфейсу Hive	—
hive.ssl.protocol.blacklist	Список версий TLS, запрещенных для использования в Hive	SSLv2Hello,SSLv3,TLSv1,TLSv1.1
metastore.keystore.path	Путь к keystore-файлу компонента Hive Metastore	—
metastore.keystore.password	Пароль для доступа к keystore-файлу компонента Hive Metastore	—
metastore.truststore.path	Путь к truststore-файлу компонента Hive Metastore	—
metastore.truststore.password	Пароль для доступа к truststore-файлу компонента Hive Metastore	—

metastore.use.SSL	Определяет, использовать ли SSL для взаимодействия с Hive Metastore	false
metastore.ssl.protocol.blacklist	Список версий TLS, запрещенных для использования с Hive Metastore	SSLv2Hello,SSLv2,SSLv3,TLSv1,TLSv1.1
iceberg.engine.hive.enabled	Включает поддержку Iceberg-таблиц	true
hive.security.authorization.sqlstd.confwhitelist.append	Регулярное выражение для добавления конфигурационных свойств в список разрешенных дополнительно к `hive.security.authorization.sqlstd.confwhitelist`	kyuubi\.operation\.handle|kyuubi\.client\.version|kyuubi\.client\.ipAddress|tez\.application\.tags
hive.server2.support.dynamic.service.discovery	Активирует поддержку динамического обнаружения сервисов через ZooKeeper	true
hive.zookeeper.quorum	Разделенный запятыми список ZooKeeper-серверов (<host>:<port>), запущенных в кластере	—
hive.server2.zookeeper.namespace	Пространство имен блокировок для ZooKeeper	hiveserver2
hive.cluster.delegation.token.store.class	Имя класса, реализующего систему хранилища токенов делегации (delegation token store)	org.apache.hadoop.hive.metastore.security.ZooKeeperTokenStore

ACID Housekeeping

Параметр	Описание	Значение по умолчанию
Enable Housekeeping Thread	Определяет, выполняются ли фоновые задачи обслуживания в Hive Metastore	true
Cleaner Interval	Интервал, через который запускаются фоновые потоки очистки для ACID-транзакций	10s
Leader Election Mode	Определяет, использовать ли хост, заданный параметром `Housekeeping Leader Hostname` в качестве лидера обслуживания (значение `host`). При значении `lock` Hive Metastore должен использовать блокировки (lock) для выбора лидера	host
Housekeeping Leader Hostname	FQDN лидера обслуживания	—
ZK Lock Namespace	Определят пространство имен блокировок для ZooKeeper	—
Compactor Initiator	Определяет, запускать ли поток `compaction-initiator`	true
Compactor Cleaner	Определяет, запускать ли поток `compaction-cleaner`	true

Custom log4j.properties

Параметр	Описание	Значение по умолчанию
HiveServer2 hive-log4j.properties	Конфигурация Log4j, используемая для логирования процессов HiveServer2	hive-log4j.properties
Hive Metastore hive-log4j2.properties	Конфигурация Log4j2, используемая для логирования процессов Hive Metastore	hive-log4j2.properties
Hive Beeline beeline-log4j2.properties	Конфигурация Log4j2, используемая для логирования процессов Hive Beeline	beeline-log4j2.properties

ranger-hive-audit.xml

Параметр	Описание	Значение по умолчанию
xasecure.audit.destination.solr.batch.filespool.dir	Путь к Spool-каталогу	/srv/ranger/hdfs_plugin/audit_solr_spool
xasecure.audit.destination.solr.urls	URL сервера Solr для сохранения аудит-событий. Оставьте значение пустым или установите значение `NONE` при использовании ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.zookeepers	Указывает строку подключения ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.force.use.inmemory.jaas.config	Использует in-memory JAAS-файл для соединения с Solr	—
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.jaas.Client.loginModuleControlFlag	Определяет требование к успешному завершению работы модуля. Значения могут быть `required`, `requisite`, `sufficient` или `optional`	—
xasecure.audit.jaas.Client.loginModuleName	Имя класса-аутентификатора	—
xasecure.audit.jaas.Client.option.keyTab	Имя keytab-файла, из которого необходимо получить секретный ключ принципала	—
xasecure.audit.jaas.Client.option.principal	Имя используемого принципала	—
xasecure.audit.jaas.Client.option.serviceName	Имя пользователя или сервиса, для которого выполнятся логин	—
xasecure.audit.jaas.Client.option.storeKey	Установите значение `true`, чтобы keytab-файл или ключ принципала сохранялись в учетных данных субъекта	false
xasecure.audit.jaas.Client.option.useKeyTab	Установите значение `true`, чтобы модуль получал ключ принципала из keytab-файла	false

ranger-hive-security.xml

Параметр	Описание	Значение по умолчанию
ranger.plugin.hive.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.hive.service.name	Имя Ranger-сервиса, который содержит политики для этого экземпляра	—
ranger.plugin.hive.policy.cache.dir	Каталог, в котором кешируются политики Ranger после успешного извлечения из источника	/srv/ranger/hive/policycache
ranger.plugin.hive.policy.pollIntervalMs	Определяет, как часто проверять политики на предмет изменения	30000
ranger.plugin.hive.policy.rest.client.connection.timeoutMs	Время ожидания подключения Hive-плагина RangerRestClient (в миллисекундах)	120000

ranger.plugin.hive.policy.rest.client.read.timeoutMs	Тайм-аут на чтение для Hive-плагина RangerRestClient (в миллисекундах)	30000
xasecure.hive.update.xapolicies.on.grant.revoke	Определяет, разрешать ли обновление политик для SQL-команд Grant/Revoke	true
ranger.plugin.hive.policy.rest.ssl.config.file	Путь к конфигурационному файлу RangerRestClient SSL плагина Hive	/etc/hive/conf/ranger-hive-policymgr-ssl.xml

ranger-hive-policymgr-ssl.xml

Параметр	Описание	Значение по умолчанию
xasecure.policymgr.clientssl.keystore	Путь к keystore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.credential.file	Путь к файлу с паролями для keystore-файла	/etc/hive/conf/ranger-hive.jceks
xasecure.policymgr.clientssl.truststore.credential.file	Путь к truststore-файлу с учетными данными	/etc/hive/conf/ranger-hive.jceks
xasecure.policymgr.clientssl.truststore	Путь к truststore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.password	Пароль для доступа к keystore-файлу	—
xasecure.policymgr.clientssl.truststore.password	Пароль для доступа к truststore-файлу	—

tez-site.xml

Параметр	Описание	Значение по умолчанию
tez.am.resource.memory.mb	Объем памяти, который YARN выделит Tez Application Master (в мегабайтах). Размер увеличивается с размером DAG	1024
tez.history.logging.service.class	Позволяет Tez использовать Timeline Server для ведения журнала истории	org.apache.tez.dag.history.logging.ats.ATSHistoryLoggingService
tez.lib.uris	Пути HDFS, где хранятся JAR-файлы Tez	${fs.defaultFS}/apps/tez/tez-0.10.3.tar.gz
tez.task.resource.memory.mb	Объем памяти, используемый запущенными задачами в контейнерах TEZ. Обычно это значение задается в DAG	1024
tez.tez-ui.history-url.base	URL для доступа к Tez UI	—
tez.use.cluster.hadoop-libs	Указывает, будет ли Tez использовать кластерные библиотеки Hadoop	true

nginx.conf

Параметр	Описание	Значение по умолчанию
ssl_certificate	Путь к SSL-сертификату для Nginx	/etc/ssl/certs/host_cert.cert

ssl_certificate_key	Путь к ключу SSL-сертификата для Nginx	/etc/ssl/host_cert.key
ssl_protocols	Список допустимых SSL-протоколов для создания SSL-соединения	TLSv1.2
nginx_http_port	HTTP-порт Nginx	8089
nginx_https_port	HTTPS-порт Nginx	9999

Другие параметры

Параметр	Описание	Значение по умолчанию
ACID Transactions	Определяет, использовать ли ACID-транзакции	false
Database type	Тип внешней базы данных, используемой для Hive Metastore	postgres
Custom hive-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hive-site.xml*	—
Custom hive-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hive-env.sh*	—
Ranger plugin enabled	Определяет, активен ли Ranger-плагин	false
Custom ranger-hive-audit.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hive-audit.xml*	—
Custom ranger-hive-security.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hive-security.xml*	—
Custom ranger-hive-policymgr-ssl.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hive-policymgr-ssl.xml*	—
Custom tez-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *tez-site.xml*	—
beeline-hs2-connection.xml	XML-шаблон для генерации name-value пар из объекта `hive_beeline_connection_conf`	beeline-hs2-connection.xml

Компонент Hive HiveServer2

Monitoring

Параметр	Описание	Значение по умолчанию
Java agent path	Путь к Java-агенту JMX Prometheus	/usr/lib/adh-utils/jmx/jmx_prometheus_javaagent.jar
Prometheus metrics port	Порт, на котором будут доступны метрики Hive HiveServer2 в формате Prometheus	9208

Mapping config path	Путь к файлу конфигурации мэппинга метрик	/etc/hive/conf/jmx_hive_server_metric_config.yml
Mapping config	Файл конфигурации мэппинга метрик	hive-mapping-config.yml

Компонент Hive Metastore

Monitoring

Параметр	Описание	Значение по умолчанию
Java agent path	Путь к Java-агенту JMX Prometheus	/usr/lib/adh-utils/jmx/jmx_prometheus_javaagent.jar
Prometheus metrics port	Порт, на котором будут доступны метрики Hive Metastore в формате Prometheus	9207
Mapping config path	Путь к файлу конфигурации мэппинга метрик	/etc/hive/conf/jmx_hive_server_metric_config.yml
Mapping config	Файл конфигурации мэппинга метрик	hive-mapping-config.yml

HUE

Компонент HUE Server

Синтаксис hue.ini

Конфигурационный файл *hue.ini*, отображаемый в веб-интерфейсе ADCM, имеет синтаксис, отличный от синтаксиса исходного файла. В исходном файле уровень вложенности определяется при помощи размещения имен секций в соответствующем количестве квадратных скобок. Пример:

```
[notebook]
show_notebooks=true
[[interpreters]]
[[[mysql]]]
name = MySQL
interface=sqlalchemy
options='{"url": "mysql://root:secret@database:3306/hue"}'
[[[hive]]]
name=Hive
interface=hiveserver2
```

В ADCM уровень вложенности определяется при помощи разделения имен секций точками. Структура из примера выше будет выглядеть следующим образом:

```
notebook.show_notebooks: true
notebook.interpreters.mysql.name: MySQL
notebook.interpreters.mysql.interface: sqlalchemy
notebook.interpreters.mysql.options: '{"url": "mysql://root:secret@database:3306/hue"}'
notebook.interpreters.hive.name: Hive
notebook.interpreters.hive.interface: hiveserver2
```

hue.ini

Параметр	Описание	Значение по умолчанию
desktop.enable_prometheus	Определяет, включен ли сбор метрик Prometheus	false

desktop.http_host	IP-адрес, который прослушивает HUE Server	0.0.0.0
desktop.http_port	Порт, который прослушивает HUE Server	8000
desktop.use_cherrypy_server	Определяет используемый веб-сервер: CherryPy (`true`) или Gunicorn (`false`)	false
desktop.gunicorn_work_class	Тип рабочего процесса для Gunicorn: `gevent`, `eventlet`, `gthread` или `sync`	gthread
desktop.secret_key	Случайная строка, используемая для безопасного хеширования в сессионном хранилище	jFE93j;2[290-eiw.KEiwN2s3['d;/.q[eIW^y#e=+Iei*@Mn<qW5o
desktop.enable_xff_for_hive_impala	Определяет, используется ли заголовок `X-Forwarded-For`, если Hive или Impala его требуют	false
desktop.enable_x_csrf_token_for_hive_impala	Определяет, используется ли заголовок `X-CSRF-Token`, если Hive или Impala его требуют	false
desktop.app_blacklist	Разделенный запятыми список приложений, которые не требуется загружать при запуске сервера	security,pig,sqoop,oozie,hbase,search
desktop.auth.backend	Список комбинаций бэкендов аутентификации в порядке приоритета, разделенных запятыми	desktop.auth.backend.AllowFirstUserDjangoBackend
desktop.database.host	Сетевой или IP-адрес БД HUE Server	{{ groups['adpg.adpg'][0] | d(omit) }}
desktop.database.port	Сетевой порт БД HUE Server	5432
desktop.database.engine	Движок, использующий БД HUE Server	postgresql_psycopg2
desktop.database.user	Имя администраторской учетной записи для БД HUE Server	hue
desktop.database.name	Имя БД HUE Server	hue
desktop.database.password	Пароль учетной записи, заданной в параметре `desktop.database.user`	—
desktop.auth_username	Имя пользователя для аутентификации в веб-интерфейсе HUE	—
desktop.auth_password	Пароль для аутентификации в веб-интерфейсе HUE	—

Interpreter Impala

Параметр	Описание	Значение по умолчанию
notebook.interpreters.impala.name	Имя интерпретатора Impala	impala
notebook.interpreters.impala.interface	Интерфейс интерпретатора Impala	hiveserver2
impala.server_host	Хост Impala Server (одного из компонентов Impala Daemon)	—
impala.server_port	Порт Impala Server	21050
impala.impersonation_enabled	Определяет, имперсонировать ли пользователей при работе с Impala	true
impala.impala_conf_dir	Путь к каталогу конфигурации Impala, содержащему файл *impalad_flags*	/etc/hue/conf
impala.ssl.cacerts	Путь к сертификатам УЦ	/etc/pki/tls/certs/ca-bundle.crt
impala.ssl.validate	Определяет, следует ли HUE валидировать сертификаты,	false

	полученные от сервера	
impala.ssl.enabled	Включает SSL-коммуникацию для этого сервера	false
impala.impala_principal	Принципал Kerberos для Impala	—
impala.auth_username	Имя пользователя для аутентификации в Impala	—
impala.auth_password	Пароль для аутентификации в Impala	—

Interpreter HDFS

Параметр	Описание	Значение по умолчанию
hadoop.hdfs_clusters.default.webhdfs_url	WebHDFS- или HttpFS-эндпойнт ссылка для доступа к данным HDFS	—
hadoop.hdfs_clusters.default.hadoop_conf_dir	Путь к каталогу конфигурационных файлов Hadoop	/etc/hadoop/conf
hadoop.hdfs_clusters.default.security_enabled	Определяет, используется ли Kerberos для защиты кластера Hyperwave	false
hadoop.hdfs_clusters.default.ssl_cert_ca_verify	Определяет, верифицировать ли SSL-сертификаты удостоверяющим центром	false

Interpreter Hive

Параметр	Описание	Значение по умолчанию
notebook.interpreters.hive.name	Имя интерпретатора Hive	hive
notebook.interpreters.hive.interface	Интерфейс интерпретатора Hive	hiveserver2
beeswax.hive_discovery_hs2	Определяет, используется ли обнаружение сервисов для HiveServer2	true
beeswax.hive_conf_dir	Путь к каталогу конфигурации Hive, содержащему файл *hive-site.xml*	/etc/hive/conf
beeswax.use_sasl	Определяет, использовать ли фреймворк SASL для установки соединения с хостом	true
beeswax.hive_discovery_hiveserver2_znode	Адрес znode HiveServer2, если Hive использует режим обнаружения сервисов ZooKeeper	hive.server2.zookeeper.namespace
libzookeeper.ensemble	Список хостов и портов участников ансамбля ZooKeeper	host1:2181,host2:2181,host3:2181
libzookeeper.principal_name	Принципал Kerberos для ZooKeeper	—
beeswax.auth_username	Имя пользователя для аутентификации в Hive	—
beeswax.auth_password	Пароль для аутентификации в Hive	—

Interpreter YARN

Параметр	Описание	Значение по умолчанию
hadoop.yarn_clusters.default.resourcemanager_host	Сетевой адрес хоста, на котором работает Resource Manager	—
hadoop.yarn_clusters.default.resourcemanager_p	Порт, который прослушивает Resource Manager IPC	8031

ort		
hadoop.yarn_clusters.default.submit_to	Определяет, что задачи отправляются на выполнение этим кластером	true
hadoop.yarn_clusters.default.logical_name	Логическое имя Resource Manager (необходимо для режима High Availability)	—
hadoop.yarn_clusters.default.security_enabled	Определяет, используется ли Kerberos для защиты кластера YARN	false
hadoop.yarn_clusters.default.ssl_cert_ca_verify	Определяет, верифицировать ли SSL-сертификаты от YARN Rest API удостоверяющим центром при использовании безопасного режима (HTTPS)	false
hadoop.yarn_clusters.default.resourcemanager_api_url	URL Resource Manager API	—
hadoop.yarn_clusters.default.proxy_api_url	URL первого Resource Manager API	—
hadoop.yarn_clusters.default.history_server_api_url	URL History Server API	—
hadoop.yarn_clusters.default.spark_history_server_url	URL Spark History Server	—
hadoop.yarn_clusters.default.spark_history_server_security_enabled	Определяет, используется ли Kerberos для защиты Spark History Server	false
hadoop.yarn_clusters.ha.resourcemanager_host	Сетевой адрес хоста, на котором работает Resource Manager (режим High Availability)	—
hadoop.yarn_clusters.ha.resourcemanager_port	Порт, который прослушивает Resource Manager IPC (режим High Availability)	—
hadoop.yarn_clusters.ha.logical_name	Логическое имя Resource Manager (необходимо для режима High Availability)	—
hadoop.yarn_clusters.ha.security_enabled	Определяет, используется ли Kerberos для защиты кластера YARN (режим High Availability)	false
hadoop.yarn_clusters.ha.submit_to	Определяет, что задачи отправляются на выполнение этим кластером (режим High Availability)	true
hadoop.yarn_clusters.ha.ssl_cert_ca_verify	Определяет, верифицировать ли SSL-сертификаты от YARN Rest API удостоверяющим центром при использовании безопасного режима (HTTPS) (режим High Availability)	false
hadoop.yarn_clusters.ha.resourcemanager_api_url	URL Resource Manager API (режим High Availability)	—
hadoop.yarn_clusters.ha.history_server_api_url	URL History Server API (режим High Availability)	—
hadoop.yarn_clusters.ha.spark_history_server_url	URL Spark History Server (режим High Availability)	—
hadoop.yarn_clusters.ha.spark_history_server_security_enabled	Определяет, используется ли Kerberos для защиты Spark History Server (режим High Availability)	false

Interpreter Kyuubi

Параметр	Описание	Значение по умолчанию

notebook.dbproxy_extra_classpath	Путь к классам, добавляемый к classpath DBProxy server по умолчанию	/usr/share/java/kyuubi-hive-jdbc.jar
notebook.interpreters.kyuubi.name	Имя интерпретатора Kyuubi	Kyuubi[Spark3]
notebook.interpreters.kyuubi.options	Специальные параметры подключения к серверу Kyuubi	—
notebook.interpreters.kyuubi.interface	Интерфейс сервиса Kyuubi	jdbc

Interpreter Trino

Параметр	Описание	Значение по умолчанию
notebook.interpreters.trino.name	Имя интерпретатора Trino	Trino
notebook.interpreters.trino.interface	Интерфейс сервиса Trino	trino
notebook.interpreters.trino.options	Специальные параметры подключения к серверу Trino	

```
{
    "url": "",
    "auth_type": "basic",
    "kerberos_principal": "",
    "kerberos_service_name": "HTTP",
    "kerberos_force_preemptive": true,
    "kerberos_delegate": true,
    "ssl_cert_ca_verify": false
}
```

Interpreter Ozone

Параметр	Описание	Значение по умолчанию
desktop.ozone.default.webhdfs_url	WebHDFS- или HttpFS-эндпойнт ссылка для доступа к данным HDFS	—
desktop.ozone.default.ozone_conf_dir	Путь к директории с конфигурацией Ozone	/etc/ozone/conf
desktop.ozone.default.security_enabled	Определяет, используется ли Kerberos для защиты кластера Ozone	false
desktop.ozone.default.ssl_cert_ca_verify	Определяет, верифицировать ли SSL-сертификаты удостоверяющим центром	false
desktop.ozone.default.fs_defaultfs	Идентификатор сервиса Ozone	—

hue.ini kerberos config

Параметр	Описание	Значение по умолчанию
desktop.kerberos.hue_keytab	Путь к keytab-файлу Kerberos для HUE	—

desktop.kerberos.hue_principal	Принципал Kerberos для HUE	—
desktop.kerberos.kinit_path	Путь к утилите `kinit`	/usr/bin/kinit
desktop.kerberos.reinit_frequency	Интервал в секундах, через который HUE обновляет свой keytab-файл	3600
desktop.kerberos.ccache_path	Путь к кешу Kerberos, хранящему учетные данные	/tmp/hue_krb5_ccache
desktop.kerberos.krb5_renewlifetime_enabled	Этот параметр должен иметь значение `false`, если параметр `renew_lifetime` в файле *krb5.conf* имеет значение `0m`	false
desktop.auth.auth	Тип аутентификации	—

Authentication on WEB UIs

Параметр	Описание	Значение по умолчанию
desktop.kerberos.kerberos_auth	Определяет, использовать ли Kerberos-аутентификацию для HTTP-клиентов на основе текущего тикета	false
desktop.kerberos.spnego_principal	Принципал Kerberos по умолчанию для HTTP-клиента	—

hue.ini SSL config

Параметр	Описание	Значение по умолчанию
desktop.ssl_certificate	Путь к файлу SSL-сертификата	/etc/ssl/certs/host_cert.cert
desktop.ssl_private_key	Путь к файлу закрытого ключа SSL RSA	/etc/ssl/host_cert.key
desktop.ssl_password	Пароль SSL-сертификата	—
desktop.ssl_no_renegotiation	Отключает всё повторное согласование для TLS версии 1.2 и более ранних	true
desktop.ssl_validate	Определяет, следует ли HUE валидировать сертификаты, полученные от сервера	false
desktop.ssl_cacerts	Этот параметр должен иметь значение `false`, если параметр `renew_lifetime` в файле *krb5.conf* имеет значение `0m`	/etc/pki/tls/certs/ca-bundle.crt
desktop.session.secure	Определяет, будут ли cookie, содержащие идентификатор сессии пользователя и csrf, использовать флаг `secure`	true
desktop.session.http_only	Определяет, будут ли cookie, содержащие идентификатор сессии пользователя и csrf, использовать флаг `HTTP only`	false

LDAP security

Параметр	Описание	Значение по умолчанию
desktop.ldap.ldap_url	URL LDAP-сервера	—
desktop.ldap.base_dn	База поиска для пользователей и групп	"DC=mycompany,DC=com"
desktop.ldap.nt_domain	NT-домен, используемый для LDAP-аутентификации	mycompany.com
desktop.ldap.ldap_cert	Файлы сертификатов в PEM-формате для УЦ, которые будут доверенными для HUE при TLS-аутентификации	—

desktop.ldap.use_start_tls	Установите значение `true`, если вы не используете Secure LDAP (LDAPS), но хотите использовать защищенные подключения с помощью TLS	true
desktop.ldap.bind_dn	Уникальное имя (DN) bind-пользователя	"CN=ServiceAccount,DC=mycompany,DC=com"
desktop.ldap.bind_password	Пароль bind-пользователя	—
desktop.ldap.ldap_username_pattern	Шаблон для поиска имени пользователя. Заполните поле `<username>` в этом параметре	"uid=<username>,ou=People,dc=mycompany,dc=com"
desktop.ldap.create_users_on_login	Определяет, создавать ли пользователей в HUE при попытке входа под учетными данными LDAP	true
desktop.ldap.sync_groups_on_login	Определяет, синхронизировать ли группы пользователей при их входе	true
desktop.ldap.login_groups	Список групп LDAP, разделенных запятыми, содержащих пользователей, которым разрешен вход	—
desktop.ldap.ignore_username_case	Определяет, игнорировать ли регистр учетных записей пользователей при поиске существующих пользователей	true
desktop.ldap.force_username_lowercase	Определяет, следует ли принудительно использовать нижний регистр для учетных записей пользователей при их создании из LDAP	true
desktop.ldap.force_username_uppercase	Определяет, следует ли принудительно использовать верхний регистр для учетных записей пользователей при их создании из LDAP. Этот параметр нельзя включать одновременно с параметром `desktop.ldap.force_username_lowercase`	false
desktop.ldap.search_bind_authentication	Включает аутентификацию search bind	true
desktop.ldap.subgroups	Определяет, какую группировку использовать: вложенную или подчиненную (не используется)	nested
desktop.ldap.nested_members_search_depth	Количество уровней поиска для вложенных членов	10
desktop.ldap.follow_referrals	Определяет, следует ли переходить по отсылкам	false
desktop.ldap.users.user_filter	Базовый фильтр для поиска пользователей	"objectclass=*"
desktop.ldap.users.user_name_attr	Атрибут имени пользователя в схеме LDAP	sAMAccountName
desktop.ldap.groups.group_filter	Базовый фильтр для поиска групп	"objectclass=*"
desktop.ldap.groups.group_name_attr	Атрибут имени группы в схеме LDAP	cn
desktop.ldap.groups.group_member_attr	Атрибут объекта группы, идентифицирующего членов группы	member

Другие параметры

Параметр	Описание	Значение по умолчанию
Enable custom ulimits	Включение этой опции	

	позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	`[Service]`
		`DefaultLimitCPU=`
		`DefaultLimitFSIZE=`
		`DefaultLimitDATA=`
		`DefaultLimitSTACK=`
		`DefaultLimitCORE=`
		`DefaultLimitRSS=`
		`DefaultLimitNOFILE=`
		`DefaultLimitAS=`
		`DefaultLimitNPROC=`
		`DefaultLimitMEMLOCK=`
		`DefaultLimitLOCKS=`
		`DefaultLimitSIGPENDING=`
		`DefaultLimitMSGQUEUE=`
		`DefaultLimitNICE=`
		`DefaultLimitRTPRIO=`
		`DefaultLimitRTTIME=`

Custom hue.ini	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *hue.ini*. Список доступных параметров можно найти в документации HUE	—
log.conf	Файл конфигурации с объявлением различных объектов логирования	log.conf
Custom impalad_flags	Пользовательские значения параметров, которые будут использоваться вместо исходных	—

Ulimit settings

Параметр	**Описание**	**Соответствующая опция команды ulimit в CentOS**
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)

DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Impala

Параметр	Описание	Значение по умолчанию
impala-env.sh	Содержимое файла *impala-env.sh*, определяющего настройки окружения для Impala	impala-env.sh
Custom impala-env.sh	Содержимое кастомного файла *impala-env.sh*, определяющего кастомные настройки окружения для Impala	custom-impala-env.sh

Credential encryption

Параметр	Описание	Значение по умолчанию
Encryption enable	Определяет, включено ли шифрование учетных данных	false
Credential provider path	Путь к провайдеру учетных данных для создания файлов *.jceks*, содержащих закрытые ключи	jceks://hdfs/apps/impala/security/impala.jceks
Ranger plugin credential provider path	Путь к провайдеру учетных данных плагина Ranger	jceks://file/etc/impala/conf/ranger-impala.jceks
Custom jceks	Определяет, используются ли кастомные файлы *.jceks*, расположенные по пути провайдера учетных данных (`true`), или автосгенерированные (`false`)	false
Password file name	Имя файла паролей в classpath сервиса, если использование файла паролей выбрано в опциях credstore	impala_credstore_pass

ranger-hive-audit.xml

Параметр	Описание	Значение по умолчанию
xasecure.audit.destination.solr.batch.filespool.dir	Путь к Spool-каталогу	/srv/ranger/hdfs_plugin/audit_solr_spool
xasecure.audit.destination.solr.urls	URL сервера Solr для сохранения аудит-событий. Оставьте значение пустым или установите значение `NONE` при использовании ZooKeeper для подключения к Solr	—

xasecure.audit.destination.solr.zookeepers	Указывает строку подключения ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.force.use.inmemory.jaas.config	Использует in-memory JAAS-файл для соединения с Solr	—
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.jaas.Client.loginModuleControlFlag	Определяет требование к успешному завершению работы модуля. Значения могут быть `required`, `requisite`, `sufficient` или `optional`	—
xasecure.audit.jaas.Client.loginModuleName	Имя класса-аутентификатора	—
xasecure.audit.jaas.Client.option.keyTab	Имя keytab-файла, из которого необходимо получить секретный ключ принципала	—
xasecure.audit.jaas.Client.option.principal	Имя используемого принципала	—
xasecure.audit.jaas.Client.option.serviceName	Имя пользователя или сервиса, для которого выполнятся логин	—
xasecure.audit.jaas.Client.option.storeKey	Установите значение `true`, чтобы keytab-файл или ключ принципала сохранялись в учетных данных субъекта	false
xasecure.audit.jaas.Client.option.useKeyTab	Установите значение `true`, чтобы модуль получал ключ принципала из keytab-файла	false

ranger-hive-security.xml

Параметр	**Описание**	**Значение по умолчанию**
ranger.plugin.hive.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.hive.service.name	Имя сервиса Ranger, хранящего политики для данного инстанса Impala	—
ranger.plugin.hive.policy.cache.dir	Директория, в которой хранятся политики Ranger после успешного извлечения из источника	/srv/ranger/impala/policycache
ranger.plugin.hive.policy.pollIntervalMs	Период проверки политик на актуальность в миллисекундах	30000
ranger.plugin.hive.policy.rest.client.connection.timeoutMs	Тайм-аут соединения клиента Ranger REST в миллисекундах	120000
ranger.plugin.hive.policy.rest.client.read.timeoutMs	Тайм-аут чтения клиента Ranger REST в миллисекундах	30000
xasecure.hive.update.xapolicies.on.grant.revoke	Следует ли плагину Impala обновлять политики Ranger после изменения прав при помощи `GRANT` / `REVOKE`	true
ranger.plugin.hive.policy.rest.ssl.config.file	Путь к конфигурационному файлу RangerRestClient SSL плагина Hive	/etc/hbase/conf/ranger-hbase-policymgr-ssl.xml

ranger-hive-policymgr-ssl.xml

Параметр	Описание	Значение по умолчанию
xasecure.policymgr.clientssl.keystore	Путь к keystore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.credential.file	Путь к файлу с паролями для keystore-файла	/etc/impala/conf/ranger-impala.jceks
xasecure.policymgr.clientssl.truststore.credential.file	Путь к truststore-файлу с учетными данными	/etc/impala/conf/ranger-impala.jceks
xasecure.policymgr.clientssl.truststore	Путь к truststore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.password	Пароль для доступа к keystore-файлу	—
xasecure.policymgr.clientssl.truststore.password	Пароль для доступа к truststore-файлу	—

Enable LDAP

Параметр	Описание	Значение по умолчанию
ldap_uri	URI сервера LDAP. Обычно начинается с `ldap://` или `ldaps://` в случае использования SSL. Опционально в URI можно указать порт, например: `ldap://ldap_server.example.com:389`	—
ldap_domain	Заменяет имя пользователя строкой `<username>@ldap_domain`, где `<username>` — имя пользователя, проходящего аутентификацию. Нельзя использовать вместе с `ldap_bind_pattern` или `ldap_baseDN`	—
ldap_bind_dn	Отличительное имя пользователя для привязки во время поиска по пользователям/группам. Требуется, только если используются фильтры пользователей/групп, а LDAP-сервер не разрешает анонимный поиск	—
ldap_bind_password	Пароль пользователя для привязки. Требуется, если на сервере не активирована анонимная привязка	—
ldap_bind_password_cmd	Команда Unix, вывод которой возвращает пароль для использования с ключом `--ldap_bind_dn`. Вывод команды обрезается до 1024 байт, конечные пробелы отбрасываются	cat /etc/impala/conf/pass.pwd
ldap_user_search_basedn	Базовый DN для поиска по пользователям	—
ldap_group_search_basedn	Базовый DN для поиска по группам	—
ldap_baseDN	База поиска. Заменяет имя пользователя на DN следующего вида: `uid=<userid>,ldap_baseDN`, где `<userid>` — имя пользователя, проходящего аутентификацию. Нельзя использовать вместе с `ldap_bind_pattern` или `ldap_domain`	—
ldap_user_filter	Фильтр для простых и поисковых привязок. В случае простой привязки — список значений, разделенных запятой. Если параметр указан, то для успешной аутентификации пользователь должен входить в этот список. В случае поисковой привязки — LDAP-фильтр, использующийся для поиска в LDAP. Может содержать шаблон `{0}`, который будет заменен на имя пользователя	—
ldap_group_filter	Список групп, разделенных запятой. Если параметр указан, то для успешной аутентификации пользователь должен состоять в одной из перечисленных групп	—

ldap_allow_anonymous_binds	При значении `true` Impala разрешает LDAP-аутентификацию с пустым паролем (анонимная привязка)	false
ldap_search_bind_authentication	Позволяет переключаться между простой и поисковой привязкой для аутентификации	true
ldap_ca_certificate	Указывает местоположение сертификата в стандартном формате PEM для SSL. Данный сертификат должен храниться локально в директории, которую могут читать только `impala` и остальные доверенные пользователи	—
ldap_passwords_in_clear_ok	Позволяет веб-серверу запускаться с LDAP-аутентификацией даже при выключенном SSL. При установке значения `true` значение параметра `auth_creds_ok_in_clear` в файле *impalarc* также становится `true`. Учтите, что это потенциально небезопасный вариант работы	false
ldap_bind_pattern	Строка, в которой `#UID` заменяется на имя пользователя. Например, при значении `user=#UID,OU=foo,CN=bar` при попытке пользователя `henry` пройти аутентификацию его имя заменится на `user=henry,OU=foo,CN=bar`. Нельзя использовать вместе с `ldap_baseDN` или `ldap_domain`	—
allow_custom_ldap_filters_with_kerberos_auth	Позволяет использовать кастомные LDAP-фильтры для пользователей и групп даже при включенном Kerberos	true

Другие параметры

Параметр	Описание	Значение по умолчанию
Ranger plugin enabled	Определяет, активен ли Ranger-плагин	false
Custom ranger-hive-audit.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hive-audit.xml*	—
Custom ranger-hive-security.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hive-security.xml*	—
Custom ranger-hive-policymgr-ssl.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-hive-policymgr-ssl.xml*	—

Компонент Impala Catalog Service

catalogstore.conf

Параметр	Описание	Значение по умолчанию
hostname	Имя хоста, на котором установлен Catalog Service. Если Kerberos включен, оно также является частью принципала Kerberos. Если этот параметр не установлен, используется системное значение по умолчанию	—
state_store_host	Хост, на котором запущен компонент Impala Statestore	—
state_store_port	Порт, на котором запущен компонент Impala Statestore	24000
catalog_service_host	Хост, на котором запущен компонент Impala Catalog Service	—

catalog_service_port	Порт, на котором запущен компонент Impala Catalog Service	26000
enable_webserver	Включает или отключает веб-сервер демона Catalog Service. Его веб-интерфейс включает информацию о базах данных, таблицах и других объектах, управляемых Impala, в дополнение к информации об использовании ресурсов и параметрам конфигурации Catalog Service	True
webserver_require_spnego	Включает аутентификацию Kerberos для веб-консолей Hadoop HTTP для всех ролей этого сервиса с использованием протокола SPNEGO. Используйте этот параметр, только если для сервиса HDFS включен Kerberos	False
webserver_port	Порт, на котором работает веб-сервер Catalog Service	25020
log_dir	Каталог, в котором демон Catalog Service хранит свои файлы журналов	/var/log/impala/catalogd/
log_filename	Префикс имени файла журнала. Полный путь к файлу — `<log_dir>/<log_filename>`	catalogd
max_log_files	Количество файлов журнала, которые должны храниться для каждого уровня важности (`INFO`, `WARNING`, `ERROR` и `FATAL`) прежде чем старые файлы журнала будут удалены. Число должно быть больше 1, чтобы по крайней мере текущий файл журнала оставался открытым. Если установлено значение `0`, все файлы журналов сохраняются, и ротация журналов не производится	10
minidump_path	Каталог для хранения Breakpad-дампов демона Catalog Service	/var/log/impala-minidumps
max_minidumps	Максимальное количество файлов Breakpad-дампов, хранимых демоном Catalog Service. Отрицательное значение или `0` интерпретируется как неограниченное число	9
hms_event_polling_interval_s	Если для этого параметра задано положительное число, Catalog Service получает новые уведомления из Hive Metastore с указанным интервалом в секундах. Если для `hms_event_polling_interval_s` установлено значение `0`, автоматическое признание метаданных недействительными и обновление отключены. См. Metadata management	2
load_auth_to_local_rules	Если активен (True) и Kerberos включен для Impala, Impala использует настройку `auth_to_local`, заданную в конфигурации HDFS `hadoop.security.auth_to_local`	True
load_catalog_in_background	Если установлено значение `True`, метаданные загружаются в фоновом режиме, даже если эти метаданные не требуются ни для одного запроса. Если `False`, метаданные загружаются при первом обращении к ним	False
catalog_topic_mode	Уровень детализация метаданных, передающихся по запросу между координатором демона Impala и Impala Catalog Service. См. Metadata management	minimal
statestore_subscriber_timeout_seconds	Тайм-аут в секундах для подключения Impala Daemon и Catalog Server к Statestore	30
state_store_subscriber_port	Порт, на котором запущен StateStoreSubscriberService. StateStoreSubscriberService прослушивает этот порт на наличие обновлений от демона Statestore	23020
enable_statestored_ha	Показывает, присутствуют ли в кластере два компонента Impala Statestore, что позволяет им работать в режиме высокой доступности	false
state_store_2_host	Хост, на котором запущен второй компонент Impala Statestore (режим высокой доступности)	—
state_store_2_port	Порт, на котором запущен второй компонент Impala Statestore (режим высокой доступности)	24000

enable_catalogd_ha	Показывает, присутствуют ли в кластере два компонента Impala Catalog Service, что позволяет им работать в режиме высокой доступности	false
kerberos_reinit_interval	Количество минут между переустановкой тикета на сервере Kerberos	60
principal	Принципал Kerberos сервиса Impala	—
keytab_file	Kerberos keytab-файл сервиса Impala	—
ssl_server_certificate	Путь к файлу TLS/SSL с ключом сертификата сервера, используемым для TLS/SSL. Он используется, когда Impala работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
ssl_private_key	Путь к файлу TLS/SSL с закрытым ключом, используемым для TLS/SSL. Он используется, когда Impala работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
ssl_client_ca_certificate	Путь к сертификату в формате PEM, используемому для подтверждения подлинности серверов SSL/TLS, к которым могут подключаться демоны Impala. Поскольку демоны Impala также подключаются друг к другу, он должен включать сертификат CA, используемый для подписи всех сертификатов SSL/TLS. SSL/TLS между демонами Impala не может быть включен без этого параметра	—
webserver_certificate_file	Путь к файлу TLS/SSL с ключом сертификата сервера, используемым для TLS/SSL. Он необходим, когда веб-сервер демона Catalog Service работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
webserver_private_key_file	Путь к файлу TLS/SSL с закрытым ключом, используемым для TLS/SSL. Он необходим, когда веб-сервер демона Catalog Service работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
ssl_minimum_version	Минимальная версия TLS	TLSv1.2

Прочее

Параметр	Описание	Значение по умолчанию
Custom catalogstore.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *catalogstore.conf*	—
Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	

Enable custom ulimits

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit

		в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Impala Client

Параметр	Описание	Значение по умолчанию
impala-shell-env.sh	Содержимое файла *impala-shell-env.sh*, устанавливающего необходимые переменные среды	impala-shell-env.sh
impalarc	Содержимое файла *impalarc*, хранящего LDAP-настройки для impala-shell	impalarc

Компонент Impala Daemon

impalastore.conf

Параметр	Описание	Значение по умолчанию
hostname	Имя хоста, которое используется для Impala Daemon. Если Kerberos включен, оно также является частью принципала Kerberos. Если этот	—

	параметр не установлен, используется системное значение по умолчанию	
beeswax_port	Порт, на котором демоны Impala обслуживают клиентские запросы Beeswax	21000
fe_port	Порт внешнего интерфейса (frontend) демона Impala	21000
be_port	Только для внутреннего использования. Демоны Impala используют этот порт для связи друг с другом на основе Thrift	22000
krpc_port	Только для внутреннего использования. Демоны Impala используют этот порт для связи друг с другом на основе KRPC	27000
hs2_port	Порт, на котором демоны Impala обслуживают клиентские запросы HiveServer2	21050
hs2_http_port	Порт, который используют клиентские приложения для передачи команд и получения результатов по HTTP по протоколу HiveServer2	28000
enable_webserver	Включает или отключает веб-сервер демона Impala. Его веб-интерфейс содержит информацию о параметрах конфигурации, запущенных и завершенных запросах и связанном с ними использовании ресурсов. В основном используется для диагностики проблем с запросами, которые можно отследить до определенной ноды	True
webserver_require_spnego	Включает аутентификацию Kerberos для веб-консолей Hadoop HTTP для всех ролей этого сервиса с использованием протокола SPNEGO. Используйте этот параметр, только если для сервиса HDFS включен Kerberos	False
webserver_port	Порт, на котором работает веб-сервер демона Impala	25000
catalog_service_host	Хост, на котором запущен компонент Impala Catalog Service	—
catalog_service_port	Порт, на котором запущен компонент Impala Catalog Service	26000
state_store_host	Хост, на котором запущен компонент Impala Statestore	—
state_store_port	Порт, на котором запущен компонент Impala Statestore	24000
state_store_subscriber_port	Порт, на котором запущен StateStoreSubscriberService. StateStoreSubscriberService прослушивает этот порт на наличие обновлений от демона Statestore	23030
scratch_dirs	Каталог, в который демоны Impala записывают данные для освобождения памяти во время больших операций сортировки, объединения, агрегирования и т.д. Файлы удаляются после завершения операции. Потенциально это могут быть большие объемы данных	/srv/impala/
log_dir	Каталог, в котором демоны Impala хранят свои файлы журналов	/var/log/impala/impalad/
profile_log_dir	Каталог, в который записываются файлы журналов профиля	/var/log/impala/impalad/profiles
log_filename	Префикс имени файла журнала. Полный путь к файлу — `<log_dir>/<log_filename>`	impalad
max_log_files	Количество файлов журнала, которые должны храниться для каждого уровня важности (`INFO`, `WARNING`, `ERROR` и `FATAL`) прежде чем старые файлы журнала будут удалены. Число должно быть больше 1, чтобы по крайней мере текущий файл журнала оставался открытым. Если установлено значение `0`, все файлы журналов сохраняются, и ротация журналов не производится	10
audit_event_log_dir	Каталог, в который записываются файлы журнала событий аудита демона Impala, если включено свойство `Impala Audit Event Generation`	/var/log/impala/impalad/audit

minidump_path	Каталог для хранения Breakpad-дампов демона Impala	/var/log/impala-minidumps
lineage_event_log_dir	Каталог, в который записываются лог-файлы lineage, если включено свойство `Impala Lineage Generation`	/var/log/impala/impalad/lineage
local_library_dir	Локальный каталог, в который Демон Impala копирует библиотеки пользовательских функций (user-defined function, UDF) из HDFS	/usr/lib/impala/udfs
max_lineage_log_file_size	Максимальный размер (в записях) лог-файла lineage демона Impala. При превышении размера создается новый файл	5000
max_audit_event_log_file_size	Максимальный размер (в запросах) файла журнала событий аудита демона Impala. При превышении размера создается новый файл	5000
fe_service_threads	Максимально допустимое количество одновременных клиентских подключений. Параметр определяет, сколько запросов может выполняться одновременно. Когда большее число клиентов пытается подключиться к Impala, более поздние клиенты должны ждать, пока предыдущие клиенты отключатся. Установка слишком высокого значения `fe_service_threads` может негативно повлиять на задержку выполнения запросов	64
mem_limit	Ограничение памяти (в байтах) для демона Impala, обеспечиваемое самим демоном. Это ограничение не включает память, потребляемую встроенной JVM демона. Impala использует этот объем памяти для обработки запросов, кешированных данных, сетевых буферов, фоновых операций и т.д. Если предел превышен, запросы будут уничтожаться до тех пор, пока используемая память не станет меньше предела	1473249280
idle_query_timeout	Время в секундах, по истечении которого отменяется бездействующий запрос (обработка не выполняется, и от клиента не поступают обновления). Если установлено значение `0`, бездействующие запросы никогда не отменяются	0
idle_session_timeout	Время в секундах, по истечении которого Impala закрывает бездействующую сессию и отменяет все запущенные запросы. Если установлено значение `0`, бездействующие сессии никогда не закрываются	0
max_result_cache_size	Максимальное количество результатов запроса, которое клиент может запросить для кеширования каждого запроса для поддержки перезапуска выборки. Этот параметр защищает от неоправданно больших кешей результатов. Запросы, превышающие указанный максимум, отклоняются	100000
max_cached_file_handles	Максимальное количество дескрипторов кешированных файлов HDFS. Кеширование файловых дескрипторов HDFS уменьшает количество открываемых новых файловых дескрипторов и, таким образом, снижает нагрузку на HDFS NameNode. Каждый кешированный дескриптор файла потребляет небольшой объем памяти. Если установлено значение `0`, кеширование дескрипторов файлов отключено	20000
unused_file_handle_timeout_sec	Максимальное время в секундах, в течение которого неиспользуемый дескриптор файла HDFS остается в кеше дескрипторов файлов HDFS. Когда базовый файл для кешированного дескриптора файла удаляется, дисковое пространство не может быть освобождено до тех пор, пока кешированный дескриптор файла не будет удален из кеша. Этот тайм-аут позволяет освободить место на диске, занятое удаленными файлами, за предсказуемый период времени. Если установлено значение `0`, неиспользуемые дескрипторы кешированных файлов HDFS не удаляются	21600
statestore_subscriber_timeout_seconds	Тайм-аут в секундах для подключения Impala Daemon и Catalog Server к Statestore	30
default_query_options	Список пар ключ/значение, представляющих собой дополнительные параметры запроса для передачи в командную строку демона Impala, разделенные запятыми	default_file_format=parquet,default_transactional_type=none

load_auth_to_local_rules	Если активен (True) и Kerberos включен для Impala, Impala использует настройку `auth_to_local`, заданную в конфигурации HDFS `hadoop.security.auth_to_local`	True
catalog_topic_mode	Уровень детализация метаданных, передающихся по запросу между координатором демона Impala и Impala Catalog Service. См. Metadata management	minimal
use_local_catalog	Позволяет координаторам кешировать метаданные, полученные от Impala Catalog Service. Если для этого параметра установлено значение `True`, координаторы получают метаданные по мере необходимости от catalogd и кешируют их локально. Кешированные метаданные автоматически удаляются при нехватке памяти или по истечении срока актуальности. См. Metadata management	True
abort_on_failed_audit_event	Определяет, следует ли выключать Impala, если возникает проблема с записью события аудита	False
max_minidumps	Максимальное количество файлов Breakpad-дампов, хранимых демоном Impala. Отрицательное значение или `0` интерпретируется как неограниченное число	9
authorized_proxy_user_config	Указывает набор авторизованных прокси-пользователей (пользователей, которые могут олицетворять других пользователей во время авторизации) и пользователей, которых им разрешено олицетворять. Пример синтаксиса опции: `authenticated_user1=delegated_user1,delegated_user2;authenticated_user2=*.` См. Configuring Impala delegation for clients. Список может содержать короткие имена пользователей или символ `*` для обозначения всех пользователей	knox=*;zeppelin=*
queue_wait_timeout_ms	Максимальное время (в миллисекундах), в течение которого запрос ожидает принятия до истечения времени ожидания. Значение должно быть положительным целым числом	60000
disk_spill_encryption	Указывает, следует ли шифровать и проверять целостность всех данных, перенесенных на диск как часть запроса	False
abort_on_config_error	Определяет, следует ли прервать запуск Impala, если параметры конфигурации заданы некорректно, или Impala запущена на неподдерживаемом оборудовании	True
llama_site_path	Путь к конфигурационному файлу *llama-site.xml*	/etc/impala/conf/llama-site.xml
fair_scheduler_allocation_path	Путь к конфигурационному файлу *fair-scheduler.xml*	/etc/impala/conf/fair-scheduler.xml
enable_statestored_ha	Показывает, присутствуют ли в кластере два компонента Impala Statestore, что позволяет им работать в режиме высокой доступности	false
state_store_2_host	Хост, на котором запущен второй компонент Impala Statestore (режим высокой доступности)	—
state_store_2_port	Порт, на котором запущен второй компонент Impala Statestore (режим высокой доступности)	24000
kerberos_reinit_interval	Количество минут между переустановкой тикета на сервере Kerberos	60
principal	Принципал Kerberos сервиса Impala	—
keytab_file	Kerberos keytab-файл сервиса Impala	—
ssl_server_certificate	Путь к файлу TLS/SSL с ключом сертификата сервера, используемым для TLS/SSL. Он используется, когда Impala работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
ssl_private_key	Путь к файлу TLS/SSL с закрытым ключом, используемым для TLS/SSL. Он используется, когда Impala работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—

ssl_client_ca_certificate	Путь к сертификату в формате PEM, используемому для подтверждения подлинности серверов SSL/TLS, к которым могут подключаться демоны Impala. Поскольку демоны Impala также подключаются друг к другу, он должен включать сертификат CA, используемый для подписи всех сертификатов SSL/TLS. SSL/TLS между демонами Impala не может быть включен без этого параметра	—
webserver_certificate_file	Путь к файлу TLS/SSL с ключом сертификата сервера, используемым для TLS/SSL. Он необходим, когда веб-сервер демона Impala работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
webserver_private_key_file	Путь к файлу TLS/SSL с закрытым ключом, используемым для TLS/SSL. Он необходим, когда веб-сервер демона Impala работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
ssl_minimum_version	Минимальная версия TLS	TLSv1.2

Прочее

Параметр	Описание	Значение по умолчанию
Custom impalastore.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *impalastore.conf*	—
log4j.properties	Настройки утилиты Apache Log4j	

```
log.threshold=INFO
main.logger=FA
impala.root.logger=DEBUG,FA
log4j.rootLogger=DEBUG,FA
log.dir=/var/log/impala/impalad
max.log.file.size=200MB
log4j.appender.FA=org.apache.log4j.FileAppender
log4j.appender.FA.File=/var/log/impalad/impalad.INFO
log4j.appender.FA.layout=org.apache.log4j.PatternLayout
log4j.appender.FA.layout.ConversionPattern=%p%d{MMdd HH:mm:ss.SSS'000'} %t %c] %m%n
log4j.appender.console=org.apache.log4j.ConsoleAppender
log4j.appender.console.target=System.err
log4j.appender.console.layout=org.apache.log4j.PatternLayout
log4j.appender.console.layout.ConversionPattern=%d{yy/MM/dd HH:mm:ss} %p %c{2}: %m%n
```

llama-site.xml	Настройки ресурсных пулов	

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
</configuration>
```

fair_scheduler.xml	Настройки ресурсных пулов	

```
<allocations>

</allocations>
```

Enable custom ulimits	Включение этой опции	

позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Ulimit settings

Параметр	**Описание**	**Соответствующая опция команды ulimit в CentOS**
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Impala Statestore

statestore.conf

Параметр	Описание	Значение по умолчанию
hostname	Имя хоста, на котором установлен Statestore. Если Kerberos включен, оно также является частью принципала Kerberos. Если этот параметр не установлен, используется системное значение по умолчанию	—
state_store_host	Хост, на котором запущен компонент Impala Statestore	—
state_store_port	Порт, на котором запущен компонент Impala Statestore	24000
catalog_service_host	Хост, на котором запущен компонент Impala Catalog Service	—
catalog_service_port	Порт, на котором запущен компонент Impala Catalog Service	26000
enable_webserver	Включает или отключает веб-сервер демона Statestore. Его веб-интерфейс содержит информацию об использовании памяти, параметрах конфигурации и текущих проверках работоспособности, выполняемых Statestore	True
webserver_require_spnego	Включает аутентификацию Kerberos для веб-консолей Hadoop HTTP для всех ролей этого сервиса с использованием протокола SPNEGO. Используйте этот параметр, только если для сервиса HDFS включен Kerberos	False
webserver_port	Порт, на котором работает веб-сервер Statestore	25010
log_dir	Каталог, в котором демон Statestore хранит свои файлы журналов	/var/log/impala/statestored/
log_filename	Префикс имени файла журнала. Полный путь к файлу — `<log_dir>/<log_filename>`	statestored
max_log_files	Количество файлов журнала, которые должны храниться для каждого уровня важности (`INFO`, `WARNING`, `ERROR` и `FATAL`) прежде чем старые файлы журнала будут удалены. Число должно быть больше 1, чтобы по крайней мере текущий файл журнала оставался открытым. Если установлено значение `0`, все файлы журналов сохраняются, и ротация журналов не производится	10
minidump_path	Каталог для хранения Breakpad-дампов демона Statestore	/var/log/impala-minidumps
max_minidumps	Максимальное количество файлов Breakpad-дампов, хранимых демоном Statestore. Отрицательное значение или `0` интерпретируется как неограниченное число	9
state_store_num_server_worker_threads	Количество рабочих потоков для диспетчера потоков сервера Statestore Thrift	4
state_store_pending_task_count_max	Максимальное количество задач, которые могут быть отложены диспетчером потоков сервера Statestore Thrift. Значение `0` разрешает бесконечное количество отложенных задач	0
enable_statestored_ha	Показывает, присутствуют ли в кластере два компонента Impala Statestore, что позволяет им работать в режиме высокой доступности	false
state_store_ha_port	RPC-порт экземпляра Statestore в режиме высокой доступности	24020
kerberos_reinit_interval	Количество минут между переустановкой тикета на сервере Kerberos	60
principal	Принципал Kerberos сервиса Impala	—
keytab_file	Kerberos keytab-файл сервиса Impala	—

ssl_server_certificate	Путь к файлу TLS/SSL с ключом сертификата сервера, используемым для TLS/SSL. Он используется, когда Impala работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
ssl_private_key	Путь к файлу TLS/SSL с закрытым ключом, используемым для TLS/SSL. Он используется, когда Impala работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
ssl_client_ca_certificate	Путь к сертификату в формате PEM, используемому для подтверждения подлинности серверов SSL/TLS, к которым могут подключаться демоны Impala. Поскольку демоны Impala также подключаются друг к другу, он должен включать сертификат CA, используемый для подписи всех сертификатов SSL/TLS. SSL/TLS между демонами Impala не может быть включен без этого параметра	—
webserver_certificate_file	Путь к файлу TLS/SSL с ключом сертификата сервера, используемым для TLS/SSL. Он необходим, когда веб-сервер демона Statestore работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
webserver_private_key_file	Путь к файлу TLS/SSL с закрытым ключом, используемым для TLS/SSL. Он необходим, когда веб-сервер демона Statestore работает как сервер TLS/SSL. Файл сертификата должен быть в формате PEM	—
ssl_minimum_version	Минимальная версия TLS	TLSv1.2

Прочее

Параметр	Описание	Значение по умолчанию
Custom statestore.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *statestore.conf*	—

Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	`[Service]`
		`DefaultLimitCPU=`
		`DefaultLimitFSIZE=`
		`DefaultLimitDATA=`
		`DefaultLimitSTACK=`
		`DefaultLimitCORE=`
		`DefaultLimitRSS=`
		`DefaultLimitNOFILE=`
		`DefaultLimitAS=`
		`DefaultLimitNPROC=`
		`DefaultLimitMEMLOCK=`
		`DefaultLimitLOCKS=`
		`DefaultLimitSIGPENDING=`
		`DefaultLimitMSGQUEUE=`
		`DefaultLimitNICE=`
		`DefaultLimitRTPRIO=`
		`DefaultLimitRTTIME=`

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)

DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Kyuubi

Компонент Kyuubi Server

kyuubi-defaults.conf

Параметр	Описание	Значение по умолчанию
kyuubi.frontend.rest.bind.port	Порт, на котором работает сервис REST	10099
kyuubi.frontend.thrift.binary.bind.port	Порт, на котором работает сервис Thrift через бинарный протокол	10099
kyuubi.frontend.thrift.http.bind.port	Порт, на котором работает сервис Thrift через HTTP	10010
kyuubi.frontend.thrift.http.path	Компонент `path` URL-эндпоинта HTTP-версии Thrift	cliservice
kyuubi.engine.share.level	Уровень разделения движка. Возможные значения: `CONNECTION` (один движок на подключение), `USER` (один движок на пользователя), `GROUP` (один движок на группу), `SERVER` (один движок на сервер)	USER
kyuubi.engine.type	Тип движка, поддерживаемый Kyuubi. Возможные значения: `SPARK_SQL`, `FLINK_SQL`, `TRINO`, `HIVE_SQL`, `JDBC`	SPARK_SQL
kyuubi.operation.language	Язык программирования для интерпретации входных команд. Возможные значения: `SQL`, `SCALA`, `PYTHON`	SQL
kyuubi.frontend.protocols	Список поддерживаемых фронтенд-протоколов, разделенных запятой. Возможные элементы списка: `THRIFT_BINARY`, `THRIFT_HTTP`, `REST`	THRIFT_BINARY

kyuubi.frontend.thrift.binary.ssl.disallowed.protocols	Версии SSL, которые запрещено использовать Thrift при использовании бинарного протокола	SSLv2,SSLv3,TLSv1.1
kyuubi.frontend.thrift.http.ssl.protocol.blacklist	Версии SSL, которые запрещено использовать Thrift при использовании протокола HTTP	SSLv2,SSLv3,TLSv1.1
kyuubi.ha.addresses	Внешние адреса экземпляров Kyuubi	<hostname_1>:2181, …, <hostname_N>:2181
kyuubi.ha.namespace	Корневая директория сервиса для разворачивания URI-экземпляра	kyuubi
kyuubi.metadata.store.jdbc.database.type	Тип базы данных для хранения серверных метаданных. Возможные значения: `SQLITE`, `MYSQL`, `POSTGRESQL`	POSTGRESQL
kyuubi.metadata.store.jdbc.url	JDBC-строка для подключения к хранилищу серверных метаданных	jdbc:postgresql://{{ groups['adpg.adpg'][0] | d(omit) }}:5432/kyuubi
kyuubi.metadata.store.jdbc.driver	Имя класса JDBC-драйвера для хранилища серверных метаданных	org.postgresql.Driver
kyuubi.metadata.store.jdbc.user	Имя пользователя для хранилища серверных метаданных	kyuubi
kyuubi.metadata.store.jdbc.password	Пароль пользователя для хранилища серверных метаданных	—
kyuubi.metrics.enabled	Включает сбор метрик с Kyuubi Server	false
kyuubi.metrics.reporters	Сервис мониторинга для сбора метрик	PROMETHEUS
kyuubi.metrics.prometheus.port	Порт Prometheus	10019
kyuubi.metrics.prometheus.path	Эндпойнт Prometheus для отображения метрик	/metrics
kyuubi.frontend.thrift.binary.ssl.enabled	Указывает, использовать ли SSL-шифрование при использовании Thrift с бинарным протоколом	false
kyuubi.frontend.thrift.http.use.SSL	Указывает, использовать ли SSL-шифрование при использовании Thrift с протоколом HTTP	false
kyuubi.frontend.ssl.keystore.type	Тип хранилища ключей SSL-сертификатов	—
kyuubi.frontend.ssl.keystore.path	Путь к хранилищу ключей SSL-сертификатов	—
kyuubi.frontend.ssl.keystore.password	Пароль от хранилища ключей SSL-сертификатов	—
kyuubi.frontend.thrift.http.ssl.keystore.path	Путь к хранилищу ключей SSL-сертификатов	—
kyuubi.frontend.thrift.http.ssl.keystore.password	Пароль от хранилища ключей SSL-сертификатов	—
kyuubi.authentication	Тип аутентификации. Возможные значения: `NONE`, `LDAP`, `LDAP,KERBEROS`	NONE
kyuubi.ha.zookeeper.acl.enabled	Указывает, керберизован ли ансамбль ZooKeeper	false
kyuubi.ha.zookeeper.auth.type	Тип аутентификации в ZooKeeper. Возможные значения: `NONE`, `KERBEROS`	NONE
kyuubi.ha.zookeeper.auth.principal	Имя принципала Kerberos, используемое для аутентификации в ZooKeeper	—

kyuubi.ha.zookeeper.auth.keytab	Путь к keytab-файлу Kyuubi Server, используемому для аутентификации в ZooKeeper	—
kyuubi.kinit.principal	Имя принципала в Kerberos	—
kyuubi.kinit.keytab	Путь к keytab-файлу Kyuubi Server	—
kyuubi.spnego.principal	Имя принципала в SPNego. Следует заполнять только в случае использования SPNego для аутентификации	—
kyuubi.spnego.keytab	Путь к keytab-файлу SPNego. Следует заполнять только в случае использования SPNego для аутентификации	—
kyuubi.engine.hive.java.options	Дополнительные Java-опции для движка Hive	—

LDAP Security

Параметр	Описание	Значение по умолчанию
kyuubi.authentication.ldap.url	Список ссылок для LDAP-подключения, разделенных пробелами	—
kyuubi.authentication.ldap.domain	LDAP-домен	—
kyuubi.authentication.ldap.binddn	Отличительное имя пользователя для привязки во время поиска по пользователям/группам. Если значение не указано, то будет использовано username пользователя, пытающегося пройти аутентификацию. Например, `CN=bindUser,CN=Users,DC=subdomain,DC=domain,DC=com`	—
kyuubi.authentication.ldap.bindpw	Пароль пользователя для привязки. Необходим, только если указан `kyuubi.authentication.ldap.binddn`	—
kyuubi.authentication.ldap.baseDN	Базовый DN для поиска	—
kyuubi.authentication.ldap.groupClassKey	Наименование LDAP-атрибута группы для поиска. Например, `group`, `groupOfNames` или `groupOfUniqueNames`	—
kyuubi.authentication.ldap.groupDNPattern	Список шаблонов, разделенных точкой с запятой, который будет применяться для поиска названий групп в данной директории. Для подстановки имени группы в нужном месте используйте `%s` — например, `CN=%s,CN=Groups,DC=subdomain,DC=domain,DC=com`	—
kyuubi.authentication.ldap.groupFilter	Список наименований (коротких имен) групп, разделенных запятыми. Например, `HiveAdmins,HadoopAdmins,Administrators`	—
kyuubi.authentication.ldap.groupMembershipKey	Наименование LDAP-атрибута группы, содержащего список отличительных имен участников группы. Например, `member`, `uniqueMember` или `memberUid`	—
kyuubi.authentication.ldap.guidKey	Наименования LDAP-атрибута, значения которого уникальны в пределах LDAP-сервера. Например, `uid` или `CN`	—
kyuubi.authentication.ldap.userDNPattern	Список шаблонов, разделенных точкой с запятой, который будет применяться для поиска названий групп в данной директории. Для подстановки имени пользователя в нужном месте используйте `%s` — например, `CN=%s,CN=Users,DC=subdomain,DC=domain,DC=com`	—
kyuubi.authentication.ldap.userFilter	Список наименований (коротких имен) пользователей, разделенных запятыми. Например, `hiveuser,impalauser,hiveadmin,hadoopadmin`	—
kyuubi.authentication.ldap.userMembershipKey	Наименование LDAP-атрибута пользователя, содержащего список групп, в которых данный пользователь является участником. Основную (primary) группу указывать не требуется. Например, `memberOf`	—

kyuubi-env.sh

Параметр	Описание	Значение по умолчанию

KYUUBI_HOME	Домашняя директория Kyuubi	/usr/lib/kyuubi
KYUUBI_CONF_DIR	Директория, в которой хранятся конфигурации Kyuubi	/etc/kyuubi/conf
KYUUBI_LOG_DIR	Директория, в которой хранятся логи Kyuubi	/var/log/kyuubi
KYUUBI_PID_DIR	Директория, в которой хранится *.pid*-файл экземпляра Kyuubi	/var/run/kyuubi
KYUUBI_ADDITIONAL_CLASSPATH	Дополнительные файлы и директории, которые должны быть добавлены в classpath Kyuubi. Например, библиотеки SSM для сбора статистики. Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	■ /usr/lib/ssm/lib/smart*
HADOOP_HOME	Домашняя директория Hadoop	/usr/lib/hadoop
HADOOP_LIB_DIR	Директория, в которой хранятся библиотеки Hadoop	${HADOOP_HOME}/lib
KYUUBI_JAVA_OPTS	Параметры Java-машины для Kyuubi	-Djava.library.path=${HADOOP_LIB_DIR}/native/ -Djava.io.tmpdir={{ cluster.config.java_tmpdir | d('/tmp') }}
HADOOP_CLASSPATH	Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	■ $HADOOP_CLASSPATH ■ /usr/lib/ssm/lib/smart*
HADOOP_CONF_DIR	Директория, в которой хранятся конфигурации Hadoop	/etc/hadoop/conf
SPARK_HOME	Домашняя директория Spark	/usr/lib/spark3
SPARK_CONF_DIR	Директория, в которой хранятся конфигурации Spark	/etc/spark3/conf
FLINK_HOME	Домашняя директория Flink	/usr/lib/flink
FLINK_CONF_DIR	Директория, в которой хранятся конфигурации Flink	/etc/flink/conf
FLINK_HADOOP_CLASSPATH	Дополнительные JAR-файлы Hadoop, необходимые Kyuubi при использовании движка Flink. Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	■ $(hadoop classpath) ■ /usr/lib/ssm/lib/smart*
HIVE_HOME	Домашняя директория Hive	/usr/lib/hive
HIVE_CONF_DIR	Директория, в которой хранятся конфигурации Hive	/etc/hive/conf
HIVE_HADOOP_CLASSPATH	Дополнительные JAR-файлы Hadoop, необходимые Kyuubi при использовании движка Hive. Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	■ $(hadoop classpath) ■ /etc/tez/conf/ ■ /usr/lib/tez/* ■ /usr/lib/tez/lib/* ■ /usr/lib/ssm/lib/smart*

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут быть использованы сервисом Kyuubi	—

Custom kyuubi-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *kyuubi-env.sh*	—
Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Custom log4j2-repl.xml	Содержимое конфигурационного файла *log4j2-repl.xml*	log4j2-repl.xml
log4j2.xml	Содержимое конфигурационного файла *log4j2.xml*	log4j2.xml

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)

DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

MySQL

root user

Параметр	Описание	Значение по умолчанию
Password	Пароль пользователя `root`	—

Ozone

Параметр	Описание	Значение по умолчанию
ad-runtime-utils	Файл конфигурации YAML, содержащий параметры установки Java: версию, путь установки и переменную среды	ad-runtime-utils.yml
ozone-env.sh	Содержимое файла *ozone-env.sh*, в котором заданы индивидуальные переменные окружения сервиса Ozone	ozone-env.sh
Topology script	Содержимое скрипта, который должен быть вызван при разрешении DNS-имен в NetworkTopology-имена. Пример: скрипт принимает в качестве аргумента `host.foo.bar` и возвращает в качестве результата `/rack1`	—
Topology data	Необязательный дополнительный текстовый файл со списком соответствий сетевых имен и номеров стоек для скрипта топологии. Будет расположен в */etc/hadoop/conf/topology.data*	—
Ranger plugin enabled	Определяет, включен ли плагин Ranger для Ozone	false

ozone-site.xml

Параметр	Описание	Значение по умолчанию
ozone.administrators	Список разделенных запятыми принципалов пользователей, имеющих права администратора в Ozone	om
ozone.administrators.groups	Список разделенных запятыми групп пользователей, участники которых имеют права администратора в Ozone	hadoop
ozone.replication	Фактор репликации данных в Ozone по умолчанию. Более высокие значения обеспечивают более высокую избыточность	3
ozone.service.id	Уникальный идентификатор сервиса Ozone, используемый для мультикластерных конфигураций	—
ozone.http.basedir	Базовая директория HTTP Jetty server для извлечения компонентов	/srv/ozone/meta/webserver
ozone.network.topology.aware.read	Определяет необходимость чтения данных из ближайшего пайплайна	false

ozone.security.enabled	Определяет использование безопасных подключений для Ozone	false
hadoop.security.authentication	Механизм аутентификации для Hadoop и Ozone	simple
ozone.security.http.kerberos.enabled	Определяет использование HTTP-аутентификации на основе Kerberos для сервисов Ozone	false
ozone.http.filter.initializers	Инициализатор HTTP-фильтра для Kerberos-аутентификации	—
ozone.http.policy	Определяет инициализатор HTTP-фильтра для Kerberos-аутентификации	HTTP_ONLY
hdds.grpc.tls.enabled	Определяет, используется ли TLS для коммуникации HDDS GRPC-сервера	false
ozone.https.client.need-auth	Определяет, должны ли HTTPS-клиенты использовать сертификаты для аутентификации	false

ssl-server.xml

Параметр	Описание	Значение по умолчанию
hdds.prometheus.endpoint.enabled	Включает эндпоинт Prometheus для HDDS	false
ssl.server.keystore.location	Путь к файлу keystore	—
ssl.server.keystore.password	Пароль к файлу keystore	—
ssl.server.truststore.location	Путь к файлу truststore	—
ssl.server.truststore.password	Пароль для файла truststore	—

ranger-ozone-audit.xml

Параметр	Описание	Значение по умолчанию
xasecure.audit.destination.solr.batch.filespool.dir	Путь к Spool-каталогу	/srv/ranger/ozone_plugin/audit_solr_spool
xasecure.audit.destination.solr.urls	URL сервера Solr для сохранения аудит-событий. Оставьте значение пустым или установите значение `NONE` при использовании ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.zookeepers	Указывает строку для подключения ZooKeeper к Solr	—
xasecure.audit.destination.solr.force.use.inmemory.jaas.config	Определяет, использовать ли in-memory JAAS-файл для соединения с Solr	—
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.jaas.Client.loginModuleControlFlag	Определяет требование к успешному завершению работы модуля. Возможные значения: `required`, `requisite`, `sufficient`, `optional`	—

xasecure.audit.jaas.Client.loginModuleName	Имя класса-аутентификатора	—
xasecure.audit.jaas.Client.option.keyTab	Имя keytab-файла, из которого необходимо получить секретный ключ принципала	—
xasecure.audit.jaas.Client.option.principal	Имя используемого принципала	—
xasecure.audit.jaas.Client.option.serviceName	Имя пользователя или сервиса, для которого выполняется вход	—
xasecure.audit.jaas.Client.option.storeKey	Установите значение `true`, чтобы keytab-файл или ключ принципала сохранялись в учетных данных субъекта	false
xasecure.audit.jaas.Client.option.useKeyTab	Установите значение `true`, чтобы модуль получал ключ принципала из keytab-файла	false

ranger-ozone-security.xml

Параметр	Описание	Значение по умолчанию
ranger.plugin.ozone.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.ozone.service.name	Имя сервиса Ranger, хранящего политики для данного экземпляра Ozone	—
ranger.plugin.ozone.policy.cache.dir	Директория, в которой хранятся политики Ranger после успешного извлечения из источника	/srv/ranger/ozone/policycache
ranger.plugin.ozone.policy.pollIntervalMs	Период проверки политик на актуальность в миллисекундах	30000
ranger.plugin.ozone.policy.rest.client.connection.timeoutMs	Тайм-аут соединения клиента Ranger REST в миллисекундах	120000
ranger.plugin.ozone.policy.rest.client.read.timeoutMs	Тайм-аут чтения клиента Ranger REST в миллисекундах	30000
ranger.plugin.ozone.policy.rest.ssl.config.file	Путь к конфигурационному файлу RangerRestClient SSL-плагина Hive	/etc/hbase/conf/ranger-hbase-policymgr-ssl.xml

ranger-ozone-policymgr-ssl.xml

Параметр	Описание	Значение по умолчанию
xasecure.policymgr.clientssl.keystore	Путь к keystore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.credential.file	Путь к файлу с паролями для keystore-файла	/etc/ozone/conf/ranger-ozone.jceks
xasecure.policymgr.clientssl.truststore.credential.file	Путь к truststore-файлу с учетными данными	/etc/ozone/conf/ranger-ozone.jceks
xasecure.policymgr.clientssl.truststore	Путь к truststore-файлу, который использует Ranger	—

xasecure.policymgr.clientssl.keystore.password	Пароль для доступа к keystore-файлу	—
xasecure.policymgr.clientssl.truststore.password	Пароль для доступа к truststore-файлу	—

Credential encryption

Параметр	Описание	Значение по умолчанию
Encryption enable	Определяет использование шифрования для учетных данных	false
Credential provider path	Путь к провайдеру учетных данных для создания *.jceks*-файлов, содержащих закрытые ключи	jceks://file/etc/ozone/conf/ozone.jceks
Ranger plugin credential provider path	Путь к провайдеру учетных данных плагина Ranger	jceks://file/etc/ozone/conf/ranger-ozone.jceks
Custom jceks	Определяет, используются ли кастомные *.jceks*-файлы, расположенные по пути провайдера учетных данных (`true`) или автосгенерированные (`false`)	false
Password file name	Имя файла пароля в classpath сервиса, если файл пароля выбран в настройках credstore	ozone_credstore_pass

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom ozone-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ozone-site.xml*	—
Custom ssl-server.xml.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ssl-server.xml.xml*	—
Custom ranger-ozone-audit.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-ozone-audit.xml*	—
Custom ranger-ozone-security.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-ozone-security.xml*	—
Custom ranger-ozone-policymgr-ssl.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-ozone-policymgr-ssl.xml*	—

Компонент Ozone Datanode

ozone-site.xml

Параметр	Описание	Значение по умолчанию
hdds.datanode.http-address	HTTP-адрес веб-интерфейса Datanode	0.0.0.0:9882

hdds.datanode.https-address	HTTPS-адрес веб-интерфейса Datanode	0.0.0.0:9883
ozone.scm.datanode.id.dir	Путь к директории, в которой Datanode хранит свой файл с уникальным идентификатором	/srv/ozone/meta/node
hdds.datanode.dir	Определяет, где в локальной файловой системе Datanode должен хранить свои блоки. Если указаны несколько каталогов, то данные будут храниться во всех именованных каталогах, как правило, на разных устройствах. Каталоги должны быть помечены соответствующими типами хранилища (`SSD` / `DISK` / `ARCHIVE` / `RAM_DISK`). Типом хранилища по умолчанию служит `DISK`, если каталог не имеет явного тега типа хранилища. Несуществующие каталоги будут созданы, если имеются соответствующие права локальной файловой системы	/srv/ozone/data:DISK
hdds.datanode.container.db.dir	Определяет, где в локальной файловой системе Datanode должен хранить файлы контейнеров БД. Если указаны несколько каталогов, то данные будут храниться во всех именованных каталогах, как правило, на разных устройствах. Каталоги должны быть помечены соответствующими типами хранилища (`SSD` / `DISK` / `ARCHIVE` / `RAM_DISK`). Типом хранилища по умолчанию служит `DISK`, если каталог не имеет явного тега типа хранилища. Несуществующие каталоги будут созданы, если имеются соответствующие права локальной файловой системы	/srv/ozone/data/db:DISK
hdds.container.ratis.datanode.storage.dir	Путь к директории для хранения логов Ratis для Datanode	/srv/ozone/data/logs
hdds.container.ratis.enabled	Определяет, используется ли Ratis для Datanode для обеспечения режима высокой доступности и для репликации	true
hdds.container.ratis.datastream.enabled	Определяет, поддерживается ли DataStream в Ratis для эффективного обмена данными между экземплярами Datanode	true
hdds.container.ratis.datastream.port	Порт, используемый для Ratis DataStream на Datanode	9855
ozone.fs.datastream.enabled	Определяет, используется ли для Ozone обмен данными через файловую систему	true
ozone.container.cache.size	Размер кеша контейнера для метаданных и других часто используемых данных, в мегабайтах	8192
ozone.container.cache.lock.stripes	Количество lock stripe для кеша контейнера. Используется для управления параллельным доступом	8192
hdds.datanode.du.factory.classname	Определяет фабричный класс для подсчета использования дискового пространства на Datanode	org.apache.hadoop.hdds.fs.DedicatedDiskSpaceUsageFactory
hdds.container.report.interval	Интервал, через который Datanode отправляет в Storage Container Manager отчеты о контейнерах	10m
hdds.container.ratis.leader.pending.bytes.limit	Ограничение на количество ожидаемых данных для ведущего Ratis в Datanode	2GB
ozone.recon.address	Адрес для подключения к серверу Recon для передачи метрик и отчетов Datanode	{{ groups['ozone.ozone_recon'][0] | d('0.0.0.0') }}:9891
dfs.datanode.kerberos.principal	Kerberos-принципал сервиса Datanode	—
dfs.datanode.kerberos.keytab.file	Путь к keytab-файлу, который демон Datanode использует как принципал сервиса для авторизации	—
hdds.datanode.http.auth.type	Механизм аутентификации для HTTP-сервера Datanode	—
hdds.datanode.http.auth.kerberos.principal	Принципал сервиса HTTP-сервера Datanode	—

hdds.datanode.http.auth.kerberos.keytab	Путь к keytab-файлу, который HTTP-сервер Datanode использует как принципал сервиса для авторизации	—

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom ozone-site.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ozone-site.xml*	—

Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	`[Service]`
		`DefaultLimitCPU=`
		`DefaultLimitFSIZE=`
		`DefaultLimitDATA=`
		`DefaultLimitSTACK=`
		`DefaultLimitCORE=`
		`DefaultLimitRSS=`
		`DefaultLimitNOFILE=`
		`DefaultLimitAS=`
		`DefaultLimitNPROC=`
		`DefaultLimitMEMLOCK=`
		`DefaultLimitLOCKS=`
		`DefaultLimitSIGPENDING=`
		`DefaultLimitMSGQUEUE=`
		`DefaultLimitNICE=`
		`DefaultLimitRTPRIO=`
		`DefaultLimitRTTIME=`

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)

DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Ozone HttpFS

httpfs-env.sh

Параметр	Описание	Значение по умолчанию
HTTPFS_CONFIG	Путь к директории с конфигурационными файлами HttpFS	${OZONE_CONF_DIR}
HTTPFS_LOG	Путь к директории с журналами HttpFS	${OZONE_LOG_DIR}
HTTPFS_TEMP	Путь к временной директории HttpFS	${OZONE_LOG_DIR}
HADOOP_PID_DIR	Путь к директории, в которой HttpFS хранит файлы идентификаторов процессов	${OZONE_LOG_DIR}
HDFS_HTTPFS_OPTS	Опции JVM для сервиса HttpFS	-Xms700m -Xmx8G

httpfs-site.xml

Параметр	Описание	Значение по умолчанию
httpfs.http.port	Порт сервиса HttpFS	14001
hadoop.http.temp.dir	Путь к временной директории сервиса HttpFS	${hadoop.tmp.dir}/httpfs
httpfs.hadoop.config.dir	Путь к конфигурационной директории Hadoop для HttpFS	/etc/ozone/conf
httpfs.http.administrators	Список администраторов сервиса HttpFS	*
httpfs.proxyuser.om.groups	Список групп пользователей для аутентификации Ozone Manager в HttpFS	*
httpfs.proxyuser.om.hosts	Список хостов для аутентификации Ozone Manager в HttpFS	*
httpfs.proxyuser.hue.groups	Список групп пользователей для аутентификации HUE в HttpFS	*
httpfs.proxyuser.hue.hosts	Список хостов для аутентификации HUE в HttpFS	*
httpfs.hadoop.authentication.kerberos.keytab	Файл keytab для Kerberos-аутентификации Ozone Manager в HttpFS	—

httpfs.hadoop.authentication.kerberos.principal	Принципал Kerberos для аутентификации Ozone Manager в HttpFS	—
httpfs.hadoop.authentication.type	Механизм аутентификации для Ozone Manager	simple
hadoop.http.authentication.kerberos.keytab	Путь к файлу keytab для Kerberos-аутентификации HttpFS	—
hadoop.http.authentication.kerberos.principal	Принципал Kerberos для аутентификации HttpFS	—
hadoop.http.authentication.type	Механизм аутентификации для HttpFS	simple
httpfs.ssl.enabled	Определяет, используется ли SSL для HttpFS	false

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom httpfs-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *httpfs-env.sh*	—
Custom httpfs-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *httpfs-site.xml*	—
Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)

DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Ozone Manager

ozone-site.xml

Параметр	Описание	Значение по умолчанию
ozone.om.address	Адрес Ozone Manager	0.0.0.0:9862
ozone.om.enable.filesystem.paths	Определяет, используются ли операции path-style в файловой системе	true
ozone.om.http-address	HTTP-адрес веб-интерфейса Ozone Manager	0.0.0.0:9874
ozone.om.https-address	HTTPS-адрес веб-интерфейса Ozone Manager	0.0.0.0:9875
ozone.om.ratis.port	Порт для коммуникации с Ratis, используемый Ozone Manager	9858
ozone.om.db.dirs	Путь к директории, в которой Ozone Manager хранит свои файлы БД	/srv/ozone/meta/db
ozone.om.db.dirs.permissions	Определяет режим доступа к директории, заданной в параметре `ozone.om.db.dirs`	750
ozone.metadata.dirs	Путь к директории для хранилища общих метаданных в Ozone	/srv/ozone/meta

ozone.metadata.dirs.permissions	Определяет режим доступа к директории, заданной в параметре `ozone.metadata.dirs`	750
ozone.om.snapshot.diff.db.dir	Путь к директории для хранения БД различий между снепшотами для Ozone Manager	/srv/ozone/meta/snapshot
ozone.om.ratis.enable	Определяет, используется ли основанная на Ratis репликация для Ozone Manager для обеспечения режима высокой доступности	true
ozone.om.service.ids	Идентификаторы для сервисов Ozone Manager, используемые в инсталляциях с множественными экземплярами для обеспечения режима высокой доступности	—
ozone.om.kerberos.principal	Принципал сервиса Ozone Manager	—
ozone.om.kerberos.keytab.file	Путь к keytab-файлу, который демон Ozone Manager использует как принципал сервиса для авторизации	—
ozone.om.http.auth.type	Механизм аутентификации для HTTP-сервера Ozone Manager	simple
ozone.om.http.auth.kerberos.principal	Принципал сервиса HTTP-сервера Ozone Manager при включенном SPNEGO	—
ozone.om.http.auth.kerberos.keytab	Путь к файлу keytab, который HTTP-сервер Ozone Manager использует как принципал сервиса для авторизации при включенном SPNEGO	—

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom ozone-site.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ozone-site.xml*	—
Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)

DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Ozone Recon

ozone-site.xml

Параметр	Описание	Значение по умолчанию
ozone.recon.db.dir	Путь к директории, в которой Recon хранит свои файлы БД для метрик и отчетов	/srv/ozone/recon/db
ozone.recon.task.pipelinesync.interval	Интервал для задачи синхронизации информации конвейеров от компонента Storage Container Manager	120s
ozone.recon.task.missingcontainer.interval	Интервал, используемый для запуска проверки отсутствующих контейнеров в кластере и формирования отчетов об этом	3600s
ozone.recon.http-address	HTTP-адрес веб-интерфейса Recon, используемый для просмотра метрик и отчетов для кластера Ozone	0.0.0.0:9888
ozone.recon.https-address	HTTPS-адрес веб-интерфейса Recon	0.0.0.0:9889
ozone.recon.kerberos.principal	Принципал сервиса Recon	—

ozone.recon.kerberos.keytab.file	Путь к keytab-файлу, который демон Recon использует как принципал сервиса для авторизации	—
ozone.recon.http.auth.type	Механизм аутентификации для HTTP-сервера Recon	simple
ozone.recon.http.auth.kerberos.principal	Принципал сервиса HTTP-сервера Recon для Kerberos-аутентификации	—
ozone.recon.http.auth.kerberos.keytab	Путь к файлу keytab, который HTTP-сервер Recon использует как принципал сервиса для авторизации	—

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom ozone-site.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ozone-site.xml*	—
Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	

Enable custom ulimits:

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для	open files (-n)

	процесса	
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Ozone S3G

ozone-site.xml

Параметр	Описание	Значение по умолчанию
ozone.s3g.http-address	HTTP-адрес шлюза S3, который предоставляет S3-совместимый API для Ozone	0.0.0.0:9878
ozone.s3g.https-address	HTTPS-адрес шлюза S3	0.0.0.0:9879
ozone.s3g.volume.name	Название тома, используемого шлюзом S3 для хранения S3-совместимых бакетов в Ozone	s3v
ozone.s3g.client.buffer.size	Размер буфера клиента, используемого шлюзом S3 для передачи данных	4KB
ozone.s3g.kerberos.principal	Принципал сервиса шлюза S3	—
ozone.s3g.kerberos.keytab.file	Путь к keytab-файлу, который шлюз S3 использует как принципал сервиса для авторизации	—
ozone.s3g.http.auth.type	Механизм аутентификации для HTTP-сервера S3G	simple
ozone.s3g.http.auth.kerberos.principal	Принципал сервиса HTTP-сервера шлюза S3 при включенном SPNEGO	—
ozone.s3g.http.auth.kerberos.keytab	Путь к файлу keytab, который HTTP-сервер шлюза S3 использует как принципал сервиса для авторизации при включенном SPNEGO	—

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom ozone-site.conf	В этой секции вы можете указать значения для кастомных	—

	параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ozone-site.xml*	
Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)

DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Ozone Storage Container Manager

ozone-site.xml

Параметр	Описание	Значение по умолчанию
ozone.scm.http-address	HTTP-адрес веб-интерфейса Storage Container Manager	0.0.0.0:9876
ozone.scm.https-address	HTTPS-адрес веб-интерфейса Storage Container Manager	0.0.0.0:9877
ozone.scm.ratis.port	Порт для коммуникации с Ratis, используемый Storage Container Manager для обеспечения режима высокой доступности	9894
ozone.scm.db.dirs	Путь к директории, в которой Storage Container Manager хранит свои файлы БД	/srv/ozone/scm/db
ozone.scm.db.dirs.permissions	Определяет режим доступа к директории, заданной в параметре `ozone.scm.db.dirs`	750
ozone.scm.ha.ratis.storage.dir	Путь к директории для хранения логов Ratis при использовании режима высокой доступности в Storage Container Manager	/srv/ozone/scm/ratis
ozone.scm.ha.ratis.snapshot.dir	Путь к директории для хранения снепшотов Ratis в Storage Container Manager	/srv/ozone/scm/ratis/snapshot
ozone.scm.service.ids	Идентификаторы для сервисов Storage Container Manager, используемые в инсталляциях с множественными экземплярами для обеспечения режима высокой доступности	—
ozone.scm.primordial.node.id	Идентификатор первого узла Storage Container Manager в кластере	{{ groups['ozone.ozone_scm'][0] }}
ozone.scm.names	Список разделенных запятыми полных доменных имен (FQDN) узлов Storage Container Manager в кластере	{{ groups['ozone.ozone_scm'] | join(',') }}
ozone.scm.client.address	Список разделенных запятыми адресов клиентов Storage Container Manager. Как правило, представляют собой полные доменные имена узлов Storage Container Manager	{{ groups['ozone.ozone_scm'] | join(',') }}
ozone.scm.ratis.enable	Определяет, включена ли репликация, основанная на Ratis, для Storage Container Manager для обеспечения режима высокой доступности	true
ozone.scm.datanode.pipeline.limit	Максимальное количество конвейеров в Storage Container Manager, в которых может участвовать каждый Datanode	10
ozone.scm.pipeline.owner.container.count	Максимальное количество контейнеров, принадлежащих каждому конвейеру	10
ozone.scm.pipeline.creation.auto.factor.one	Определяет, должен ли Storage Container Manager автоматически создавать конвейеры с фактором репликации, равным 1	false
ozone.scm.container.placement.impl	Определяет политику размещения контейнеров для Storage Container Manager, обычно для оптимизации использования пространства в зависимости от доступных ресурсов	org.apache.hadoop.hdds.scm.container.placement.algorithms.SCMContainerPlacementCapacity
ozone.scm.kerberos.principal	Принципал сервиса Storage Container Manager	—

ozone.scm.kerberos.keytab.file	Путь к keytab-файлу, который демон Storage Container Manager использует как принципал сервиса для авторизации	—
ozone.scm.http.auth.type	Механизм аутентификации для HTTP-сервера Storage Container Manager	simple
ozone.scm.http.auth.kerberos.principal	Принципал сервиса HTTP-сервера Storage Container Manager при включенном SPNEGO	—
ozone.scm.http.auth.kerberos.keytab	Путь к файлу keytab, который HTTP-сервер Storage Container Manager использует как принципал сервиса для авторизации при включенном SPNEGO	—

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom ozone-site.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ozone-site.xml*	—
Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)

DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Solr

solr-env.sh

Параметр	Описание	Значение по умолчанию
SOLR_HOME	Расположение индексных данных и конфигураций	/srv/solr/server
SOLR_AUTH_TYPE	Указывает тип аутентификации для Solr	—
SOLR_AUTHENTICATION_OPTS	Параметры аутентификации Solr	—
SOLR_AUTHENTICATION_OPTS_CUSTOM	Кастомные параметры аутентификации Solr	—
GC_TUNE	JVM-параметры для Solr	-XX:-UseLargePages
SOLR_SSL_KEY_STORE	Путь к keystore-файлу Solr	—
SOLR_SSL_KEY_STORE_TYPE	Тип keystore-хранилища	JKS
SOLR_SSL_KEY_STORE_PASSWORD	Пароль к keystore-файлу Solr	—
SOLR_SSL_TRUST_STORE	Путь к truststore-файлу Solr	—
SOLR_SSL_TRUST_STORE_TYPE	Тип truststore-хранилища	JKS
SOLR_SSL_TRUST_STORE_PASSWORD	Пароль к truststore-файлу Solr	—
SOLR_SSL_NEED_CLIENT_AUTH	Определяет, включена ли клиентская аутентификация	false

SOLR_SSL_WANT_CLIENT_AUTH	Позволяет клиентам выполнять клиентскую аутентификацию (но не требует этого)	false
SOLR_SSL_CLIENT_HOSTNAME_VERIFICATION	Определяет, включена ли проверка имен хостов	false
SOLR_HOST	Указывает имя хоста Solr-сервера	—
SOLR_PORT	Номер порта, который слушает каждый Solr-узел	8983
LOG4J_PROPS	Разделенный запятыми список абсолютных путей, указывающих на конфигурационные файлы Log4j для использования Solr-сервером	/etc/solr/conf/log4j2.xml,/etc/solr/conf/log4j2-console.xml

External zookeeper

Параметр	Описание	Значение по умолчанию
ZK_HOST	Список всех серверов в Ensemble, включая порты, по которым они взаимодействуют. Вы можете добавить путь ZooKeeper chroot в конец строки подключения `ZK_HOST`. Например, `host1.mydomain.com:2181,host2.mydomain.com:2181,host3.mydomain.com:2181/solr`	—
The external zookeeper is kerberized	Если внешний ZooKeeper керберизирован, значение данного параметра должно быть `true`	false

Solr server heap memory settings

Параметр	Описание	Значение по умолчанию
Solr Server Heap Memory	Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap для Solr Server	-Xms512m -Xmx512m

Solr collections ttl settings

Параметр	Описание	Значение по умолчанию
collection_name	Имя коллекции для настройки автоматической очистки	—
ttl	TTL (время жизни) документов в коллекции	—
auto_delete_period	Временной интервал для автоматического удаления документов из коллекции	—

Credential Encryption

Параметр	Описание	Значение по умолчанию
Credstore password	Пароль провайдера шифрования	false
Credstore options	Способ хранения пароля провайдера шифрования. Доступны следующие опции: ▪ `no password` — пароль не используется; ▪ `password in the environment` — пароль хранится в переменной окружения.	no password

Credential provider path	Путь к хранилищу ключей JCEKS	jceks://file/etc/solr/conf/solr.jceks
Ranger plugin credential provider path	Путь к хранилищу ключей JCEKS, используемому Ranger	jceks://file/etc/solr/conf/ranger-solr.jceks
Custom jceks	Укажите значение `true` при использовании кастомного файла JCEKS-хранилища. Укажите значение `false` для использования дефолтного файла хранилища, который генерируется автоматически	false

ranger-solr-audit.xml

Параметр	Описание	Значение по умолчанию
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.solr.solr_url	Путь к Solr-коллекции для сохранения аудит-логов	—
xasecure.audit.solr.async.max.queue.size	Максимальный размер внутренней очереди для хранения аудит-логов	1
xasecure.audit.solr.async.max.flush.interval.ms	Максимальный временной интервал между сбросом лог-данных на диск (в миллисекундах)	100
xasecure.audit.solr.is.enabled	Активирует аудит Ranger для Solr	true
ranger.solr.plugin.audit.excluded.users	Список пользователей, для которых записи аудита должны быть исключены	HTTP,rangeradmin,rangerkms

ranger-solr-security.xml

Параметр	Описание	Значение по умолчанию
ranger.plugin.solr.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.solr.service.name	Имя Ranger-сервиса, который содержит политики для этого экземпляра	—
ranger.plugin.solr.policy.cache.dir	Каталог, в котором кешируются политики Ranger после успешного извлечения из источника	/srv/ranger/solr/policycache
ranger.plugin.solr.policy.pollIntervalMs	Определяет, как часто проверять политики на предмет изменения	30000
ranger.plugin.solr.policy.rest.client.connection.timeoutMs	Время ожидания подключения Solr-плагина RangerRestClient (в миллисекундах)	120000
ranger.plugin.solr.policy.rest.client.read.timeoutMs	Тайм-аут на чтение для Solr-плагина RangerRestClient (в миллисекундах)	30000
ranger.plugin.solr.policy.rest.ssl.config.file	Путь к файлу, содержащему данные SSL для связи с Ranger Admin	/usr/lib/solr/server/resources/ranger-solr-policymgr-ssl.xml
ranger.plugin.solr.policy.source.impl	Класс для получения политик из источника	org.apache.ranger.admin.client.RangerAdminRESTClient

ranger-solr-policymgr-ssl.xml

Параметр	Описание	Значение по умолчанию
xasecure.policymgr.clientssl.keystore	Путь к keystore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.credential.file	Путь к файлу с паролями для keystore-файла	/etc/solr/conf/ranger-solr.jceks
xasecure.policymgr.clientssl.truststore.credential.file	Путь к truststore-файлу с учетными данными	/etc/solr/conf/ranger-solr.jceks
xasecure.policymgr.clientssl.truststore	Путь к truststore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.password	Пароль для доступа к keystore-файлу	—
xasecure.policymgr.clientssl.truststore.password	Пароль для доступа к truststore-файлу	—

Другие параметры

Параметр	Описание	Значение по умолчанию
solr.xml	Содержимое файла *solr.xml*	Default solr.xml
Custom solr-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *solr-env.sh*	—
Ranger plugin enabled	Отображает статус Ranger-плагина	false
Custom ranger-solr-audit.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-solr-audit.xml*	—
Custom ranger-solr-security.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-solr-security.xml*	—
Custom ranger-solr-policymgr-ssl.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-solr-policymgr-ssl.xml*	—

Spark

Common

Параметр	Описание	Значение по умолчанию
Dynamic allocation (spark.dynamicAllocation.enabled)	Определяет, использовать ли динамическое выделение ресурсов для масштабирования количества исполнителей, зарегистрированных в этом приложении, в зависимости от рабочей нагрузки	false

Credential Encryption

Параметр	Описание	Значение по умолчанию
Encryption enable	Включает или выключает функцию шифрования учетных данных. Если опция включена, Spark хранит учетные данные (пароли настроек, учетные данные для взаимодействия с другими сервисами) в зашифрованном виде	false
Credential provider path	Путь к хранилищу ключей JCEKS	jceks://hdfs/apps/spark/security/spark.jceks
Custom jceks	Укажите значение `true` при использовании кастомного файла JCEKS-хранилища. Укажите значение `false` для использования дефолтного файла хранилища, который генерируется автоматически	false

spark-defaults.conf

Параметр	Описание	Значение по умолчанию
spark.yarn.archive	Архив, содержащий необходимые JAR-файлы Spark для помещения в кеш YARN. Если параметр задан, он заменяет `spark.yarn.jars` и указанный архив используется во всех контейнерах приложения. Архив должен содержать JAR-файлы в своем корневом каталоге. Архив также может быть размещен на HDFS для ускорения распространения файлов	hdfs:///apps/spark/spark-yarn-archive.tgz
spark.yarn.historyServer.address	Адрес Spark History Server	—
spark.master	Менеджер кластера для соединения	yarn
spark.dynamicAllocation.enabled	Определяет, использовать ли динамическое выделение ресурсов для масштабирования количества исполнителей, зарегистрированных в этом приложении, в зависимости от рабочей нагрузки	false
spark.shuffle.service.enabled	Задействует внешний shuffle-сервис. Сервис сохраняет shuffle-файлы, созданные исполнителями. Таким образом, исполнители могут быть безопасно удалены, а процесс shuffle может продолжаться в случае сбоя исполнителя. Внешний shuffle-сервис должен быть настроен, чтобы использовать его	false
spark.eventLog.enabled	Определяет, следует ли логировать события Spark, что полезно для восстановления веб-интерфейса после завершения работы приложения	true
spark.eventLog.dir	Базовый каталог, в котором регистрируются события Spark, если `spark.eventLog.enabled=true`. В этом каталоге Spark создает подкаталог для каждого приложения и логирует события, относящиеся к приложению в этом каталоге. Вы можете задать единое местоположение, например каталог HDFS, чтобы History Server имел доступ к файлам истории	hdfs:///var/log/spark/apps
spark.serializer	Класс, используемый для сериализации объектов, которые будут отправлены по сети или должны быть кешированы в сериализованной форме. Сериализация Java по умолчанию работает с любым `Serializable` Java-объектом, но довольно медленно, поэтому рекомендуется использовать `org.apache.spark.serializer.KryoSerializer` и настроить сериализацию Kryo, когда необходима скорость. Класс должен наследовать `org.apache.spark.Serializer`	org.apache.spark.serializer.KryoSerializer
spark.dynamicAllocation.executorIdleTimeout	Если динамическое размещение включено и исполнитель бездействует дольше указанного времени, такой исполнитель будет удален. Дополнительные сведения доступны в документации Spark	120s

spark.dynamicAllocation.cachedExecutorIdleTimeout	Если динамическое размещение включено и исполнитель, у которого имеются блоки данных в кеше, бездействует дольше указанного времени, такой исполнитель будет удален. Дополнительные сведения доступны в документации Spark	600s
spark.history.provider	Имя класса, реализующего backend-часть истории приложения. В настоящее время Spark предоставляет только одну реализацию, которая ищет логи приложений, хранящиеся в файловой системе	org.apache.spark.deploy.history.FsHistoryProvider
spark.history.fs.cleaner.enabled	Указывает, должен ли History Server периодически удалять логи событий из хранилища	true
spark.history.store.path	Локальный каталог, в котором кешируются данные истории приложений. Если параметр задан, History Server будет хранить данные приложения на диске, а не в памяти. Данные, записанные на диск, будут повторно использованы в случае перезапуска History Server	/var/log/spark3/history
spark.driver.extraClassPath	Дополнительные зависимости, которые будут добавлены в classpath драйвера	▪ /usr/lib/hive/lib/hive-shims-scheduler.jar ▪ /usr/lib/hadoop-yarn/hadoop-yarn-server-resourcemanager.jar
spark.executor.extraClassPath	Дополнительные зависимости, которые будут добавлены в executor classpath	—
spark.history.ui.port	Номер порта web-интерфейса History Server	18082
spark.history.fs.logDirectory	Директория с логами History Server	hdfs:///var/log/spark/apps
spark.driver.extraLibraryPath:	Путь к дополнительным нативным библиотекам для драйвера	/usr/lib/hadoop/lib/native/
spark.yarn.am.extraLibraryPath:	Путь к дополнительным нативным библиотекам для Application Master	/usr/lib/hadoop/lib/native/
spark.executor.extraLibraryPath	Путь к дополнительным нативным библиотекам для Executor	/usr/lib/hadoop/lib/native/
spark.yarn.appMasterEnv.HIVE_CONF_DIR	Путь к директории на Application Master, где хранятся настройки Hive, необходимые для запуска в кластерном режиме	/etc/spark3/conf
spark.yarn.historyServer.allowTracking	Позволяет использовать Spark History Server для интерфейса трекинга, если web UI отключен для задачи	true
spark.ssl.enabled	Определяет, использовать ли SSL для Spark	false
spark.ssl.protocol	Используемый TLS-протокол. Протокол должен поддерживаться JVM	TLSv1.2
spark.ssl.ui.port	Порт, через который устанавливается SSL-соединение	4040
spark.ssl.historyServer.port	Порт для доступа к веб-интерфейсу History Server	18082
spark.ssl.keyPassword	Пароль приватного ключа в keystore-файле	—
spark.ssl.keyStore	Путь к keystore-файлу	—
spark.ssl.keyStoreType	Тип keystore-файла	JKS
spark.ssl.trustStorePassword	Пароль к truststore-файлу, который использует Spark	—
spark.ssl.trustStore	Путь к truststore-файлу	—

spark.ssl.trustStoreType	Тип truststore-файла	JKS
spark.history.kerberos.enabled	Указывает, должен ли History Server использовать Kerberos для входа в систему. Этот параметр необходим, если History Server обращается к файлам HDFS в защищенном кластере Hyperwave	false
spark.acls.enable	Активирует список управления доступом Spark	false
spark.modify.acls	Список пользователей, разделенных запятыми, у которых есть права на изменение приложений Spark	spark,hdfs
spark.modify.acls.groups	Список групп пользователей, разделенных запятыми, у которых есть права на изменение приложений Spark	spark,hdfs
spark.history.ui.acls.enable	Указывает, следует ли проверять списки управления доступом, чтобы разрешить пользователям просматривать приложения в History Server. Если указано `true`, проверка контроля доступа выполняются независимо от того, какие значения `spark.ui.acls.enable` установлены для отдельных приложений. Если указано `false`, проверки контроля доступа не выполняются для всех пользовательских интерфейсов приложений, доступных через History Server	false
spark.history.ui.admin.acls	Разделенный запятыми список пользователей, у которых есть доступ к просмотру всех приложений Spark в History Server	spark,hdfs,dr.who
spark.history.ui.admin.acls.groups	Разделенный запятыми список групп пользователей, у которых есть доступ к просмотру всех приложений Spark в History Server	spark,hdfs,dr.who
spark.ui.view.acls	Разделенный запятыми список пользователей, у которых есть доступ к просмотру приложения Spark. По умолчанию только пользователь, запустивший Spark-задачу, имеет доступ к просмотру. Использование `*` в качестве значения означает, что любой пользователь может иметь доступ к просмотру задачи Spark	spark,hdfs,dr.who
spark.ui.view.acls.groups	Разделенный запятыми список групп пользователей, у которых есть доступ к просмотру приложения Spark. По умолчанию только пользователь, запустивший Spark-задачу, имеет доступ к просмотру. Использование `*` в качестве значения означает, что любой пользователь может иметь доступ к просмотру задачи Spark. Группы пользователей предоставляет экземпляр провайдера сопоставления групп (group mapping provider), указанный в `spark.user.groups.mapping`	spark,hdfs,dr.who

Spark Heap Memory settings

Параметр	Описание	Значение по умолчанию
Spark History Server Heap Memory	Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap для Spark History Server	1G
Livy Server Heap Memory	Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap для Livy Server	-Xms300m -Xmx4G

Custom log4j.properties

Параметр	Описание	Значение по умолчанию
Spark spark-log4j.properties	Содержит конфигурацию Log4j, которая используется Spark	spark-log4j.properties
Livy livy-log4j.properties	Содержит конфигурацию Log4j, которая используется Livy	livy-log4j.properties

livy.conf

Параметр	Описание	Значение по умолчанию
livy.server.host	Хост, на котором запускается Livy-сервер. По умолчанию Livy слушает все сетевые интерфейсы	0.0.0.0
livy.server.port	Порт, на котором запускается Livy-сервер	8999
livy.spark.master	Spark master для Livy-сессий	yarn
livy.impersonation.enabled	Определяет, следует ли Livy имперсонировать пользователей при создании новой сессии	true
livy.server.csrf-protection.enabled	Определяет, задействовать ли CSRF-защиту. Если защита включена, клиентские приложения должны добавлять HTTP-заголовок `X-Requested-By` для HTTP-методов POST/DELETE/PUT/PATCH	true
livy.repl.enable-hive-context	Определяет, задействовать ли HiveContext для интерпретатора Livy. Если указано значение `true`, *hive-site.xml* и classpath Livy-сервера будут определены автоматически по запросу пользователя	true
livy.server.recovery.mode	Устанавливает режим восстановления для Livy	recovery
livy.server.recovery.state-store	Определяет, где Livy хранит состояние (state) для восстановления	filesystem
livy.server.recovery.state-store.url	Если выбран тип хранилища состояния `filesystem`, определяет путь к каталогу хранилища. Не рекомендуется использовать файловые системы, которые не поддерживают атомарное переименование (например, S3). Например: `file:///tmp/livy` or `hdfs:///`. Для ZooKeeper необходимо указать адрес ZooKeeper-серверов. Например: `host1:port1,host2:port2`	/livy-recovery
livy.server.auth.type	Указывает тип аутентификации для Livy	—
livy.server.access_control.enabled	Определяет, включать ли контроль доступа для сервера Livy. Если установлено значение `true`, все входящие запросы будут проверяться на наличие у запрошенного пользователя разрешения	false
livy.server.access_control.users	Пользователи, которым разрешен доступ к Livy. По умолчанию любому пользователю разрешен доступ к Livy. Если пользователь хочет ограничить доступ, он должен перечислить всех разрешенных пользователей через запятую	livy,hdfs,spark
livy.superusers	Список разделенных запятыми пользователей, у которых есть разрешения на изменение сеанса другого пользователя, например на выполнение запросов, удаление сеанса и так далее	livy,hdfs,spark
livy.keystore	Путь к keystore-файлу. Путь может быть как абсолютным, так и относительным каталогу, в котором был запущен процесс	—
livy.keystore.password	Пароль для доступа к keystore-файлу	—
livy.key-password	Пароль для доступа к ключу в keystore-файле	—
livy.server.thrift.ssl.protocol.blacklist	Список запрещенных к использованию TLS-протоколов	SSLv2,SSLv3,TLSv1,TLSv1.1

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom spark-defaults.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *spark-defaults.conf*	—

spark-env.sh	Содержимое файла *spark-env.sh*, который используется для инициализации окружения на worker-узлах	spark-env.sh
Custom livy.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *livy.conf*	—
livy-env.sh	Содержимое файла *livy-env.sh*, который используется для инициализации окружения перед запуском Livy	livy-env.sh
spark-history-env.sh	Содержимое файла *spark-history-env.sh*, который используется для инициализации окружения перед запуском History Server	spark-history-env.sh
Ranger plugin enabled	Включает или отключает плагин Ranger	false

Spark3

Common

Параметр	Описание	Значение по умолчанию
Dynamic allocation (spark.dynamicAllocation.enabled)	Определяет, использовать ли динамическое выделение ресурсов, для масштабирования количества исполнителей, зарегистрированных в этом приложении, в зависимости от рабочей нагрузки	false

Credential Encryption

Параметр	Описание	Значение по умолчанию
Encryption enable	Включает или выключает функцию шифрования учетных данных. Если опция включена, Spark хранит учетные данные (пароли настроек, учетные данные для взаимодействия с другими сервисами) в зашифрованном виде	false
Credential provider path	Путь к хранилищу ключей JCEKS	jceks://hdfs/apps/spark/security/spark.jceks
Custom jceks	Укажите значение `true` при использовании кастомного файла JCEKS-хранилища. Укажите значение `false` для использования дефолтного файла хранилища, который генерируется автоматически	false

spark3_iceberg_extensions

Параметр	Описание	Значение по умолчанию
version	Версия пакета spark3_iceberg_extensions	1.5.2_arenadata1

spark-defaults.conf

Параметр	Описание	Значение по умолчанию

spark.yarn.archive	Архив, содержащий необходимые JAR-файлы Spark для помещения в кеш YARN. Если параметр задан, он заменяет `spark.yarn.jars`, и указанный архив используется во всех контейнерах приложения. Архив должен содержать JAR-файлы в своем корневом каталоге. Архив также может быть размещен на HDFS для ускорения распространения файлов	hdfs:///apps/spark/spark3-yarn-archive.tgz
spark.yarn.historyServer.address	Адрес Spark History server	—
spark.master	Менеджер кластера для соединения	yarn
spark.dynamicAllocation.enabled	Определяет, использовать ли динамическое выделение ресурсов для масштабирования количества исполнителей, зарегистрированных в этом приложении, в зависимости от рабочей нагрузки	false
spark.shuffle.service.enabled	Задействует внешний shuffle-сервис. Сервис сохраняет shuffle-файлы, созданные исполнителями. Таким образом, исполнители могут быть безопасно удалены, а процесс shuffle может продолжаться в случае сбоя исполнителя. Внешний shuffle-сервис должен быть настроен, чтобы использовать его	false
spark.eventLog.enabled	Определяет, следует ли логировать события Spark, что полезно для восстановления веб-интерфейса после завершения работы приложения	true
spark.eventLog.dir	Базовый каталог, в котором регистрируются события Spark, если `spark.eventLog.enabled=true`. В этом каталоге Spark создает подкаталог для каждого приложения и логирует события, относящиеся к приложению в этом каталоге. Вы можете задать единое местоположение, например каталог HDFS, чтобы History Server имел доступ к файлам истории	hdfs:///var/log/spark/apps
spark.dynamicAllocation.executorIdleTimeout	Если динамическое размещение включено и исполнитель бездействует дольше указанного времени, такой исполнитель будет удален. Дополнительные сведения доступны в документации Spark	120s
spark.dynamicAllocation.cachedExecutorIdleTimeout	Если динамическое размещение включено и исполнитель, у которого имеются блоки данных в кеше, бездействует дольше указанного времени, такой исполнитель будет удален. Дополнительные сведения доступны в документации Spark	600s
spark.history.provider	Имя класса, реализующего backend-часть истории приложения. В настоящее время Spark предоставляет только одну реализацию, которая ищет логи приложений, хранящиеся в файловой системе	org.apache.spark.deploy.history.FsHistoryProvider
spark.history.fs.cleaner.enabled	Указывает, должен ли History Server периодически удалять логи событий из хранилища	true
spark.history.store.path	Локальный каталог, в котором кешируются данные истории приложений. Если параметр задан, History Server будет хранить данные приложения на диске, а не в памяти. Данные, записанные на диск, будут повторно использованы в случае перезапуска History Server	/var/log/spark3/history
spark.serializer	Имя класса, который выполняет сериализацию объектов для отправки их по сети или для кеширования в сериализованном виде. По умолчанию работает с любым объектом, который реализует интерфейс Serializable, однако такой подход является медленным. Поэтому рекомендуется использовать `org.apache.spark.serializer.KryoSerializer` и настроить сериализацию Kryo для большей производительности. В значении может быть любой класс-наследник `org.apache.spark.Serializer`	org.apache.spark.serializer.KryoSerializer
spark.driver.extraClassPath	Дополнительные зависимости, которые будут добавлены в classpath драйвера	■ /usr/lib/hive/lib/hive-shims-scheduler.jar ■ /usr/lib/hadoop-yarn/hadoop-yarn-server-resourcemanager.jar ■ /usr/lib/spark3/jars/adb-spark-connector-assembly-release-1.0.5-spark-3.5.2_arenadata1.jar

		/usr/lib/spark3/jars/adqm-spark-connector-assembly-release-1.0.0-spark-3.5.2_arenadata1.jar
spark.executor.extraClassPath	Дополнительные зависимости, которые будут добавлены в executor classpath	/usr/lib/spark3/jars/adb-spark-connector-assembly-release-1.0.5-spark-3.5.2_arenadata1.jar /usr/lib/spark3/jars/adqm-spark-connector-assembly-release-1.0.0-spark-3.5.2_arenadata1.jar
spark.history.ui.port	Номер порта web-интерфейса History Server	18092
spark.ui.port	Номер порта web-интерфейса Spark	4140
spark.history.fs.logDirectory	Директория с логами History Server	hdfs:///var/log/spark/apps
spark.sql.extensions	Классы SQL-расширений Iceberg, разделенные запятой	org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
spark.sql.catalog.spark_catalog	Класс-имплементация каталога Iceberg	org.apache.iceberg.spark.SparkSessionCatalog
spark.sql.hive.metastore.jars	Путь к JAR-файлам, которые нужны для инициализации HiveMetastoreClient	path
spark.sql.hive.metastore.jars.path	Список путей, разделенных запятыми и указывающих на JAR-файлы, которые необходимы для инициализации HiveMetastoreClient	file:///usr/lib/hive/lib/*.jar
spark.sql.hive.metastore.version	Версия Hive Metastore	3.1.2
spark.driver.extraLibraryPath:	Путь к дополнительным нативным библиотекам для драйвера	/usr/lib/hadoop/lib/native/
spark.yarn.am.extraLibraryPath:	Путь к дополнительным нативным библиотекам для Application Master	/usr/lib/hadoop/lib/native/
spark.executor.extraLibraryPath	Путь к дополнительным нативным библиотекам для Executor	/usr/lib/hadoop/lib/native/
spark.yarn.appMasterEnv.HIVE_CONF_DIR	Путь к директории на Application Master, где хранятся настройки Hive, необходимые для запуска в кластерном режиме	/etc/spark3/conf
spark.yarn.historyServer.allowTracking	Позволяет использовать Spark History Server для интерфейса трекинга, если web UI отключен для задачи	True
spark.connect.grpc.binding.port	Номер порта для gRPC-соединения с сервером Spark Connect	15002
spark.artifactory.dir.path	Директория с артефактами Spark Connect	tmp
spark.sql.security.confblacklist	Запрещает переопределение указанных параметров для корректной работы приложения или из соображений безопасности	spark.sql.extensions
spark.history.kerberos.enabled	Указывает, должен ли History Server использовать Kerberos для входа в систему. Этот параметр необходим, если History Server обращается к файлам HDFS в защищенном кластере Hyperwave	false
spark.acls.enable	Определяет, активен ли Spark ACL (access control list). Если опция включена, выполняется проверка, имеет ли определенный пользователь права доступа на просмотр или изменение задачи Spark. Обратите внимание, что пользователь должен быть известен. Если имя пользователя `null`, никаких проверок не выполняется. В веб-интерфейсе можно использовать фильтры для аутентификации и установки пользователя	false

spark.modify.acls	Список пользователей, разделенных запятыми, у которых есть права на изменение приложений Spark	spark,hdfs
spark.modify.acls.groups	Список групп пользователей, разделенных запятыми, у которых есть права на изменение приложений Spark	spark,hdfs
spark.history.ui.acls.enable	Указывает, следует ли проверять списки управления доступом, чтобы разрешить пользователям просматривать приложения на History Server. Если указано `true`, проверка контроля доступа выполняются независимо от того, какие значения `spark.ui.acls.enable` установлены для отдельных приложений. Если указано `false`, проверки контроля доступа не выполняются для всех пользовательских интерфейсов приложений, доступных через History Server	false
spark.history.ui.admin.acls	Разделенный запятыми список пользователей, у которых есть доступ к просмотру всех приложений Spark на History Server	spark,hdfs,dr.who
spark.history.ui.admin.acls.groups	Разделенный запятыми список групп пользователей, у которых есть доступ к просмотру всех приложений Spark на History Server	spark,hdfs,dr.who
spark.ui.view.acls	Разделенный запятыми список пользователей, у которых есть доступ к просмотру приложения Spark. По умолчанию только пользователь, запустивший Spark-задачу, имеет доступ к просмотру. Использование `*` в качестве значения означает, что любой пользователь может иметь доступ к просмотру задачи Spark	spark,hdfs,dr.who
spark.ui.view.acls.groups	Разделенный запятыми список групп пользователей, у которых есть доступ к просмотру приложения Spark. По умолчанию только пользователь, запустивший Spark-задачу, имеет доступ к просмотру. Использование `*` в качестве значения означает, что любой пользователь может иметь доступ к просмотру задачи Spark. Группы пользователей предоставляет экземпляр провайдера сопоставления групп (group mapping provider), указанный в `spark.user.groups.mapping`	spark,hdfs,dr.who
spark.ssl.keyPassword	Пароль к приватному ключу в keystore	—
spark.ssl.keyStore	Путь к keystore-файлу. Путь может быть как абсолютным, так и относительным директории, в которой был запущен процесс	—
spark.ssl.keyStoreType	Тип используемого keystore	JKS
spark.ssl.trustStorePassword	Пароль к приватному ключу в truststore	—
spark.ssl.trustStoreType	Тип используемого truststore	JKS
spark.ssl.enabled	Определяет, нужно ли задействовать SSL для Spark	—
spark.ssl.protocol	Определяет TLS-протокол. Указанный протокол должен поддерживаться JVM	TLSv1.2
spark.ssl.ui.port	Номер порта, на котором доступен веб-интерфейс Spark с включенным SSL	4141
spark.ssl.historyServer.port	Номер порта, на котором доступен веб-интерфейс Spark History Server с включенным SSL	18092

Custom log4j.properties

Параметр	**Описание**	**Значение по умолчанию**
Spark3 spark-log4j2.properties	Содержит конфигурацию Log4j2, которая используется Spark	spark-log4j2.properties
Livy livy-log4j.properties	Содержит конфигурацию Log4j, которая используется Livy	livy-log4j.properties

livy.conf

Параметр	Описание	Значение по умолчанию
livy.server.host	Хост, на котором запускается Livy-сервер. По умолчанию Livy слушает все сетевые интерфейсы	0.0.0.0
livy.server.port	Порт, на котором запускается Livy-сервер	8999
livy.spark.master	Spark master для Livy-сессий	yarn
livy.impersonation.enabled	Определяет, следует ли Livy имперсонировать пользователей при создании новой сессии	true
livy.server.csrf-protection.enabled	Определяет, задействовать ли CSRF-защиту. Если защита включена, клиентские приложения должны добавлять HTTP-заголовок `X-Requested-By` для HTTP-методов POST/DELETE/PUT/PATCH	true
livy.repl.enable-hive-context	Определяет, задействовать ли HiveContext для интерпретатора Livy. Если указано значение `true`, *hive-site.xml* и classpath Livy-сервера будут определены автоматически по запросу пользователя	true
livy.server.recovery.mode	Устанавливает режим восстановления для Livy	recovery
livy.server.recovery.state-store	Определяет, где Livy хранит состояние (state) для восстановления	filesystem
livy.server.recovery.state-store.url	Если выбран тип хранилища состояния `filesystem`, определяет путь к каталогу хранилища. Не рекомендуется использовать файловые системы, которые не поддерживают атомарное переименование (например, S3). Например: `file:///tmp/livy` or `hdfs:///`. Для ZooKeeper необходимо указать адрес ZooKeeper-серверов. Например: `host1:port1,host2:port2`	/livy-recovery
livy.server.auth.type	Указывает тип аутентификации для Livy	—
livy.server.access_control.enabled	Определяет, включать ли контроль доступа для сервера Livy. Если установлено значение `true`, все входящие запросы будут проверяться на наличие у запрошенного пользователя разрешения	false
livy.server.access_control.users	Пользователи, которым разрешен доступ к Livy. По умолчанию любому пользователю разрешен доступ к Livy. Если пользователь хочет ограничить доступ, он должен перечислить всех разрешенных пользователей через запятую	livy,hdfs,spark
livy.superusers	Список разделенных запятыми пользователей, у которых есть разрешения на изменение сеанса другого пользователя, например на выполнение запросов, удаление сеанса и так далее	livy,hdfs,spark
livy.keystore	Путь к keystore-файлу. Путь может быть как абсолютным, так и относительным каталогу, в котором был запущен процесс	—
livy.keystore.password	Пароль для доступа к keystore-файлу	—
livy.key-password	Пароль для доступа к ключу в keystore-файле	—
livy.server.thrift.ssl.protocol.blacklist	Список запрещенных к использованию TLS-протоколов	SSLv2,SSLv3,TLSv1,TLSv1.1

Spark heap memory settings

Параметр	Описание	Значение по умолчанию

Spark History Server Heap Memory	Максимальный размер области Java heap для Spark History Server	1G
Spark3 Connect Heap Memory	Максимальный размер области Java heap для сервера Spark Connect	1G

ranger-spark-audit.xml

Параметр	Описание	Значение по умолчанию
xasecure.audit.destination.solr.batch.filespool.dir	Путь к Spool-каталогу	/srv/ranger/hdfs_plugin/audit_solr_spool
xasecure.audit.destination.solr.urls	URL сервера Solr для сохранения аудит-событий. Оставьте значение пустым или установите значение `NONE` при использовании ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.zookeepers	Указывает строку подключения ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.force.use.inmemory.jaas.config	Использует in-memory JAAS-файл для соединения с Solr	—
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.jaas.Client.loginModuleControlFlag	Определяет требование к успешному завершению работы модуля. Значения могут быть `required`, `requisite`, `sufficient` или `optional`	—
xasecure.audit.jaas.Client.loginModuleName	Имя класса-аутентификатора	—
xasecure.audit.jaas.Client.option.keyTab	Имя keytab-файла, из которого необходимо получить секретный ключ принципала	—
xasecure.audit.jaas.Client.option.principal	Имя используемого принципала	—
xasecure.audit.jaas.Client.option.serviceName	Имя пользователя или сервиса, для которого выполнятся логин	—
xasecure.audit.jaas.Client.option.storeKey	Установите значение `true`, чтобы keytab-файл или ключ принципала сохранялись в учетных данных субъекта	false
xasecure.audit.jaas.Client.option.useKeyTab	Установите значение `true`, чтобы модуль получал ключ принципала из keytab-файла	false

ranger-spark-security.xml

Параметр	Описание	Значение по умолчанию
ranger.plugin.spark.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.spark.service.name	Имя Ranger-сервиса, который содержит политики для этого экземпляра	—
ranger.plugin.spark.policy.cache.dir	Каталог, в котором кешируются политики Ranger после успешного извлечения из источника	/srv/ranger/spark/policycache

ranger.plugin.spark.policy.pollIntervalMs	Определяет, как часто проверять политики на предмет изменения	30000
ranger.plugin.spark.policy.rest.client.connection.timeoutMs	Время ожидания подключения Spark-плагина RangerRestClient (в миллисекундах)	120000
ranger.plugin.spark.policy.rest.client.read.timeoutMs	Тайм-аут на чтение для Spark-плагина RangerRestClient (в миллисекундах)	30000
ranger.plugin.spark.policy.rest.ssl.config.file	Путь к конфигурационному файлу RangerRestClient SSL-плагина Spark	/etc/spark3/conf/ranger-spark-policymgr-ssl.xml

ranger-spark3-policymgr-ssl.xml

Параметр	Описание	Значение по умолчанию
xasecure.policymgr.clientssl.keystore	Путь к keystore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.credential.file	Путь к файлу с паролями для keystore-файла	/etc/spark3/conf/ranger-spark3.jceks
xasecure.policymgr.clientssl.truststore.credential.file	Путь к truststore-файлу с учетными данными	/etc/spark3/conf/ranger-spark3.jceks
xasecure.policymgr.clientssl.truststore	Путь к truststore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.password	Пароль для доступа к keystore-файлу	—
xasecure.policymgr.clientssl.truststore.password	Пароль для доступа к truststore-файлу	—

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom spark-defaults.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *spark-defaults.conf*	—
spark-env.sh	Содержимое файла *spark-env.sh*, который используется для инициализации переменных окружения на worker-узлах	spark-env.sh
Custom livy.conf	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *livy.conf*	—
livy-env.sh	Содержимое файла *livy-env.sh*, который используется для инициализации переменных окружения, необходимых для работы Spark3 Livy Server	livy-env.sh
spark-history-env.sh	Содержимое файла *spark-history-env.sh*, который используется для инициализации переменных окружения, необходимых для работы Spark3 History Server	spark-history-env.sh
Ranger plugin enabled	Включает или отключает плагин Ranger	false

SSM

Credentials Encryption

Параметр	Описание	Значение по умолчанию
Encryption enable	Установите значение `true` чтобы включить шифрование учетных данных	false
Credential provider path	Путь к хранилищу ключей, используемому для шифрования учетных данных	jceks://file/etc/ssm/conf/ssm.jceks
Custom jceks	Установите значение `true` для использования пользовательского JCEKS-хранилища по заданному пути. Установите значение `false`, чтобы использовать автоматически сгенерированное хранилище ключей JCEKS	false

smart-site.xml

Параметр	Описание	Значение по умолчанию
smart.hadoop.conf.path	Путь к директории конфигурации Hadoop	/etc/hadoop/conf
smart.conf.dir	Путь к директории конфигурации SSM	/etc/ssm/conf
smart.server.rpc.address	RPC-адрес сервера SSM	0.0.0.0:7042
smart.file.access.count.aggregator.failover	Стратегия отказоустойчивости для агрегатора событий доступа к файлам. Возможные значения: `FAIL` — бросить исключение, отказоустойчивости нет. `SAVE_FAILED_WITH_RETRY` — сохранить все события доступа к файлам, вызвавшие исключение	SAVE_FAILED_WITH_RETRY
smart.server.http.address	HTTP-адрес сервера SSM (веб-интерфейс)	0.0.0.0:7045
smart.agent.master.address	Адрес активного SSM-сервера	<hostname>
smart.agent.address	Определяет адрес компонента SSM Agent на каждом хосте	0.0.0.0
smart.agent.port	Номер порта, используемый агентами SSM для общения с SSM-сервером	7048
smart.agent.master.port	Номер порта, используемый SSM-сервером для общения с агентами SSM	7051
smart.rest.server.port	Порт REST-сервера SSM	7045
smart.rest.server.security.enabled	Параметр включает или отключает безопасность REST-сервера SSM	false
smart.rest.server.auth.spnego.enabled	Параметр включает или отключает аутентификацию SPNEGO для REST-сервера SSM	false
smart.rest.server.auth.predefined.enabled	Параметр включает или отключает базовую аутентификацию для пользователей из списка опции `smart.rest.server.auth.predefined.users`	false
smart.rest.server.auth.predefined.users	Список пользователей с их учетными данными, которые имеют доступ к REST-серверу SSM	—
smart.ignore.dirs	Список HDFS-директорий, разделенных запятыми, которые необходимо игнорировать. SSM игнорирует все файлы в указанных	—

	директориях	
smart.cover.dirs	Список HDFS-директорий, разделенных запятыми, в которых SSM будет сканировать файлы. По умолчанию охватываются все файлы HDFS	—
smart.work.dir	Директория HDFS, используемая SSM в качестве рабочей директории для хранения временных файлов. В этой директории SSM будет игнорировать события `inotify` для файлов. Можно указать только одну директорию	/system/ssm
smart.client.concurrent.report.enabled	Включает/отключает параллельные отчеты для Smart Client. Если опция включена, в целях оптимизации Smart Client пытается параллельно подключиться к нескольким настроенным Smart Server для определения активного сервера. Только активный Smart Server ответит на запрос для установления соединения. Если отчет был успешно доставлен на активный Smart Server, попытки подключения к другим серверам отменяются	—
smart.server.rpc.handler.count	Количество RPC-обработчиков на сервере	80
smart.namespace.fetcher.batch	Размер пакета для namespace-сборщика. SSM получает информацию о namespace от NameNode во время запуска. Большие размеры namespace могут привести к увеличению времени запуска. Увеличение размера пакета может ускорить работу сборщика и уменьшить время запуска	500
smart.namespace.fetcher.producers.num	Количество производителей (producers) в namespace-сборщике	3
smart.namespace.fetcher.consumers.num	Количество потребителей (consumers) в namespace-сборщике	6
smart.rule.executors	Максимальное количество правил, которые могут выполняться параллельно	5
smart.cmdlet.executors	Максимальное количество команд, которые могут выполняться параллельно	10
smart.dispatch.cmdlets.extra.num	Количество дополнительных команд, отправляемых Smart Server	10
smart.cmdlet.dispatchers	Максимальное количество диспетчеров команд, работающих параллельно	3
smart.cmdlet.mover.max.concurrent.blocks.per.srv.inst	Максимальное количество команд file mover для одного сервиса SSM, которые могут выполняться параллельно. Значение `0` снимает ограничение	0
smart.action.move.throttle.mb	Предельная пропускная способность (в МБ) для действия move overall	0
smart.action.copy.throttle.mb	Предельная пропускная способность (в МБ) для действия copy overall	0
smart.action.ec.throttle.mb	Предельная пропускная способность (в МБ) для действия EC overall	0
smart.action.local.execution.disabled	Определяет, может ли активный Smart Server также выполнять функции агента. Если установлено значение `true`, активный сервер SSM не сможет выполнять действия агента. Этот параметр не влияет на поведение резервного Smart Server	false
smart.cmdlet.max.num.pending	Максимальное количество ожидающих команд в SSM Server	20000
smart.cmdlet.hist.max.num.records	Максимальное количество исторических записей команд, хранящихся на сервере SSM. При превышении этого значения SSM удаляет самые старые команды	100000

smart.cmdlet.hist.max.record.lifetime	Максимальное время жизни исторических записей команд, хранящихся на сервере SSM. SSM-сервер удаляет записи команд по истечении указанного интервала. Допустимые единицы времени: `day`, `hour`, `min`, `sec`. Минимальная гранулярность обновления составляет `5sec`	30day
smart.cmdlet.cache.batch	Максимальный размер для пакетной вставки команд	600
smart.copy.scheduler.base.sync.batch	Максимальный размер пакета базовой синхронизации для Copy Scheduler	500
smart.file.diff.max.num.records	Максимальное количество записей файловых изменений с состоянием useless	10000
smart.status.report.period	Период между статус-отчетами действий в миллисекундах	10
smart.status.report.period.multiplier	Период отчета, умноженный на данное значение, определяет наибольший интервал отчета	50
smart.status.report.ratio	Если соотношение завершенных действий равно или превышает это значение, будет сгенерирован статус-отчет	0.2
smart.top.hot.files.num	Топ "горячих" файлов, отображаемых в веб-интерфейсе	200
smart.cmdlet.dispatcher.log.disp.result	Определяет, нужно ли регистрировать результат диспетчеризации для каждой команды	false
smart.cmdlet.dispatcher.log.disp.metrics.interval	Интервал времени в миллисекундах для регистрации статистических показателей диспетчера команд. Если в течение этого интервала не было отправлено ни одной команды, вывод для этого интервала отсутствует. Значение `0` отключает логирование	5000
smart.compression.codec	Устанавливает кодек сжатия по умолчанию (Zlib, Lz4, Bzip2, snappy). Также кодеки можно указать в качестве аргументов действия. Указание кодека в аргументе действия имеет более высокий приоритет, чем данный параметр	Zlib
smart.compression.max.split	Максимальное количество фрагментов, разделяемых для сжатия	1000
smart.compact.batch.size	Максимальное количество маленьких файлов (small files), которые будут скомпонованы при выполнении действия compact	200
smart.compact.container.file.threshold.mb	Максимальный размер файла контейнера в МБ при выполнении действия compact	1024
smart.access.count.day.tables.num	Максимальное количество таблиц, которые могут быть созданы в базе данных Metastore для хранения количества обращений к файлам в день	30
smart.access.count.hour.tables.num	Максимальное количество таблиц, которые могут быть созданы в базе данных Metastore для хранения количества обращений к файлам в час	48
smart.access.count.minute.tables.num	Максимальное количество таблиц, которые могут быть созданы в базе данных Metastore для хранения количества обращений к файлам в минуту	120
smart.access.count.second.tables.num	Максимальное количество таблиц, которые могут быть созданы в базе данных Metastore для хранения количества обращений к файлам в секунду	30
smart.access.event.fetch.interval.ms	Интервал в миллисекундах между получением данных о доступах к файлам	1000
smart.cached.file.fetch.interval.ms	Интервал в миллисекундах между получением кешированных файлов из HDFS	5000

smart.namespace.fetch.interval.ms	Интервал в миллисекундах между получением данных о пространствах имен из HDFS	1
smart.mover.scheduler.storage.report.fetch.interval.ms	Интервал в миллисекундах между получением отчетов о хранении из HDFS DataNodes в планировщике перемещений (mover scheduler)	120000
smart.metastore.small-file.insert.batch.size	Максимальный размер пакета вставки Metastore с информацией о маленьких файлах	200
smart.agent.master.ask.timeout.ms	Максимальное время в миллисекундах для ожидания ответа компонентом Smart Agent от Smart Server при отправке действия	5000
smart.ignore.path.templates	Список regex-шаблонов путей HDFS, которые SSM должен игнорировать	—
smart.internal.path.templates	Список regex-шаблонов внутренних файлов, которые SSM должен игнорировать	.*/\..*,.*/__.*,.*_COPYING_.*
smart.security.enable	Включает Kerberos-аутентификацию для SSM	false
smart.server.keytab.file	Путь к keytab-файлу сервера SSM	—
smart.server.kerberos.principal	Kerberos-принципал сервера SSM	—
smart.agent.keytab.file	Путь к keytab-файлу агента SSM	—
smart.agent.kerberos.principal	Kerberos-принципал агента SSM	—
smart.rest.server.auth.spnego.principal	Kerberos-принципал для SSM REST Server	—
smart.rest.server.auth.spnego.keytab	Keytab REST-сервера SSM	—
smart.proxy.user.strategy	Область поиска пользователей в LDAP. Возможные значения: ▪ `DISABLED` — имперсонация отключена, все действия выполняются от имени пользователя ноды SSM (либо принципала Kerberos, либо пользователя, запустившего SSM). ▪ `NODE_SCOPE` — имперсонация включена на уровне ноды, все действия выполняются от имени пользователя, указанного в параметре `smart.proxy.user`. ▪ `CMDLET_SCOPE` — имперсонация включена на уровне cmdlet, все действия выполняются от имени владельца cmdlet (в настоящее время — это создатель cmdlet).	DISABLED
smart.proxy.users.cache.ttl	Минимальное время, которое должно пройти после последнего обращения к записи кеша прокси-пользователей, прежде чем она будет удалена. Значение должно быть указано в формате `[Amount][TimeUnit]`, где `Amount` — число, а `TimeUnit` — одно из следующих значений: ▪ `day` или `d` — для дней; ▪ `hour` или `h` — для часов; ▪ `min` или `m` — для минут; ▪ `sec` или `s` — для секунд.	2m
smart.proxy.users.cache.size	Максимальный размер кеша прокси-пользователей	20

smart-env.sh

Параметр	Описание	Значение по умолчанию
LD_LIBRARY_PATH	Путь к дополнительным нативным библиотекам для SSM	/usr/lib/hadoop/lib/native
HADOOP_HOME	Путь к домашней директории Hadoop	/usr/lib/hadoop

Другие параметры

Параметр	Описание	Значение по умолчанию
Enable SmartFileSystem for Hadoop	Если эта опция включена, при расчете `AccessCount` для файлов учитываются запросы от разных клиентов (Spark, HDFS, Hive и так далее). В противном случае значение `AccessCount` увеличивается только при обращении к файлу из SSM	false
log4j.properties	Содержимое конфигурационного файла *log4j.properties*	—
Custom smart-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *smart-site.xml*	—
Custom smart-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *smart-env.sh*	—

Компонент SSM Server

Druid configuration

Параметр	Описание	Значение по умолчанию
db_url	URL к базе данных Metastore	jdbc:postgresql://{{ groups['adpg.adpg'][0] | d(omit) }}:5432/ssm
db_user	Имя пользователя для подключения к базе данных	ssm
db_password	Пароль пользователя для подключения к базе данных	—
initialSize	Начальное количество соединений, создаваемых при старте пула	10
minIdle	Минимальное количество установленных соединений, которое должно постоянно храниться в пуле. Пул соединений может уменьшиться ниже этого значения, если запросы валидации не проходят проверку	4
maxActive	Максимальное количество активных соединений, которые могут быть выделены из этого пула одновременно	50
maxWait	Максимальное время в миллисекундах, которое пул будет ожидать (при отсутствии доступных соединений), пока соединение не будет возвращено, прежде чем бросить исключение	60000
timeBetweenEvictionRunsMillis	Время в миллисекундах для ожидания между запусками потоков idle connection validation/cleaner. Это значение не должно быть меньше 1 секунды. Оно определяет, как часто следует проверять бездействующие и заброшенные (abandoned) соединения	90000
minEvictableIdleTimeMillis	Минимальное время, в течение которого объект соединения может бездействовать в пуле, прежде чем его можно будет удалить	300000
validationQuery	SQL-запрос, используемый для проверки соединений из пула перед возвратом их вызывающей стороне	SELECT 1

testWhileIdle	Определяет, проверять ли объекты с помощью idle object evictor (если таковой имеется)	true
testOnBorrow	Определяет, проверяются ли объекты перед заимствованием из пула	false
testOnReturn	Определяет, проверяются ли объекты перед возвращением в пул	false
poolPreparedStatements	Активирует пуллинг для подготовленных выражений (prepared statements)	true
maxPoolPreparedStatementPerConnectionSize	Максимальное количество подготовленных выражений, которые можно объединить в пул для одного соединения	30
removeAbandoned	Удаляет брошенные соединения, если они превысили `removeAbandonedTimeout`	true
removeAbandonedTimeout	Тайм-аут в секундах, после которого заброшенное (используемое) соединение может быть удалено	180
logAbandoned	Флаг для логирования стектрейса кода приложения, которое забросило соединение. Логирование брошенных соединений накладывает дополнительные расходы для каждого заимствованного соединения	true
filters	Устанавливает фильтры, применяемые к источнику данных	stat

Trino

Hive configuration

Параметр	Описание	Значение по умолчанию
connector.name	Тип коннектора	hive
hive.metastore.uri	URI для сервиса Hive Metastore	—
hive.storage-format	Формат хранения данных Hive	PARQUET
hive.compression-codec	Кодек компрессии для данных Hive	SNAPPY
hive.metastore.thrift.impersonation.enabled	Определяет, используется ли механизм имперсонации Thrift для запросов к Hive Metastore	true
fs.hadoop.enabled	Определяет, поддерживается ли доступ к HDFS	false
hive.config.resources	Необязательный список конфигурационных файлов HDFS, разделенных запятыми	/etc/hadoop/conf/core-site.xml,/etc/hadoop/conf/hdfs-site.xml
hive.non-managed-table-writes-enabled	Определяет, включена ли возможность записи во внешние таблицы Hive	true
hive.metastore.authentication.type	Определяет, используется ли аутентификация Kerberos для доступа к Hive Metastore	—
hive.metastore.service.principal	Принципал Kerberos для сервиса Hive Metastore	—
hive.metastore.client.principal	Принципал Kerberos для клиента Trino при подключении к Hive Metastore	—
hive.metastore.client.keytab	Путь к файлу keytab для Kerberos-клиента Trino	—

hive.hdfs.authentication.type	Определяет, используется ли Kerberos-аутентификация при доступе к HDFS	—
hive.hdfs.impersonation.enabled	Определяет, используется ли механизм имперсонации для конечных пользователей при осуществлении доступа к HDFS	false
hive.hdfs.trino.principal	Принципал Kerberos для клиента Trino при подключении к Hive Metastore	—
hive.hdfs.trino.keytab	Путь к файлу keytab для Kerberos-клиента Trino	—
hive.metastore.thrift.client.ssl.enabled	Определяет, используется ли SSL для Thrift-клиента Hive Metastore	false
hive.metastore.thrift.client.ssl.key	Путь к keystore для SSL-аутентификации	—
hive.metastore.thrift.client.ssl.key-password	Пароль для keystore, используемый при SSL-аутентификации	—
hive.metastore.thrift.client.ssl.trust-certificate	Путь к truststore для SSL-аутентификации	—
hive.metastore.thrift.client.ssl.trust-certificate-password	Пароль для truststore, используемый при SSL-аутентификации	—

Iceberg configuration

Параметр	Описание	Значение по умолчанию
connector.name	Тип коннектора	iceberg
hive.metastore.uri	URI для сервиса Hive Metastore	—
iceberg.catalog.type	Тип каталога для Iceberg	hive_metastore
iceberg.file-format	Формат файлов для таблиц Iceberg	PARQUET
iceberg.compression-codec	Кодек компрессии для таблиц Iceberg	SNAPPY
hive.metastore.thrift.impersonation.enabled	Определяет, используется ли механизм имперсонации Thrift для запросов к Hive Metastore	true
fs.hadoop.enabled	Определяет, поддерживается ли доступ к HDFS	false
hive.config.resources	Необязательный список конфигурационных файлов HDFS, разделенных запятыми	/etc/hadoop/conf/core-site.xml,/etc/hadoop/conf/hdfs-site.xml
hive.metastore.authentication.type	Определяет, используется ли аутентификация Kerberos для доступа к Hive Metastore	—
hive.metastore.service.principal	Принципал Kerberos для сервиса Hive Metastore	—
hive.metastore.client.principal	Принципал Kerberos для клиента Trino при подключении к Hive Metastore	—
hive.metastore.client.keytab	Путь к файлу keytab для Kerberos-клиента Trino	—
hive.hdfs.authentication.type	Определяет, используется ли Kerberos-аутентификация при доступе к HDFS	—

hive.hdfs.impersonation.enabled	Определяет, используется ли механизм имперсонации для конечных пользователей при осуществлении доступа к HDFS	false
hive.hdfs.trino.principal	Принципал Kerberos для клиента Trino при подключении к Hive Metastore	—
hive.hdfs.trino.keytab	Путь к файлу keytab для Kerberos-клиента Trino	—
hive.metastore.thrift.client.ssl.enabled	Определяет, используется ли SSL для Thrift-клиента Hive Metastore	false
hive.metastore.thrift.client.ssl.key	Путь к файлу keystore для SSL-аутентификации	—
hive.metastore.thrift.client.ssl.key-password	Пароль для файла keystore, используемый при SSL-аутентификации	—
hive.metastore.thrift.client.ssl.trust-certificate	Путь к файлу truststore для SSL-аутентификации	—
hive.metastore.thrift.client.ssl.trust-certificate-password	Пароль для файла truststore, используемый при SSL-аутентификации	—

Custom catalogs

Параметр	**Описание**	**Значение по умолчанию**
Connector properties	В этой секции можно добавлять кастомные каталоги, нажимая ***+1*** и вводя свойства для нового коннектора	—

resource-groups.properties

Параметр	**Описание**	**Значение по умолчанию**
resource-groups.configuration-manager	Источник конфигурации для менеджера групп ресурсов: файл JSON или база данных	file
resource-groups.config-db-url	URL базы данных, из которой следует загружать конфигурацию	—
resource-groups.config-db-user	Пользователь базы данных, используемый для подключения	—
resource-groups.config-db-password	Пароль пользователя базы данных, используемый для подключения	—
resource-groups.config-file	Путь к конфигурационному файлу JSON для менеджера групп ресурсов	/etc/trino/conf/resource-groups.json
resource-groups.max-refresh-interval	Максимальный период времени, в течение которого кластер будет продолжать прием запросов после неудачной попытки обновления конфигурации. По истечении этого периода конфигурация станет недействительной	—
resource-groups.refresh-interval	Интервал времени (в секундах) между загрузками конфигурации	—

session-property-config.properties

Параметр	Описание	Значение по умолчанию
Session property managers	Источник конфигурации для менеджеров свойств сессии	file
session-property-manager.config-file	Путь к конфигурационному файлу JSON для менеджеров свойств сессии	/etc/trino/conf/session-property-config.json

Fault-tolerant execution

Параметр	Описание	Значение по умолчанию
retry-policy	Определяет, должен ли Trino в случае неудачного выполнения перезапустить весь запрос (`QUERY`), его отдельные стадии (`TASK`), или не делать ничего (`NONE`). Данный параметр принимает следующие значения: - `NONE` — отключает режим FTE (значение по умолчанию). - `QUERY` — если Trino worker завершает выполнение запроса с ошибкой, Trino повторно выполняет весь запрос (query). Данная политика рекомендуется, когда большинство запросов Trino небольшие. - `TASK` — если Trino worker завершает выполнение запроса с ошибкой, Trino повторно выполняет отдельные задачи (tasks). Данный режим оптимален для больших пакетных запросов, однако может привести к повышенной задержке при большом количестве маленьких запросов.	NONE
exchange.deduplication-buffer-size	Размер внутреннего буфера координатора, используемого для хранения выходных данных стадий запросов при обеспечении отказоустойчивого исполнения	32MB
fault-tolerant-execution-exchange-encryption-enabled	Определяет, использовать ли шифрование для данных спулов	false
task.low-memory-killer.policy	Определяет, завершать ли принудительно стадии запросов в случае недостаточного количества памяти. Принимает одно из следующих значений: - `NONE` — не завершать стадии. - `total-reservation-on-blocked-nodes` — завершать стадии, являющиеся частью запросов, которые используют наибольшее количество памяти на узлах, где память исчерпана. - `least-waste` — завершать стадии, являющиеся частью запросов, которые используют значительное количество памяти на узлах, где память исчерпана. При этой политике стадии, затратившие на исполнение наибольшее время, завершаются в последнюю очередь, чтобы выполненная ими работа не пропала напрасно. Этот параметр используется, только если параметр `retry-policy` имеет значение `TASK`	total-reservation-on-blocked-nodes
query-retry-attempts	Максимальное количество попыток, предпринимаемых Trino для повторного исполнения запроса, прежде чем он будет считаться неуспешным. Этот параметр используется, только если параметр `retry-policy` имеет значение `QUERY`	4
task-retry-attempts-per-task	Максимальное количество попыток, предпринимаемых Trino для повторного исполнения одной стадии прежде чем запрос, к которой она принадлежит, будет считаться неуспешным. Этот параметр используется, только если параметр `retry-policy` имеет значение `TASK`	4
retry-initial-delay	Минимальное время ожидания перед повторным исполнением запроса или стадии. Этот параметр используется, только если параметр `retry-policy` имеет значение `TASK` или `QUERY`	10s

retry-max-delay	Максимальное время ожидания перед повторным исполнением запроса или стадии. Время ожидания увеличивается после каждого последовательного неудачного исполнения запроса или стадии. Этот параметр используется, только если параметр `retry-policy` имеет значение `TASK` или `QUERY`	1m
retry-delay-scale-factor	Величина, на которую умножается время ожидания перед повторным исполнением стадии или запроса после неудачной попытки. Этот параметр используется, только если параметр `retry-policy` имеет значение `TASK` или `QUERY`	2

exchange-manager.properties

Параметр	**Описание**	**Значение по умолчанию**
exchange-manager.name	Тип хранилища для данных промежуточного обмена между стадиями распределенного запроса: HDFS или локальная файловая система	hdfs
exchange.base-directories	Список URI расположений, которые менеджер Exchange использует для хранения данных спулов, разделенных запятыми	hdfs://<adh_cluster_nameservice>/exchange-spooling-directory
hdfs.config.resources	Список путей к конфигурационным файлам HDFS, разделенных запятыми. Эти файлы должны существовать на всех узлах кластера Trino	/etc/hadoop/conf/core-site.xml, /etc/hadoop/conf/hdfs-site.xml
exchange.hdfs.auto-create-storage-dirs	Определяет, нужно ли автоматически создавать директории хранилищ HDFS при их отсутствии	false
exchange.hdfs.auto-create-storage-dirs-perms	Разрешения на доступ к автоматически созданным директориям хранилищ	755
exchange.sink-buffer-pool-min-size	Минимальный размер пула буферов для пространства обмена. Чем больше пул буферов, тем больше объем использованной памяти и выше степень параллельности записи	10
exchange.sink-buffers-per-partition	Количество буферов в одной секции в пуле буферов. Чем больше пул буферов, тем интенсивнее использование памяти и выше степень параллельности записи	2

log.properties

Параметр	**Описание**	**Значение по умолчанию**
io.trino	Глобальный уровень логирования для Trino	INFO
io.trino.server	Уровень логирования для серверных частей Trino	INFO

ranger-trino-audit.xml

Параметр	**Описание**	**Значение по умолчанию**
xasecure.audit.destination.solr.batch.filespool.dir	Путь к Spool-каталогу	/srv/ranger/trino_plugin/audit_solr_spool
xasecure.audit.destination.solr.urls	URL сервера Solr для сохранения аудит-событий. Оставьте значение пустым или установите значение `NONE` при использовании ZooKeeper для подключения к Solr	—

xasecure.audit.destination.solr.zookeepers	Указывает строку для подключения ZooKeeper к Solr	—
xasecure.audit.destination.solr.force.use.inmemory.jaas.config	Определяет, использовать ли in-memory JAAS-файл для соединения с Solr	—
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.jaas.Client.loginModuleControlFlag	Определяет требование к успешному завершению работы модуля. Возможные значения: `required`, `requisite`, `sufficient`, `optional`	—
xasecure.audit.jaas.Client.loginModuleName	Имя класса-аутентификатора	—
xasecure.audit.jaas.Client.option.keyTab	Имя keytab-файла, из которого необходимо получить секретный ключ принципала	—
xasecure.audit.jaas.Client.option.principal	Имя используемого принципала	—
xasecure.audit.jaas.Client.option.serviceName	Имя пользователя или сервиса, для которого выполняется вход	—
xasecure.audit.jaas.Client.option.storeKey	Установите значение `true`, чтобы keytab-файл или ключ принципала сохранялись в учетных данных субъекта	false
xasecure.audit.jaas.Client.option.useKeyTab	Установите значение `true`, чтобы модуль получал ключ принципала из keytab-файла	false
xasecure.audit.jaas.Client.option.useTicketCache	Установите значение `true`, чтобы модуль получал ключ принципала из кеша	—

ranger-trino-security.xml

Параметр	**Описание**	**Значение по умолчанию**
ranger.plugin.trino.access.cluster.name	Имя для идентификации кластера Trino. Оно записывается в логи аудита, создаваемые плагином	—
ranger.plugin.trino.super.users	Список имен пользователей, разделенных запятыми, которые имеют права суперпользователя	—
ranger.plugin.trino.super.groups	Список групп, разделенных запятыми. Пользователи из этих групп считаются суперпользователями	—
ranger.plugin.trino.use.rangerGroups	Определяет, извлекать ли матрицу соответствия пользователей группам из Apache Ranger	—
ranger.plugin.trino.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.trino.service.name	Имя сервиса Ranger, хранящего политики для данного экземпляра Trino	—
ranger.plugin.trino.policy.cache.dir	Директория, в которой хранятся политики Ranger после успешного извлечения из источника	/srv/ranger/trino/policycache
ranger.plugin.trino.policy.pollIntervalMs	Период проверки политик на	30000

	актуальность в миллисекундах	
ranger.plugin.trino.policy.rest.client.connection.timeoutMs	Тайм-аут соединения клиента Ranger REST в миллисекундах	120000
ranger.plugin.trino.policy.rest.client.read.timeoutMs	Тайм-аут чтения клиента Ranger REST в миллисекундах	30000
ranger.plugin.trino.enable.implicit.userstore.enricher	Определяет, использовать ли UserStoreEnricher для извлечения атрибутов пользователей и групп при использовани макросов или скриптов в фильтрах строк начиная с версии Ranger 2.3	true
ranger.plugin.trino.policy.rest.ssl.config.file	Путь к конфигурационному файлу RangerRestClient SSL-плагина Trino	/etc/hbase/conf/ranger-hbase-policymgr-ssl.xml
ranger.plugin.trino.ugi.initialize	Определяет, инициализировать ли идентификатор Kerberos, использованный для аутентификации на сервере Ranger Admin	true
ranger.plugin.trino.ugi.login.type	Тип логина. Используйте значение `keytab`	keytab
ranger.plugin.trino.ugi.keytab.principal	Принципал Kerberos	—
ranger.plugin.trino.ugi.keytab.file	Расположение keytab-файла	—

ranger-trino-policymgr-ssl.xml

Параметр	Описание	Значение по умолчанию
xasecure.policymgr.clientssl.keystore	Путь к keystore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.credential.file	Путь к файлу с паролями для keystore-файла	/etc/trino/conf/ranger-trino.jceks
xasecure.policymgr.clientssl.truststore.credential.file	Путь к truststore-файлу с учетными данными	/etc/trino/conf/ranger-trino.jceks
xasecure.policymgr.clientssl.truststore	Путь к truststore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.password	Пароль для доступа к keystore-файлу	—
xasecure.policymgr.clientssl.truststore.password	Пароль для доступа к truststore-файлу	—

access-control.properties

Параметр	Описание	Значение по умолчанию
access-control.name	Тип контроля доступа. Возможные значения: ▪ `default` — разрешены все операции, кроме имперсонации пользователей и запуска аккуратного выключения. ▪ `allow-all` — разрешены все операции.	default

	▪ `read-only` — разрешено чтение данных или метаданных, запись данных или метаданных запрещена. ▪ `file` — правила разрешений заданы в конфигурационном файле. ▪ `opa` — разрешения устанавливаются с помощью Open Policy Agent (OPA). ▪ `ranger` — разрешения устанавливаются с помощью политик Apache Ranger.	
ranger.service.name	Имя сервиса Ranger с политиками	—
ranger.plugin.config.resource	Список конфигурационных файлов плагина Ranger, разделенных запятыми. Относительные пути разрешаются динамически поиском в classpath	—
ranger.hadoop.config.resource	Список конфигурационных файлов Hadoop, разделенных запятыми. Относительные пути разрешаются динамически поиском в classpath	—
security.config-file	Путь к конфигурационному файлу с правилами доступа	/etc/trino/conf/rules.json
opa.policy.uri	URL OPA-сервера (open policy agent)	—

Прочее

Параметр	Описание	Значение по умолчанию
Custom Hive configuration	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурации Hive	—
Custom Iceberg configuration	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурации Iceberg	—
Custom resource-groups.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурации менеджера групп ресурсов	—
Custom session-property-config.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурации менеджеров свойств сессии	—
Custom exchange-manager.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурации менеджера Exchange	—
Custom log.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурации логирования	—
Custom ranger-trino-audit.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-trino-audit.xml*	—

Custom ranger-trino-security.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-trino-security.xml*	—
Custom ranger-trino-policymgr-ssl.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-trino-policymgr-ssl.xml*	—
Ranger plugin enabled	Определяет, использовать ли плагин Ranger для Trino	false
resource-groups.json	Файл JSON, содержащий конфигурацию менеджера групп ресурсов	resource-groups.json
session-property-config.json	Файл JSON, содержащий конфигурацию менеджеров свойств сессии	session-property-config.json
ad-runtime-utils	Файл конфигурации YAML, содержащий параметры установки Java: версию, путь установки и переменную среды	ad-runtime-utils.yml

Компонент Trino Coordinator

config.properties

Параметр	Описание	Значение по умолчанию
coordinator	Определяет, выступает ли данный узел как координатор	true
http-server.authentication.allow-insecure-over-http	Определяет, разрешены ли небезопасные HTTP-соединения при включенной аутентификации	false
node-scheduler.include-coordinator	Определяет, должен ли узел-координатор также планировать и выполнять запросы	false
http-server.http.port	HTTP-порт сервера	18188
discovery.uri	URI для сервиса обнаружения. Обычно — полное доменное имя (FQDN) с номером порта	—
join-distribution-type	Используемый тип распределенного JOIN	AUTOMATIC
redistribute-writes	Определяет, перераспределять ли данные перед записью	true
query.max-cpu-time	Максимальное количество процессорного времени, которое может быть использовано для выполнения запроса по всему кластеру. Запросы, выполнение которых превышает это ограничение, принудительно завершаются	—
query.max-memory-per-node	Максимальное количество памяти для выполнения запроса на узле Trino worker	—
query.max-memory	Максимальное количество памяти для выполнения запроса по всему кластеру	20GB
query.max-total-memory	Максимальное количество памяти для выполнения запроса по всему кластеру, включая отзываемую память	40GB
memory.heap-headroom-per-node	Количество памяти, оставляемое для резерва (буфер) в heap-памяти Java-машины для распределения, не отслеживаемого Trino	—
exchange.deduplication-buffer-size	Размер буфера, используемого для данных спулов при отказоустойчивом выполнении	—

query.client.timeout	Тайм-аут для работы кластера без связи с клиентским приложением прежде чем он завершит работу	5m
query.execution-policy	Алгоритм для организации обработки всех стадий запроса	phased
query.determine-partition-count-for-write-enabled	Определяет, выбирается ли количество секций на основе количества данных, прочитанных и обработанных запросом для запросов на запись	false
query.max-hash-partition-count	Максимальное количество секций для использования при обработке распределенных операций	100
query.min-hash-partition-count-for-write	Минимальное количество секций для использования при обработке распределенных операций в запросах на запись	50
query.max-writer-task-count	Максимальное количество задач, которые примут участие в записи данных во время выполнения запросов `INSERT`, `CREATE TABLE AS SELECT` и `EXECUTE`. Это ограничение применяется, только если параметры `redistribute-writes` или `scale-writers` имеют значение `true`	100
query.low-memory-killer.policy	Определяет режим принудительного завершения выполнения запросов при исчерпании памяти	total-reservation-on-blocked-nodes
query.max-execution-time	Максимальное время исполнения запроса в кластере перед его завершением. Это время не включает в себя анализ, планирование и нахождение в очереди	100d
query.max-length	Максимальное количество символов в тексте SQL-запроса. Запросы длиннее этого значения не обрабатываются и завершаются с ошибкой `QUERY_TEXT_TOO_LARGE`	1000000
query.max-planning-time	Максимальное количество времени для планирования исполнения запроса	10m
query.max-run-time	Максимальное количество времени для обработки запроса в кластере перед его завершением	100d
query.max-scan-physical-bytes	Максимальное количество байт, которое может быть сканировано запросом в процессе исполнения	—
query.remote-task.max-error-duration	Тайм-аут для удаленных (remote) задач, которым не удается связаться с Trino coordinator. Если координатор не получает никаких данных от удаленной задачи в течение этого времени, он считает задачу завершенной с ошибкой	1m
query.max-stage-count	Максимальное разрешенное количество стадий на один запрос	150
query.max-history	Максимальное количество запросов, хранимое в истории для предоставления статистики и другой информации, а также для отображения данных в веб-интерфейсе	100
query.min-expire-age	Минимальный возраст запроса в истории для того, чтобы он мог считаться неактуальным. Неактуальные запросы удаляются из буфера истории и становятся недоступными в веб-интерфейсе	15m
query.remote-task.enable-adaptive-request-size	Определяет, используется ли динамическая генерация сплитов серверных запросов, присылаемых задачами. Это может предотвратить ошибки исчерпания памяти для больших схем	true
query.remote-task.guaranteed-splits-per-task	Минимальное количество сплитов, которые должны быть назначены каждой удаленной задаче для обеспечения минимального количества работы у каждой из этих задач	3
query.remote-task.max-request-size	Максимальный размер одного запроса, создаваемого удаленной задачей. Для работы этого параметра необходимо, чтобы параметр `query.remote-task.enable-adaptive-request-size` имел значение `true`	8MB

query.remote-task.request-size-headroom	Определяет размер резервной памяти, которую необходимо выделять сверх размера данных запроса. Для работы этого параметра необходимо, чтобы параметр `query.remote-task.enable-adaptive-request-size` имел значение `true`	2MB
query.info-url-template	URL для перенаправления клиентов к альтернативному расположению информации запроса	—
retry-policy	Политика повторного запуска для отказоустойчивого выполнения	NONE
catalog.prune.update-interval	Интервал для сокращения отброшенных каталогов. Для работы этого параметра необходимо, чтобы параметр `catalog.management` имел значение `dynamic`. Отбрасывание каталога не прерывает использующие его исполняемые запросы, но делает его недоступным для новых запросов	5s
catalog.store	При значении этого параметра, равном `file`, создание и отбрасывание каталогов с использованием SQL-команд дообавляет и удаляет файлы свойств каталога на узле Trino coordinator. Для работы этого параметра необходимо, чтобы параметр `catalog.management` имел значение `dynamic`	—
sql.forced-session-time-zone	Определяет, устанавливать ли принудительно часовой пояс для всех обрабатываемых запросов в заданное значение и таким образом игнорировать часовой пояс клиента. Часовой пояс должен быть задан строкой	—
sql.default-catalog	Каталог по умолчанию для всех клиентов. Любая конфигурация каталога по умолчанию, предоставляемая клиентом, перезаписывает это значение	—
sql.default-schema	Схема по умолчанию для всех клиентов. Значением этого параметра должно являться имя схемы, допустимое для каталога по умолчанию. Любая конфигурация схемы по умолчанию, предоставляемая клиентом, перезаписывает это значение	—
sql.default-function-catalog	Каталог по умолчанию для хранилища пользовательских функций для всех клиентов	—
sql.default-function-schema	Схема по умолчанию для UDF-хранилища для всех клиентов	—
sql.path	Набор путей по умолчанию к функциям или табличным функциям в определенных каталогах и схемах. Пути определяются в формате `catalog_name.schema_name`	—
spill-enabled	Определяет, используется ли переброс памяти на диск во избежание превышения ограничений памяти для запроса	false
spiller-spill-path	Путь для переброса памяти на диск	—
spiller-max-used-space-threshold	Если для заданного пути переброса памяти процент использования дискового пространства превышает значение этого парамтра, то этот путь не будет использоваться для переброса	0.9
spiller-threads	Количество потоков переброса памяти. Увеличьте это значение, если его будет не хватать для насыщения диска, на который ведется переброс (например, при использовании RAID)	4
max-spill-per-node	Максимальный размер пространства, используемый всеми запросами на одном узле для переброса памяти. Это значение необходимо задавать только на узлах Trino worker	100GB
query-max-spill-per-node	Максимальный размер пространства, используемый одним запросом на одном узле для переброса памяти. Это значение необходимо задавать только на узлах Trino worker	100GB
aggregation-operator-unspill-memory-limit	Ограничение размера памяти, используемой для обратного переброса одним экземпляром оператора агрегации	4MB
spill-compression-codec	Кодек компрессии, используемый при перебросе страниц памяти на диск	NONE

spill-encryption-enabled	Установите значение `true` для этого параметра, чтобы создать случайный секретный ключ для каждого файла переброса с целью шифрования и расшифрования данных, передаваемых на диск	false
exchange.client-threads	Количество потоков, используемых Exchange-клиентами для извлечения данных из других узлов Trino nodes. Более высокое значение этого параметра может привести к увеличению производительности для больших кластеров или кластеров с большой многопоточностью. Слишком большие значения могут привести к падению производительности из-за переключений контекста и дополнительного использования памяти	25
exchange.concurrent-request-multiplier	Множитель, определяющий количество одновременных запросов относительно доступного размера буфера памяти	3
exchange.compression-codec	Кодек, используемый для сжатия и распаковки файлов при обмене данными между узлами и exchange-хранилищем при отказоустойчивом режиме исполнения	LZ4
exchange.data-integrity-verification	Действие при обнаружении нарушения целостности данных	ABORT
exchange.max-buffer-size	Размер буфера exchange-клиента, содержащего данные, полученные от остальных узлов перед началом их обработки	32MB
exchange.max-response-size	Максимальный размер ответа, возвращаемого из exchange-запроса	16MB
sink.max-buffer-size	Размер выходного буфера для данных задач, которые ожидают подбора задачами выше по плану исполнения	32MB
sink.max-broadcast-buffer-size	Размер широковещательного буфера для данных задач, которые ожидают подбора задачами выше по плану исполнения	200MB
task.concurrency	Локальная многопоточность по умолчанию для параллельных операторов (например, объединение и агрегация)	—
task.http-response-threads	Максимальное количество потоков, которые могут быть созданы для обработки HTTP-ответов	100
task.http-timeout-threads	Количество потоков, используемых для обработки тайм-аутов при создании HTTP-ответов	3
task.info-update-interval	Интервал времени для обновления информации о задаче, используемый планировщиком. Более высокие значения могут привести к снижению нагрузки на процессор координатора, но также и к неоптимальному планированию генерации сплитов	3s
task.max-drivers-per-task	Максимальное количество драйверов, которые задача может выполнять одновременно	2147483647
task.max-partial-aggregation-memory	Максимальный размер результата частичной агрегации для распределенных агрегаций	16MB
task.max-worker-threads	Количество потоков, используемых узлом worker для обработки сплитов	—
task.min-drivers	Целевое количество работающих leaf-сплитов на узле Trino worker	—
task.min-drivers-per-task	Минимальное количество драйверов, которые гарантированно будут выполняться одновременно для одной задачи при условии, что у задачи остаются сплиты для обработки	3
task.min-writer-count	Минимальное количество одновременных потоков на запись для одного запроса одного узла Trino worker при условии, что не используются предпочтительное секционирование и масштабирование задач записи	1

task.max-writer-count	Максимальное количество одновременных потоков на запись для одного запроса одного узла Trino worker при условии, что не используются предпочтительное секционирование и масштабирование задач записи	—
task.interrupt-stuck-split-tasks-enabled	Установите значение `true` для этого параметра, чтобы позволить Trino обнаруживать задачи и считать их завершенными с ошибкой, если они содержат сплиты, которые не выполняются и не завершаются	true
task.interrupt-stuck-split-tasks-warning-threshold	Ограничение на время выполнения сплитов, после которого выводится стек вызовов на эндпойнте */v1/maxActiveSplits* и генерируются метрики JMX	10m
task.interrupt-stuck-split-tasks-timeout	Тайм-аут блокировки потока обработки сплита, после которого Trino считает задачу завершенной с ошибкой	10m
task.interrupt-stuck-split-tasks-detection-interval	Интервал проверки наличия сплитов, чье время обработки превышает значение параметра `task.interrupt-stuck-split-tasks-timeout`	2m
use-preferred-write-partitioning	Определяет, включено ли использование предпочтительного секционирования записи. Если имеет значение `true`, то каждая секция записывается отдельным обработчиком	true
scale-writers	Определяет, используется ли масштабирование записи с помощью динамического увеличения количества задач на запись в кластере	true
task.scale-writers.enabled	Определяет, используется ли масштабирование количества одновременных обработчиков записи в одной задаче	true
writer-scaling-min-data-processed	Минимальный размер несжатых данных, которые должны быть обработаны прежде чем может быть добавлен новый обработчик записи	100MB
optimizer.dictionary-aggregation	Определяет, используется ли оптимизация для агрегации в справочниках	false
optimizer.optimize-hash-generation	Установите значение `true` для этого параметра, чтобы вычисление хеш-кодов для распределений, объединений и агрегаций происходило заранее в процессе исполнения, что позволяет впоследствии делиться результатами между операциями в запросе	false
optimizer.optimize-metadata-queries	Определяет, следует ли перемещать агрегацию после оператора `OUTER JOIN`, если агрегация находится перед ним, а все столбцы из внешней части оператора `JOIN` находится в операторе группировки	true
optimizer.distinct-aggregations-strategy	Режим оптимизации, используемый для множественных раздельных агрегаций	AUTOMATIC
optimizer.push-table-write-through-union	Определяет, использовать ли параллельную запись при использовании оператора `UNION ALL` в запросах, записывающих данные	true
optimizer.join-reordering-strategy	Режим переупорядочивания объединений. Значение `NONE` подразумевает использование того же порядка таблиц, в котором они перечислены в запросе	AUTOMATIC
optimizer.max-reordered-joins	Максимальное количество объединений, которые могут быть переупорядочены за один раз, если параметр `optimizer.join-reordering-strategy` имеет значение `cost-based`	8
optimizer.optimize-duplicate-insensitive-joins	Определяет, следует ли уменьшать количество строк, порождаемых при объединениях, если оптимизатор обнаруживает возможность пропуска строк в результате выполнения дублирующихся объединений	true
optimizer.use-exact-partitioning	Определяет, следует ли заново секционировать данные, если секционирование в стадии ранее по плану исполнения не соответствует в точности тому, что ожидает стадия далее по плану исполнения	false

optimizer.use-table-scan-node-partitioning	Определяет, следует ли при чтении таблиц использовать секционирование табличного узла, предоставленное коннектором	true
optimizer.table-scan-node-partitioning-min-bucket-to-task-ratio	Минимальное соотношение количества bucket и задач, при достижении или превышении которого используется секционирование табличных узлов для сканирования	0.5
optimizer.filter-conjunction-independence-factor	Масштаб допуска независимости для оценки селективности конъюнкции множественных предикатов	0.75
optimizer.join-multi-clause-independence-factor	Масштаб допуска независимости для оценки результатов объединения множественных операторов	0.25
optimizer.non-estimatable-predicate-approximation.enabled	Определяет, следует ли использовать оптимизатор на основе затрат для определения необходимости повторного секционирования результатов стадии, которые уже были секционированы	true
optimizer.join-partitioned-build-min-row-count	Минимальное количество строк на стороне сборки объединения, необходимое для использования секционированного поиска соединения	1000000
optimizer.min-input-size-per-task	Минимальный размер входных данных, необходимых для одной задачи	5GB
optimizer.min-input-rows-per-task	Минимальное количество входных строк, необходимых для одной задачи	10000000
log.annotation-file	Имя файла с дополнительными свойствами, содержащего аннотации для добавления к сообщениям в логе для TCP-вывода или файлового вывода в формате JSON, определенного параметрами `log.path` и `log.format`	—
log.path	Путь к лог-файлу, используемому Trino. Этот путь является относительным к директории данных, заданной как `var/log/server.log` скриптом запуска	—
log.max-size	Максимальный размер для лог-файла главного приложения	100MB
log.max-total-size	Максимальный общий размер лог-файлов главного приложения	1GB
log.compression	Формат компрессии, используемый для лог-файлов при ротации. Возможные значения: `GZIP` и `NONE`. При значении `NONE` компрессия не используется	GZIP
http-server.log.enabled	Определяет, используется ли логирование для HTTP-сервера	true
http-server.log.compression.enabled	Определяет, используется ли компрессия лог-файлов для HTTP-сервера	true
http-server.log.path	Путь к лог-файлу HTTP-сервера	/var/log/trino/http-request.log
http-server.log.max-history	Максимальное количество лог-файлов HTTP-сервера перед использованием ротации	15
http-server.log.max-size	Максимальный размер лог-файла HTTP-сервера	—
re2j.dfa-states-limit	Максимальное количество состояний для использования, когда RE2J собирает быстрый, но потенциально ресурсоемкий детерминированный конечный автомат (ДКА) для поиска соответствий регулярных выражений	2147483647
re2j.dfa-retries	Количество повторных попыток использования алгоритма ДКА RE2J по достижении ограничения количества состояний. После этого для всех последующих вводных данных для этого запроса он использует недетерминированный конечный автомат (НКА), который является более медленным, но менее ресурсоемким	5

http-server.authentication.type	Механизм аутентификации для пользовательского доступа к Trino	—
node.internal-address-source	Установите значение `FQDN` для этого параметра для обеспечения корректной работы и использования действительных DNS-имен в Kerberos	FQDN
access-control.config-files	Путь к файлу конфигурации контроля доступа	/etc/trino/conf/access-control.properties
password-authenticator.config-files	Файл конфигурации Password Authenticator	/etc/trino/conf/coordinator/password-authenticator.properties
web-ui.user	Имя пользователя для аутентификации в веб-интерфейсе	trino
internal-communication.shared-secret	Строка, используемая узлами Trino из одного кластера для аутентификации в нем	—
web-ui.authentication.type	Механизм аутентификации, используемый для пользовательского доступа к веб-интерфейсу	NONE

Kerberos Configuration

Параметр	Описание	Значение по умолчанию
http-server.authentication.krb5.service-name	Имя сервиса Kerberos для аутентификации	trino
http-server.authentication.krb5.principal-hostname	Сетевое имя хоста из принципала Kerberos	—
http-server.authentication.krb5.keytab	Путь к файлу keytab для Kerberos-аутентификации	/etc/security/keytabs/trino.service.keytab
http.authentication.krb5.config	Путь к файлу конфигурации Kerberos	/etc/krb5.conf
http-server.authentication.krb5.user-mapping.file	Файл, содержащий правила для соотнесения принципалов Kerberos с локальными именами пользователей	—

SSL Configuration

Параметр	Описание	Значение по умолчанию
http-server.http.enabled	Определяет, включен ли HTTP для HTTP-сервера	true
internal-communication.https.required	Определяет, используется ли SSL/TLS для всех внутренних коммуникаций	false
http-server.https.enabled	Определяет, используется ли HTTPS для HTTP-сервера	false
http-server.https.port	HTTPS-порт сервера	18188
http-server.https.keystore.path	Путь к файлу keystore для HTTPS	—
http-server.https.keystore.key	Пароль для keystore, используемый в HTTPS	—

http-server.https.truststore.path	Путь к файлу truststore для HTTPS	—
http-server.https.truststore.key	Пароль для truststore, используемый в HTTPS	—
internal-communication.https.keystore.path	Путь к файлу keystore для внутренней коммуникации в кластере Trino через HTTPS	—
internal-communication.https.keystore.key	Пароль для keystore для внутренней коммуникации в кластере Trino через HTTPS	—
internal-communication.https.truststore.path	Путь к файлу truststore для внутренней коммуникации в кластере Trino через HTTPS	—
internal-communication.https.truststore.key	Пароль для truststore для внутренней коммуникации в кластере Trino через HTTPS	—

Enable LDAP

Параметр	Описание	Значение по умолчанию
password-authenticator.name	Название реализации аутентификатора	ldap
ldap.url	URL LDAP-сервера	—
ldap.ssl.truststore.path	Пусть к PEM- или JKS-файлу keystore	—
ldap.user-bind-pattern	Пользовательская строка соединения LDAP для аутентификации по паролю. Должна содержать шаблон `${USER}` который заменяется на имя пользователя во время аутентификации по паролю. Может содержать несколько шаблонов, разделенных двоеточиями	${USER}@example.com
ldap.allow-insecure	Определяет, разрешено ли подключение к LDAP-серверу без TLS	false
ldap.user-base-dn	База поиска LDAP для пользователя, пытающегося подключиться к серверу. Пример: `OU=Peoples,DC=example,DC=com`	—
ldap.bind-dn	LDAP-имя пользователя, используемое Trino при создании запросов на членство в группе. Пример: `CN=admin,OU=CITY_OU,DC=domain`	—
ldap.bind-password	Пароль, используемый Trino при создании запросов на членство в группе	—
ldap.group-auth-pattern	Задает LDAP-запрос для авторизации членства в группе LDAP	(&(objectClass=person)(sAMAccountName=${USER})(memberof=CN=AuthorizedGroup,OU=Groups,DC=example,DC=com))

OAuth 2.0

Параметр	Описание	Значение по умолчанию
http-server.authentication.oauth2.issuer	URL сервера авторизации OAuth2	—
http-server.authentication.oauth2.client-id	Идентификатор клиента OAuth2 для Trino	—

http-server.authentication.oauth2.client-secret	Пароль клиента OAuth2 для аутентификации	—

JWT Auth

Параметр	Описание	Значение по умолчанию
http-server.authentication.jwt.key-file	Определяет либо URL сервиса JWKS, либо путь к файлу PEM или HMAC	—
http-server.authentication.jwt.required-issuer	Определяет строку, которая должна совпадать с полем автора JWT (`iss`)	—
http-server.authentication.jwt.required-audience	Определяет строку, которая должна совпадать с полем аудитории JWT (`aud`)	—
http-server.authentication.jwt.principal-field	Поле в JWT, идентифицирующее субъект	sub
http-server.authentication.jwt.user-mapping.pattern	Шаблон регулярного выражения для приведения всех имен пользователей к формату Trino для JWT-аутентификации	—
http-server.authentication.jwt.user-mapping.file	Файл JSON с правилами приведения пользователей для JWT-аутентификации	—

Certificate Auth

Параметр	Описание	Значение по умолчанию
http-server.authentication.certificate.user-mapping.pattern	Шаблон регулярного выражения для приведения всех имен пользователей к формату Trino для аутентификации на основе сертификатов	—
http-server.authentication.certificate.user-mapping.file	Путь к файлу JSON с правилами приведения пользователей для аутентификации на основе сертификатов	/etc/trino/conf/coordinator/user-mapping.json

Password-file Auth

Параметр	Описание	Значение по умолчанию
password-authenticator.name	Определяет название реализации аутентификатора	file
file.password-file	Путь к файлу пароля	—
file.refresh-period	Определяет, как часто нужно перезагружать файл пароля	5s
file.auth-token-cache.max-size	Максимальное количество аутентифицированных паролей в кеше	1000

Trino catalog management

Параметр	Описание	Значение по умолчанию
catalog.management	Тип менеджера каталогов	static
Postgres JDBC URL	URL для JDBC-подключения к хранилищу каталогов Postgres	jdbc:postgresql://{{ groups['adpg.adpg'][0] | d(omit) }}:5432/catalog

Postgres username	Имя пользователя хранилища каталогов Postgres для JDBC-подключения	catalog
Postgres password	Пароль для JDBC-подключения к хранилищу каталогов Postgres	—
Enable cleanup	Определяет, включена ли очистка каталогов старых версий	true
Encryption password	Пароль для шифрования свойств	—
RegEx pattern	Шаблон регулярного выражения для определения свойств, подлежащих шифрованию	—
Cleanup old version catalogs	Интервал очистки каталогов старых версий. Минимальное значение -- `10d`	30d
Time to live of old version catalogs	Время жизни (TTL) каталогов старых версий. Минимальное значение — `1d`	365d

node.properties

Параметр	Описание	Значение по умолчанию
node.environment	Название окружения, в котором функционирует узел. Должно совпадать для всех узлов кластера Trino	adh
node.id	Уникальный идентификатор узла	—
node.data-dir	Путь к директории для хранения данных узла	/srv/trino/data/coordinator
node.server-log-file	Путь к журнальному файлу сервера для узла	/var/log/trino/coordinator/server.log
node.launcher-log-file	Путь к журнальному файлу запуска для узла	/var/log/trino/coordinator/launcher.log

Прочее

Параметр	Описание	Значение по умолчанию
user-mapping.json	Содержимое файла *user-mapping.json*, определяющего регулярные выражения для приведения всех имен пользователей к формату Trino	user-mapping.json
env.sh	Содержимое файла *env.sh*, определяющего настройки окружения для Trino	env.sh
jvm.config	Содержимое файла *jvm.config*, определяющего настройки виртуальной машины Java для Trino	jvm.config
Custom password-authenticator.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *password-authenticator.properties*	—
Custom config.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *config.properties*	—
Custom node.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *node.properties*	—
Enable custom ulimits	Включение этой опции позволяет указать	

ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже

```
[Service]
DefaultLimitCPU=
DefaultLimitFSIZE=
DefaultLimitDATA=
DefaultLimitSTACK=
DefaultLimitCORE=
DefaultLimitRSS=
DefaultLimitNOFILE=
DefaultLimitAS=
DefaultLimitNPROC=
DefaultLimitMEMLOCK=
DefaultLimitLOCKS=
DefaultLimitSIGPENDING=
DefaultLimitMSGQUEUE=
DefaultLimitNICE=
DefaultLimitRTPRIO=
DefaultLimitRTTIME=
```

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)
DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Trino Worker

config.properties

Параметр	Описание	Значение по умолчанию
coordinator	Определяет, выступает ли данный узел как координатор	false
http-server.authentication.allow-insecure-over-http	Определяет, разрешены ли небезопасные HTTP-соединения при включенной аутентификации	true
http-server.http.port	HTTP-порт сервера	18189
discovery.uri	URI для сервиса обнаружения. Обычно — полное доменное имя (FQDN) с номером порта	—
join-distribution-type	Используемый тип распределенного JOIN	AUTOMATIC
redistribute-writes	Определяет, перераспределять ли данные перед записью	true
query.max-cpu-time	Максимальное количество процессорного времени, которое может быть использовано запросом по всему кластеру. Запросы, превышающие это ограничение, принудительно завершаются	1_000_000_000d
query.max-memory-per-node	Максимальное количество памяти, которое запрос может использовать на узле Trino worker	—
query.max-memory	Максимальное количество памяти, которое запрос может использовать по всему кластеру	20GB
query.max-total-memory	Максимальное количество памяти, которое запрос может использовать по всему кластеру, включая отзываемую память	40GB
memory.heap-headroom-per-node	Количество памяти, оставляемое для резерва (буфер) в heap-памяти Java-машины для распределения, не отслеживаемого Trino	—
exchange.deduplication-buffer-size	Размер буфера, используемого для данных спулов при отказоустойчивом выполнении	—
query.client.timeout	Тайм-аут для работы кластера без связи с клиентским приложением прежде чем он завершит работу	5m
query.execution-policy	Алгоритм для организации обработки всех стадий запроса	phased
query.determine-partition-count-for-write-enabled	Определяет, выбирается ли количество секций на количестве данных, которые были считаны и подготовлены запросом для запросов на запись	false
query.max-hash-partition-count	Максимальное количество секций для использования при обработке распределенных операций	100
query.min-hash-partition-count-for-write	Минимальное количество секций для использования при обработке распределенных операций в запросах на запись	50
query.max-writer-task-count	Максимальное количество задач, которые примут участие в записи данных во время запросов `INSERT`, `CREATE TABLE AS SELECT` и `EXECUTE`. Это ограничение применяется, только если включены параметры `redistribute-writes` или `scale-writers`	100
query.low-memory-killer.policy	Определяет режим принудительного завершения запросов при исчерпании памяти	total-reservation-on-blocked-nodes
query.max-execution-time	Максимальное время исполнения запроса в кластере перед его завершением. Это время не включает в себя анализ, планирование и нахождение в очереди	100d

query.max-length	Максимальное количество символов в тексте SQL-запроса. Запросы длиннее этого значения не обрабатываются и завершаются с ошибкой `QUERY_TEXT_TOO_LARGE`	1000000
query.max-planning-time	Максимальное количество времени для планирования исполнения запроса	10m
query.max-run-time	Максимальное количество времени для обработки запроса в кластере перед его завершением	100d
query.max-scan-physical-bytes	Максимальное количество байт, которое может быть сканировано запросом в процессе исполнения	—
query.remote-task.max-error-duration	Тайм-аут для удаленных (remote) задач, которым не удается связаться с Trino coordinator. Если координатор не получает никаких данных от удаленной задачи в течение этого времени, он считает задачу завершенной с ошибкой	1m
query.max-stage-count	Максимальное разрешенное количество стадий на один запрос	150
query.max-history	Максимальное количество запросов, хранимое в истории для предоставления статистики и другой информации, а также для отображения данных в веб-интерфейсе	100
query.min-expire-age	Минимальный возраст запроса в истории для того, чтобы он мог считаться неактуальным. Неактуальные запросы удаляются из буфера истории и становятся недоступными в веб-интерфейсе	15m
query.remote-task.enable-adaptive-request-size	Определяет, используется ли динамическая генерация сплитов серверных запросов, присылаемых задачами. Это может предотвратить ошибки исчерпания памяти для больших схем	true
query.remote-task.guaranteed-splits-per-task	Минимальное количество сплитов, которые должны быть назначены каждой удаленной задаче для обеспечения минимального количества работы у каждой из этих задач	3
query.remote-task.max-request-size	Максимальный размер одного запроса, создаваемого удаленной задачей. Для работы этого параметра необходимо, чтобы был включен параметр `query.remote-task.enable-adaptive-request-size`	8MB
query.remote-task.request-size-headroom	Определяет размер резервной памяти, которую необходимо выделять сверх размера данных запроса. Для работы этого параметра необходимо, чтобы был включен параметр `query.remote-task.enable-adaptive-request-size`	2MB
query.info-url-template	URL для перенаправления клиентов к альтернативному расположению информации запроса	—
retry-policy	Политика повторного запуска для отказоустойчивого выполнения	NONE
sql.forced-session-time-zone	Определяет, устанавливать ли принудительно часовой пояс для всех обрабатываемых запросов в заданное значение и таким образом игнорировать часовой пояс клиента. Часовой пояс должен быть задан строкой	—
sql.default-catalog	Каталог по умолчанию для всех клиентов. Любая конфигурация каталога по умолчанию, предоставляемая клиентом, перезаписывает это значение	—
sql.default-schema	Схема по умолчанию для всех клиентов. Значением этого параметра должно являться имя схемы, допустимое для каталога по умолчанию. Любая конфигурация схемы по умолчанию, предоставляемая клиентом, перезаписывает это значение	—
sql.default-function-catalog	Каталог по умолчанию для хранилища пользовательских функций для всех клиентов	—
sql.default-function-schema	Схема по умолчанию для UDF-хранилища для всех клиентов	—

sql.path	Набор путей по умолчанию к функциям или табличным функциям в определенных каталогах и схемах. Пути определяются в формате `catalog_name.schema_name`	—
spill-enabled	Определяет, используется ли переброс памяти на диск во избежание превышения ограничений памяти для запроса	false
spiller-spill-path	Путь для переброса памяти на диск	—
spiller-max-used-space-threshold	Если для заданного пути переброса памяти процент использования дискового пространства превышает значение этого парамтра, то этот путь не будет использоваться для переброса	0.9
spiller-threads	Количество потоков переброса памяти. Увеличьте это значение, если его будет не хватать для насыщения диска, на который ведется переброс (например, при использовании RAID)	4
max-spill-per-node	Максимальный размер пространства, используемый всеми запросами на одном узле для переброса памяти. Это значение необходимо задавать только на узлах Trino worker	100GB
query-max-spill-per-node	Максимальный размер пространства, используемый одним запросом на одном узле для переброса памяти. Это значение необходимо задавать только на узлах Trino worker	100GB
aggregation-operator-unspill-memory-limit	Ограничение размера памяти, используемой для обратного переброса одним экземпляром оператора агрегации	4MB
spill-compression-codec	Кодек компрессии, используемый при перебросе страниц памяти на диск	NONE
spill-encryption-enabled	Установите значение `true` для этого параметра, чтобы создать случайный секретный ключ для каждого файла переброса с целью шифрования и расшифрования данных, передаваемых на диск	false
exchange.client-threads	Количество потоков, используемых Exchange-клиентами для извлечения данных из других узлов Trino nodes. Более высокое значение этого параметра может привести к увеличению производительности для больших кластеров или кластеров с большой многопоточностью. Слишком большие значения могут привести к падению производительности из-за переключений контекста и дополнительного использования памяти	25
exchange.concurrent-request-multiplier	Множитель, определяющий количество одновременных запросов относительно доступного размера буфера памяти	3
exchange.compression-codec	Кодек, используемый для сжатия и распаковки файлов при обмене данными между узлами и exchange-хранилищем при отказоустойчивом режиме исполнения	LZ4
exchange.data-integrity-verification	Действие при обнаружении нарушения целостности данных	ABORT
exchange.max-buffer-size	Размер буфера exchange-клиента, содержащего данные, полученные от остальных узлов перед началом их обработки	32MB
exchange.max-response-size	Максимальный размер ответа, возвращаемого из exchange-запроса	16MB
sink.max-buffer-size	Размер выходного буфера для данных задач, которые ожидают подбора задачами выше по плану исполнения	32MB
sink.max-broadcast-buffer-size	Размер широковещательного буфера для данных задач, которые ожидают подбора задачами выше по плану исполнения	200MB
task.concurrency	Локальная многопоточность по умолчанию для параллельных операторов (например, объединение и агрегация)	—
task.http-response-threads	Максимальное количество потоков, которые могут быть созданы для обработки HTTP-ответов	100

task.http-timeout-threads	Количество потоков, используемых для обработки тайм-аутов при создании HTTP-ответов	3
task.info-update-interval	Интервал времени для обновления информации о задаче, используемый планировщиком. Более высокие значения могут привести к снижению нагрузки на процессор координатора, но также и к неоптимальному планированию генерации сплитов	3s
task.max-drivers-per-task	Максимальное количество драйверов, которые задача может выполнять одновременно	2147483647
task.max-partial-aggregation-memory	Максимальный размер результата частичной агрегации для распределенных агрегаций	16MB
task.max-worker-threads	Количество потоков, используемых узлом worker для обработки сплитов	—
task.min-drivers	Целевое количество работающих leaf-сплитов на узле Trino worker	—
task.min-drivers-per-task	Минимальное количество драйверов, которые гарантированно будут выполняться одновременно для одной задачи при условии, что у задачи остаются сплиты для обработки	3
task.min-writer-count	Минимальное количество одновременных потоков на запись для одного запроса одного узла Trino worker при условии, что не используются предпочтительное секционирование и масштабирование задач записи	1
task.max-writer-count	Максимальное количество одновременных потоков на запись для одного запроса одного узла Trino worker при условии, что не используются предпочтительное секционирование и масштабирование задач записи	—
task.interrupt-stuck-split-tasks-enabled	Установите значение `true` для этого параметра, чтобы позволить Trino обнаруживать задачи и считать их завершенными с ошибкой, если они содержат сплиты, которые не выполняются и не завершаются	true
task.interrupt-stuck-split-tasks-warning-threshold	Ограничение на время выполнения сплитов, после которого выводится стек вызовов на эндпойнте */v1/maxActiveSplits* и генерируются метрики JMX	10m
task.interrupt-stuck-split-tasks-timeout	Тайм-аут блокировки потока обработки сплита, после которого Trino считает задачу завершенной с ошибкой	10m
task.interrupt-stuck-split-tasks-detection-interval	Интервал проверки наличия сплитов, чье время обработки превышает значение параметра `task.interrupt-stuck-split-tasks-timeout`	2m
use-preferred-write-partitioning	Определяет, включено ли использование предпочтительного секционирования записи. При значении `true` каждая секция записывается отдельным обработчиком	true
scale-writers	Определяет, используется ли масштабирование записи с помощью динамического увеличения количества задач на запись в кластере	true
task.scale-writers.enabled	Определяет, используется ли масштабирование количества одновременных обработчиков записи в одной задаче	true
writer-scaling-min-data-processed	Минимальный размер несжатых данных, которые должны быть обработаны прежде чем может быть добавлен новый обработчик записи	100MB
node-scheduler.max-splits-per-node	Целевое значение общего количества сплитов, которые могут выполняться на каждом узле Trino worker, при условии, что все сплиты имеют стандартный вес	256

node-scheduler.min-pending-splits-per-task	Минимальное количество сплитов со стандартным весом, не находящихся в обработке, которые гарантированно будут распределены на узел (даже если узел уже достиг предела общего количества сплитов) для одной задачи, при условии, что у задачи остались сплиты для обработки	16
node-scheduler.max-adjusted-pending-splits-per-task	Максимальное количество сплитов со стандартным весом, не находящихся в обработке, которые гарантированно будут распределены на узел (даже если узел уже достиг предела общего количества сплитов) для одной задачи, при условии, что у задачи остались сплиты для обработки	2000
node-scheduler.max-unacknowledged-splits-per-task	Максимальное количество сплитов, которые либо стоят в очереди координатора, но еще не отправлены, либо получены Trino worker с подтверждением	2000
node-scheduler.min-candidates	Минимальное количество узлов-кандидатов, которые должны пройти оценку планировщиком при выборе узла для генерации сплита	10
node-scheduler.policy	Политика планировщика при выборе узлов для генерации сплита	uniform
optimizer.dictionary-aggregation	Определяет, используется ли оптимизация для агрегации в справочниках	false
optimizer.optimize-hash-generation	Установите значение `true` для этого параметра, чтобы вычисление хеш-кодов для распределений, объединений и агрегаций происходило заранее в процессе исполнения, что позволяет впоследствии делиться результатами между операциями в запросе	false
optimizer.optimize-metadata-queries	Определяет, следует ли перемещать агрегацию после оператора `OUTER JOIN`, если агрегация находится перед ним, а все столбцы из внешней части оператора `JOIN` находится в операторе группировки	true
optimizer.distinct-aggregations-strategy	Режим оптимизации, используемый для множественных раздельных агрегаций	AUTOMATIC
optimizer.push-table-write-through-union	Определяет, использовать ли параллельную запись при использовании оператора `UNION ALL` в запросах, записывающих данные	true
optimizer.join-reordering-strategy	Режим переупорядочивания объединений. Значение `NONE` подразумевает использование того же порядка таблиц, в котором они перечислены в запросе	AUTOMATIC
optimizer.max-reordered-joins	Максимальное количество объединений, которые могут быть переупорядочены за один раз, если параметр `optimizer.join-reordering-strategy` имеет значение `cost-based`	8
optimizer.optimize-duplicate-insensitive-joins	Определяет, следует ли уменьшать количество строк, порождаемых при объединениях, если оптимизатор обнаруживает возможность пропуска строк в результате выполнения дублирующихся объединений	true
optimizer.use-exact-partitioning	Определяет, следует ли заново секционировать данные, если секционирование в предыдущей стадии по плану исполнения не соответствует в точности тому, что ожидает следующая стадия по плану исполнения	false
optimizer.use-table-scan-node-partitioning	Определяет, следует ли при чтении таблиц использовать секционирование табличного узла, предоставленное коннектором	true
optimizer.table-scan-node-partitioning-min-bucket-to-task-ratio	Минимальное соотношение количества bucket и задач, при достижении или превышении которого используется секционирование табличных узлов для сканирования	0.5
optimizer.filter-conjunction-independence-factor	Масштаб допуска независимости для оценки селективности конъюнкции множественных предикатов	0.75

optimizer.join-multi-clause-independence-factor	Масштаб допуска независимости для оценки результатов объединения множественных операторов	0.25
optimizer.non-estimatable-predicate-approximation.enabled	Определяет, следует ли использовать оптимизатор на основе затрат для определения необходимости повторного секционирования результатов стадии, которые уже были секционированы	true
optimizer.join-partitioned-build-min-row-count	Минимальное количество строк на стороне сборки объединения, необходимое для использования секционированного поиска соединения	1000000
optimizer.min-input-size-per-task	Минимальный размер входных данных, необходимых для одной задачи	5GB
optimizer.min-input-rows-per-task	Минимальное количество входных строк, необходимых для одной задачи	10000000
log.annotation-file	Имя файла с дополнительными свойствами, содержащего аннотации для добавления к сообщениям в логе для TCP-вывода или файлового вывода в формате JSON, определенного параметрами `log.path` и `log.format`	—
log.path	Путь к лог-файлу, используемому Trino. Этот путь является относительным к директории данных, заданной как `var/log/server.log` скриптом запуска	—
log.max-size	Максимальный размер для лог-файла главного приложения	100MB
log.max-total-size	Максимальный общий размер лог-файлов главного приложения	1GB
log.compression	Формат компрессии, используемый для лог-файлов при ротации. Возможные значения: `GZIP` и `NONE`. При значении `NONE` компрессия не используется	GZIP
http-server.log.enabled	Определяет, используется ли логирование для HTTP-сервера	true
http-server.log.compression.enabled	Определяет, используется ли компрессия лог-файлов для HTTP-сервера	true
http-server.log.path	Путь к лог-файлу HTTP-сервера	/var/log/trino/http-request.log
http-server.log.max-history	Максимальное количество лог-файлов HTTP-сервера перед использованием ротации	15
http-server.log.max-size	Максимальный размер лог-файла HTTP-сервера	—
re2j.dfa-states-limit	Максимальное количество состояний для использования, когда RE2J собирает быстрый, но потенциально ресурсоемкий детерминированный конечный автомат (ДКА) для поиска соответствий регулярных выражений	2147483647
re2j.dfa-retries	Количество повторных попыток использования алгоритма ДКА RE2J по достижении ограничения количества состояний. После этого для всех последующих вводных данных для этого запроса он использует недетерминированный конечный автомат (НКА), который является более медленным, но менее ресурсоемким	5
node.internal-address-source	Установите значение `FQDN` для этого параметра для обеспечения корректной работы и использования действительных DNS-имен в Kerberos	FQDN
access-control.config-files	Путь к файлу конфигурации контроля доступа	/etc/trino/conf/access-control.properties
internal-communication.shared-secret	Строка, используемая координаторами и исполнителями одного и того же кластера для аутентификации внутри него	—

Kerberos Configuration

Параметр	Описание	Значение по умолчанию
http-server.authentication.krb5.service-name	Имя сервера Kerberos для аутентификации	HTTP
http-server.authentication.krb5.principal-hostname	Принципал имени узла для аутентификации Kerberos	—
http-server.authentication.krb5.keytab	Путь к файлу keytab для аутентификации Kerberos	/etc/security/keytabs/HTTP.service.keytab
http.authentication.krb5.config	Путь к конфигурационному файлу Kerberos	/etc/krb5.conf
http-server.authentication.krb5.user-mapping.file	Файл, содержащий правила для соотнесения принципалов Kerberos с локальными именами пользователей	—

SSL Configuration

Параметр	Описание	Значение по умолчанию
http-server.http.enabled	Определяет, включен ли HTTP для HTTP-сервера	true
internal-communication.https.required	Определяет, используется ли SSL/TLS для всех внутренних коммуникаций	false
http-server.https.enabled	Определяет, используется ли HTTPS для HTTP-сервера	false
http-server.https.port	HTTPS-порт сервера	18189
http-server.https.keystore.path	Путь к файлу keystore для HTTPS	—
http-server.https.keystore.key	Пароль для keystore, используемый в HTTPS	—
http-server.https.truststore.path	Путь к файлу truststore для HTTPS	—
http-server.https.truststore.key	Пароль для truststore, используемый в HTTPS	—
internal-communication.https.keystore.path	Путь к файлу keystore для внутренней коммуникации в кластере Trino через HTTPS	—
internal-communication.https.keystore.key	Пароль для keystore для внутренней коммуникации в кластере Trino через HTTPS	—
internal-communication.https.truststore.path	Путь к файлу truststore для внутренней коммуникации в кластере Trino через HTTPS	—
internal-communication.https.truststore.key	Пароль для truststore для внутренней коммуникации в кластере Trino через HTTPS	—

node.properties

Параметр	Описание	Значение по умолчанию
node.environment	Название окружения, в котором функционирует узел. Должно совпадать для всех узлов кластера Trino	adh
node.id	Уникальный идентификатор узла	—
node.data-dir	Путь к директории для хранения данных узла	/srv/trino/data/worker
catalog.config-dir	Путь к директории для конфигураций каталога	/etc/trino/conf/catalog
node.server-log-file	Путь к журнальному файлу сервера для узла	/var/log/trino/worker/server.log
node.launcher-log-file	Путь к журнальному файлу запуска для узла	/var/log/trino/worker/launcher.log

Прочее

Параметр	Описание	Значение по умолчанию
env.sh	Содержимое файла *env.sh*, определяющего настройки окружения для Trino	env.sh
jvm.config	Содержимое файла *jvm.config*, определяющего настройки виртуальной машины Java для Trino	jvm.config
Custom config.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *config.properties*	—
Custom node.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *node.properties*	—

Enable custom ulimits	Включение этой опции позволяет указать ограничения на использование ресурсов для процесса (ulimits). Если значения не указаны, будут использованы системные настройки по умолчанию. Настройки ulimit описаны в таблице ниже	`[Service]`
		`DefaultLimitCPU=`
		`DefaultLimitFSIZE=`
		`DefaultLimitDATA=`
		`DefaultLimitSTACK=`
		`DefaultLimitCORE=`
		`DefaultLimitRSS=`
		`DefaultLimitNOFILE=`
		`DefaultLimitAS=`
		`DefaultLimitNPROC=`
		`DefaultLimitMEMLOCK=`
		`DefaultLimitLOCKS=`
		`DefaultLimitSIGPENDING=`
		`DefaultLimitMSGQUEUE=`
		`DefaultLimitNICE=`
		`DefaultLimitRTPRIO=`
		`DefaultLimitRTTIME=`

Ulimit settings

Параметр	Описание	Соответствующая опция команды ulimit в CentOS
DefaultLimitCPU	Ограничение в секундах на количество процессорного времени, которое может потреблять процесс	cpu time (-t)
DefaultLimitFSIZE	Максимальный размер файлов, которые может создать процесс, в блоках по 512 байт	file size (-f)
DefaultLimitDATA	Максимальный размер сегмента данных процесса, в килобайтах	data seg size (-d)

DefaultLimitSTACK	Максимальный размер стека, выделенный процессу, в килобайтах	stack size (-s)
DefaultLimitCORE	Максимальный размер файла дампа (core dump) в байтах, который процесс может сохранить, в блоках по 512 байт	core file size (-c)
DefaultLimitRSS	Максимальное количество памяти RAM (resident set size), которое может быть выделено процессу, в килобайтах	max memory size (-m)
DefaultLimitNOFILE	Максимальное количество дескрипторов открытых файлов, разрешенное для процесса	open files (-n)
DefaultLimitAS	Максимальный размер виртуальной памяти (адресного пространства) процесса, в килобайтах	virtual memory (-v)
DefaultLimitNPROC	Максимальное количество процессов	max user processes (-u)
DefaultLimitMEMLOCK	Максимальный размер памяти, который может быть заблокирован для процесса, в килобайтах. Блокировка памяти гарантирует, что память всегда находится в оперативной памяти (RAM) и файл подкачки не используется	max locked memory (-l)
DefaultLimitLOCKS	Максимальное количество файлов, заблокированных процессом	file locks (-x)
DefaultLimitSIGPENDING	Максимальное количество сигналов, ожидающих доставки в вызывающий поток	pending signals (-i)
DefaultLimitMSGQUEUE	Максимальное количество байтов в очередях сообщений POSIX. Очереди сообщений POSIX позволяют процессам обмениваться данными в форме сообщений	POSIX message queues (-q)
DefaultLimitNICE	Максимальный уровень приоритета NICE, который может быть назначен процессу	scheduling priority (-e)
DefaultLimitRTPRIO	Максимальный приоритет планирования в реальном времени	real-time priority (-r)
DefaultLimitRTTIME	Максимальный размер буфера конвейера (pipe), в блоках по 512 байт	pipe size (-p)

Компонент Haproxy Trino

Параметр	Описание	Значение по умолчанию
haproxy-trino.cfg	Шаблон Jinja с конфигурацией HAProxy	haproxy-trino.cfg
Specifies the location of the certificate in standard .PEM format	Путь к сертификату в формате PEM	/etc/ssl/serverkey.pem
/etc/syslog-ng/conf.d/haproxy-trino.conf	Конфигурация syslog-ng для Haproxy Trino. Используется, если syslog-ng установлен в системе	haproxy-trino.conf
/etc/rsyslog.d/haproxy-trino.conf	Конфигурация rsyslog для Haproxy Trino. Используется, если rsyslog установлен в системе	haproxy-trino.conf

Компонент Trino Client

config.properties

Параметр	Описание	Значение по умолчанию
catalog	Каталог Trino по умолчанию	—
client-info	Дополнительная информация о клиенте, отправляющем запросы	—

client-request-timeout	Тайм-аут на выполнение запроса от клиента	2m
client-tags	Разделенный запятыми список тегов, предоставляющих дополнительную информацию о клиенте и пользователе	—
editing-mode	Режим редактирования. Возможные значения: `VI`, `EMACS`	EMACS
encoding	Устанавливает кодировку для протокола буферизации (spooling)	json+zstd,json+lz4,json
external-authentication-redirect-handler	Указывает обработчик редиректов для внешней аутентификации	ALL
extra-credential	Дополнительные учетные данные в формате ключ/значение	—
history-file	Путь к файлу с историей команд	${user.home}/.trino_history
network-logging	Уровень детализации сетевых логов клиента. Возможные значения: `NONE`, `BASIC`, `HEADERS`, `BODY`	NONE
output-format	Формат вывода для пакетной обработки (batch mode)	CSV
output-format-interactive	Формат вывода для интерактивной обработки	ALIGNED
pager	Путь к pager-программе, используемой для просмотра результатов запроса	less
path	Устанавливает дефолтный путь SQL на уровне сессии	—
schema	Схема по умолчанию. Указывается вместе с `catalog`	—
server	Расположение сервера Trino (Trino Coordinator или Trino Haproxy)	<TrinoCoordinatorHost> или <TrinoHaproxyHost>
session	Свойства сессии Trino в формате ключ/значение	—
source	Имя приложения или источника, подключающегося к Trino	trino-cli
timezone	Часовой пояс для сессии	UTC
user	Имя пользователя для подключения к серверу Trino	—

Kerberos Configuration

Параметр	Описание	Значение по умолчанию
krb5-remote-service-name	Сервисное имя для удаленного подключения	HTTP

SSL configuration

Параметр	Описание	Значение по умолчанию
keystore-password	Пароль к keystore-файлу	—
keystore-path	Расположение keystore-файла	—
truststore-password	Пароль к truststore-файлу	—
truststore-path	Расположение truststore-файла	—

YARN

mapred-site.xml

Параметр	Описание	Значение по умолчанию
mapreduce.application.classpath	classpath для приложений MapReduce. Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните + . Если установлено свойство `mapreduce.application.framework`, то в значении этого свойства должен содержаться classpath, также подходящий для этого архива, а имя архива должно содержаться в classpath. Если `mapreduce.app-submission.cross-platform=false`, для составления classpath-записей будет использован платформозависимый синтаксис расширения переменной среды. Если `mapreduce.app-submission.cross-platform=true`, будет использоваться classpath по умолчанию, не зависящий от платформы, для приложений MapReduce: {{HADOOP_MAPRED_HOME}}/share/hadoop/mapreduce/*, {{HADOOP_MAPRED_HOME}}/share/hadoop/mapreduce/lib/* NodeManager заменит маркер расширения параметра при запуске контейнера в зависимости от операционной системы	- /etc/hadoop/conf/* - /usr/lib/hadoop/* - /usr/lib/hadoop/lib/* - /usr/lib/hadoop-hdfs/* - /usr/lib/hadoop-hdfs/lib/* - /usr/lib/hadoop-yarn/* - /usr/lib/hadoop-yarn/lib/* - /usr/lib/hadoop-mapreduce/* - /usr/lib/hadoop-mapreduce/lib/*
mapreduce.cluster.local.dir	Локальный каталог, в котором MapReduce хранит промежуточные файлы данных. В значении параметра может быть указан разделенный запятыми список каталогов на разных устройствах для распределения нагрузки на диски. Несуществующие каталоги игнорируются	/srv/hadoop-yarn/mr-local
mapreduce.framework.name	Фреймворк для выполнения MapReduce-задач. Допустимые значения: `local`, `classic` или `yarn`	yarn
mapreduce.jobhistory.address	Адрес MapReduce JobHistory Server (в формате <host>:<port>)	—
mapreduce.jobhistory.bind-host	Установка значения `0.0.0.0` приводит к тому, что MapReduce-демоны слушают все адреса и интерфейсы хостов в кластере	0.0.0.0
mapreduce.jobhistory.webapp.address	Адрес для доступа к веб-интерфейсу MapReduce JobHistory Server (в формате <host>:<port>)	—
mapreduce.map.env	Переменные среды для процессов map-задач, добавленные пользователем в виде списка, разделенного запятыми. Пример: `VAR1=value1,VAR2=value2`	HADOOP_MAPRED_HOME=/usr/lib/hadoop-mapreduce
mapreduce.reduce.env	Переменные среды для процессов reduce-задач, добавленные пользователем в виде списка, разделенного запятыми. Пример: `VAR1=value1,VAR2=value2`	HADOOP_MAPRED_HOME=/usr/lib/hadoop-mapreduce
yarn.app.mapreduce.am.env	Переменные среды для процессов MapReduce App Master, добавленные пользователем. Примеры: - `A=foo`. Устанавливает переменную `A` со значением `foo`. - `B=$B:c`. Наследует значение переменной `B`.	HADOOP_MAPRED_HOME=/usr/lib/hadoop-mapreduce
yarn.app.mapreduce.am.staging-dir	Промежуточный каталог, используемый при отправке задач	/user
mapreduce.jobhistory.keytab	Kerberos keytab-файл для MapReduce JobHistory Server	/etc/security/keytabs/mapreduce-historyserver.service.keytab
mapreduce.jobhistory.principal	Принципал Kerberos для MapReduce JobHistory Server	mapreduce-historyserver/_HOST@REALM
mapreduce.jobhistory.http.policy	Настраивает HTTP-endpoint для веб-интерфейса JobHistoryServer. Доступны следующие значения: - `HTTP_ONLY` — обеспечивает работу через HTTP; - `HTTPS_ONLY` — обеспечивает работу через HTTPS.	HTTP_ONLY

mapreduce.jobhistory.webapp.https.address	HTTPS-адрес веб-приложения MapReduce JobHistory Server	0.0.0.0:19890
mapreduce.shuffle.ssl.enabled	Определяет, использовать ли SSL для HTTP-endpoint shuffle	false

ranger-yarn-audit.xml

Параметр	Описание	Значение по умолчанию
xasecure.audit.destination.solr.batch.filespool.dir	Путь к Spool-каталогу	/srv/ranger/hdfs_plugin/audit_solr_spool
xasecure.audit.destination.solr.urls	URL сервера Solr для сохранения аудит-событий. Оставьте значение пустым или установите значение `NONE` при использовании ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.zookeepers	Указывает строку подключения ZooKeeper для подключения к Solr	—
xasecure.audit.destination.solr.force.use.inmemory.jaas.config	Использует in-memory JAAS-файл для соединения с Solr	—
xasecure.audit.is.enabled	Активирует аудит Ranger	true
xasecure.audit.jaas.Client.loginModuleControlFlag	Определяет требование к успешному завершению работы модуля. Значения могут быть `required`, `requisite`, `sufficient` или `optional`	—
xasecure.audit.jaas.Client.loginModuleName	Имя класса-аутентификатора	—
xasecure.audit.jaas.Client.option.keyTab	Имя keytab-файла, из которого необходимо получить секретный ключ принципала	—
xasecure.audit.jaas.Client.option.principal	Имя используемого принципала	—
xasecure.audit.jaas.Client.option.serviceName	Имя пользователя или сервиса, для которого выполнятся логин	—
xasecure.audit.jaas.Client.option.storeKey	Установите значение `true`, чтобы keytab-файл или ключ принципала сохранялись в учетных данных субъекта	false
xasecure.audit.jaas.Client.option.useKeyTab	Установите значение `true`, чтобы модуль получал ключ принципала из keytab-файла	false

ranger-yarn-security.xml

Параметр	Описание	Значение по умолчанию
ranger.plugin.yarn.policy.rest.url	URL для Ranger Admin	—
ranger.plugin.yarn.service.name	Имя Ranger-сервиса, который содержит политики для этого экземпляра	—
ranger.plugin.yarn.policy.cache.dir	Каталог, в котором кешируются политики Ranger после успешного извлечения из источника	/srv/ranger/yarn/policycache

ranger.plugin.yarn.policy.pollIntervalMs	Определяет, как часто проверять политики на предмет изменения	30000
ranger.plugin.yarn.policy.rest.client.connection.timeoutMs	Время ожидания подключения YARN-плагина RangerRestClient (в миллисекундах)	120000
ranger.plugin.yarn.policy.rest.client.read.timeoutMs	Тайм-аут на чтение для YARN-плагина RangerRestClient (в миллисекундах)	30000
ranger.add-yarn-authorization	Установите значение `true` для использования списков управления доступом Ranger	false
ranger.plugin.yarn.policy.rest.ssl.config.file	Путь к конфигурационному файлу RangerRestClient SSL плагина YARN	/etc/yarn/conf/ranger-yarn-policymgr-ssl.xml

yarn-site.xml

Параметр	**Описание**	**Значение по умолчанию**
yarn.application.classpath	classpath для приложений YARN. Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	■ /etc/hadoop/conf/* ■ /usr/lib/hadoop/* ■ /usr/lib/hadoop/lib/* ■ /usr/lib/hadoop-hdfs/* ■ /usr/lib/hadoop-hdfs/lib/* ■ /usr/lib/hadoop-yarn/* ■ /usr/lib/hadoop-yarn/lib/* ■ /usr/lib/hadoop-mapreduce/* ■ /usr/lib/hadoop-mapreduce/lib/* ■ /usr/lib/hive/lib/*.jar
yarn.cluster.max-application-priority	Определяет максимальный приоритет приложения в кластере. Приоритет на уровне Leaf Queue: каждая Leaf-очередь предоставляет приоритет по умолчанию администратором. Приоритет очереди по умолчанию будет использоваться для любого приложения, отправленного без явно указанного приоритета. *$HADOOP_HOME/etc/hadoop/capacity-scheduler.xml* является файлом конфигурации приоритетов на уровне очереди	0
yarn.log.server.url	URL сервера агрегации логов	—
yarn.log-aggregation-enable	Активирует агрегацию логов. Логи агрегируются из каждого контейнера и перемещаются в файловую систему, например HDFS, после завершения работы приложения. Пользователи могут настроить свойства `yarn.nodemanager.remote-app-log-dir` и `yarn.nodemanager.remote-app-log-dir-suffix`, чтобы определить, куда перемещаются логи. Доступ к логам можно получить через Application Timeline Server	true
yarn.log-aggregation.retain-seconds	Определяет, как долго хранить логи перед их удалением. Значение `-1` отключает сохранение логов. Будьте осторожны: установка слишком маленького значения приведет к спаму NameNode	172800
yarn.nodemanager.local-dirs	Список локальных каталогов для хранения. Каталог локальных файлов приложения находится в: *${yarn.nodemanager.local-dirs}/usercache/${user}/appcache/application_${appid}*. Отдельные рабочие каталоги контейнеров, называемые *container_${contid}*, будут подкаталогами	/srv/hadoop-yarn/nm-local

yarn.node-labels.enabled	Активирует функцию меток узлов	true
yarn.node-labels.fs-store.root-dir	URI для NodeLabelManager. Значение по умолчанию — `/tmp/hadoop-yarn-${user}/node-labels/` в локальной файловой системе	hdfs:///system/yarn/node-labels
yarn.timeline-service.bind-host	Фактический адрес, к которому будет привязан сервер. Если этот необязательный адрес установлен, серверы RPC и Webapp будут привязаны к этому адресу и порту, указанным в `yarn.timeline-service.address` и `yarn.timeline-service.webapp.address`, соответственно. Данный параметр может быть полезен, чтобы заставить сервис слушать все интерфейсы, установив значение `0.0.0.0`	0.0.0.0
yarn.timeline-service.leveldb-timeline-store.path	Сохраняет имя файла для leveldb Timeline-хранилища	/srv/hadoop-yarn/leveldb-timeline-store
yarn.nodemanager.address	Адрес менеджера контейнеров в NodeManager	0.0.0.0:8041
yarn.nodemanager.aux-services	Список сервисов, разделенных запятыми, где имя сервиса должно содержать только символы `a-zA-Z0-9_` и не может начинаться с цифр	mapreduce_shuffle,spark_shuffle
yarn.nodemanager.aux-services.mapreduce_shuffle.class	Вспомогательный класс сервиса	org.apache.hadoop.mapred.ShuffleHandler
yarn.nodemanager.aux-services.spark_shuffle.class	Имя класса *YarnShuffleService* — внешнего shuffle-сервиса для Spark3 в YARN	org.apache.spark.network.yarn.YarnShuffleService
yarn.nodemanager.aux-services.spark_shuffle.classpath	classpath для внешнего shuffle-сервиса Spark3 в YARN. Список файлов и директорий, которые должны быть добавлены в classpath. Чтобы добавить директорию или файл в classpath, кликните +	■ /usr/lib/spark3/yarn/lib/*
yarn.nodemanager.recovery.enabled	Позволяет NodeManager восстанавливаться после запуска	true
yarn.nodemanager.recovery.dir	Локальный каталог файловой системы, в котором NodeManager сохраняет состояние, если функция восстановления включена	/srv/hadoop-yarn/nm-recovery
yarn.nodemanager.remote-app-log-dir	Определяет каталог для агрегации логов	/logs
yarn.nodemanager.resource-plugins	Включает дополнительное обнаружение/изоляцию ресурсов в NodeManager. По умолчанию этот параметр пуст. Допустимые значения: `yarn.io/gpu`, `yarn.io/fpga`	—
yarn.nodemanager.resource-plugins.gpu.path-to-discovery-executables	Если `yarn.nodemanager.resource.gpu.allowed-gpu-devices=auto`, YARN NodeManager должен запустить бинарный файл обнаружения графического процессора (в данный момент поддерживается только nvidia-smi), чтобы получить информацию о GPU. если значение не установлено (по умолчанию), YARN NodeManager попытается самостоятельно найти файл обнаружения. Пример значения конфигурации: `/usr/local/bin/nvidia-smi`	/usr/bin/nvidia-smi
yarn.nodemanager.resource.detect-hardware-capabilities	Включает автоматическое определение параметров узла, таких как память и CPU	true
yarn.nodemanager.vmem-check-enabled	Определяет, будут ли применяться ограничения виртуальной памяти для контейнеров	false
yarn.resource-types	Типы ресурсов, которые будут использоваться для планирования. Используйте *resource-types.xml* для указания сведений об отдельных типах ресурсов	—

yarn.resourcemanager.bind-host	Фактический адрес, к которому будет привязан сервер. Если этот необязательный адрес установлен, серверы RPC и Webapp будут привязаны к этому адресу и порту, указанным в `yarn.resourcemanager.address` и `yarn.resourcemanager.webapp.address`, соответственно. При установке значения `0.0.0.0` Resource Manager слушает все интерфейсы	0.0.0.0
yarn.resourcemanager.cluster-id	Имя кластера. В режиме высокой доступности (High Availability, HA) этот параметр используется для того, чтобы Resource Manager участвовал в выборе лидера для этого кластера и чтобы он не влиял на другие кластеры	—
yarn.resource-types.memory-mb.increment-allocation	FairScheduler предоставляет память, равную приращениям значения параметра. Если вы отправляете задачу с запросом ресурсов, который не кратен `memory-mb.increment-allocation`, запрос будет округлен до ближайшего увеличения	1024
yarn.resource-types.vcores.increment-allocation	FairScheduler предоставляет vcore с приращением значения, указанного в параметре. Если вы отправляете задачу с запросом ресурсов, который не кратен `vcores.increment-allocation`, запрос будет округлен до ближайшего увеличения	1
yarn.resourcemanager.ha.enabled	Включает режим высокой доступности для Resource Manager. При активации режима: ■ Resource Manager по умолчанию запускается в режиме Standby и переходит в активный режим по запросу. ■ Узлы в ансамбле Resource Manager перечислены в `yarn.resourcemanager.ha.rm-ids`. ■ Идентификатор каждого Resource Manager либо берется из `yarn.resourcemanager.ha.id` (если `yarn.resourcemanager.ha.id` указан явно), либо может быть получен путем сопоставления `yarn.resourcemanager.address.{id}` с локальным адресом. ■ Фактические физические адреса берутся из конфигураций шаблона `{rpc-config}.{id}`.	false
yarn.resourcemanager.ha.rm-ids	Список узлов Resource Manager в кластере, в режиме высокой доступности. Для дополнительной информации смотрите описание параметра `yarn.resourcemanager.ha.enabled`	—
yarn.resourcemanager.hostname	Имя хоста Resource Manager	—
yarn.resourcemanager.leveldb-state-store.path	Локальный путь, по которому будет храниться состояние Resource Manager при использовании `org.apache.hadoop.yarn.server.resourcemanager.recovery.LeveldbRMStateStore` в качестве значения для `yarn.resourcemanager.store.class`	/srv/hadoop-yarn/leveldb-state-store
yarn.resourcemanager.monitor.capacity.queue-management.monitoring-interval	Время между вызовами политики QueueManagementDynamicEditPolicy (в миллисекундах)	1500
yarn.resourcemanager.reservation-system.enable	Активирует ReservationSystem в ResourceManager	false
yarn.resourcemanager.reservation-system.planfollower.time-step	Частота таймера PlanFollower (в миллисекундах)	1000
Resource scheduler	Тип подключаемого планировщика для Hadoop. Доступные значения: `CapacityScheduler` и `FairScheduler`. `CapacityScheduler` позволяет нескольким участникам безопасно использовать общий кластер, чтобы их приложениям своевременно выделялись ресурсы с учетом ограничений выделенных мощностей. `FairScheduler` позволяет поровну распределять ресурсы между YARN-приложениями в больших кластерах	CapacityScheduler
yarn.resourcemanager.scheduler.monitor.enable	Включает набор периодических мониторов (monitors, указанных в `yarn.resourcemanager.scheduler.monitor.policies`), которые влияют на планировщик	false

yarn.resourcemanager.scheduler.monitor.policies	Список классов SchedulingEditPolicy, взаимодействующих с планировщиком. Конкретный модуль может быть несовместим с планировщиком, другими политиками или их конфигурацией	org.apache.hadoop.yarn.server.resourcemanager.monitor.capacity.ProportionalCapacityPreemptionPolicy
yarn.resourcemanager.monitor.capacity.preemption.observe_only	Если установлено значение `true`, запускает политику, но в кластере не будут выполняться события вытеснения (preemption) и уничтожения	false
yarn.resourcemanager.monitor.capacity.preemption.monitoring_interval	Время между вызовами политики ProportionalCapacityPreemptionPolicy (в миллисекундах)	3000
yarn.resourcemanager.monitor.capacity.preemption.max_wait_before_kill	Время между запросом на вытеснение из приложения и уничтожением контейнера (в миллисекундах)	15000
yarn.resourcemanager.monitor.capacity.preemption.total_preemption_per_round	Максимальный процент ресурсов, вытесняемых за один раунд. Управляя этим значением, можно регулировать скорость, с которой контейнеры удаляются из кластера. После вычисления общего объема данных на вытеснение политика уменьшает это значение в пределах установленного лимита	0.1
yarn.resourcemanager.monitor.capacity.preemption.max_ignored_over_capacity	Максимальное количество ресурсов, которое может превышать целевой объем и игнорируется при вытеснении. Этот параметр определяет мертвую зону вокруг целевого объема, что помогает предотвратить погрешности вычисляемого целевого баланса. Высокие значения замедлят время достижения заданного объема и (при отсутствии `natural.completions`) могут помешать достижению гарантированного объема	0.1
yarn.resourcemanager.monitor.capacity.preemption.natural_termination_factor	Для определенного объекта вытеснения учитывает контейнеры с естественным сроком действия и вытесняет только процент от дельты. Это определяет скорость геометрического вхождения в мертвую зону (`MAX_IGNORED_OVER_CAPACITY`). Например, коэффициент завершения 0,5 вернет почти 95% ресурсов в течение 5 * `#WAIT_TIME_BEFORE_KILL`, даже при отсутствии естественного завершения	0.2
yarn.resourcemanager.nodes.exclude-path	Путь к файлу с узлами для исключения	/etc/hadoop/conf/exclude-path.xml
yarn.resourcemanager.nodes.include-path	Путь к файлу с узлами для включения	/etc/hadoop/conf/include-path
yarn.resourcemanager.recovery.enabled	Позволяет Resource Manager восстанавливать состояние после запуска. Если установлено значение `true`, необходимо указать `yarn.resourcemanager.store.class`	true
yarn.resourcemanager.store.class	Класс для использования в качестве постоянного хранилища. Если используется `org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore`, хранилище неявно ограждается (fenced); это означает, что один Resource Manager может использовать хранилище в любой момент времени	—
yarn.resourcemanager.system-metrics-publisher.enabled	Параметр определяет, должен ли Resource Manager публиковать системные метрики YARN в Timeline Server	true
yarn.scheduler.fair.user-as-default-queue	Определяет, использовать ли имя пользователя, связанное с выделением ресурсов, в качестве имени очереди по умолчанию в случае, если имя очереди не указано. Если установлено значение `false` или параметр не задан, все задачи используют общую очередь по умолчанию с именем `default`. Значение по умолчанию — `true`. Если в файле распределения задана политика размещения очереди, это свойство игнорируется	true
yarn.scheduler.fair.preemption	Определяет, использовать ли вытеснение (preemption)	false
yarn.scheduler.fair.preemption.cluster-utilization-threshold	Порог использования, после которого срабатывает вытеснение (preemption). Использование определяется как максимальное отношение использования к объему среди всех ресурсов	0.8f

yarn.scheduler.fair.sizebasedweight	Определяет, следует ли предоставлять общие ресурсы отдельным приложениям в зависимости от их размера, вместо того, чтобы предоставлять равное количество ресурсов всем приложениям независимо от размера. Если установлено значение `true`, приложения оцениваются по формуле: натуральный логарифм единицы + (память приложения, деленная на натуральный логарифм двойки)	false
yarn.scheduler.fair.assignmultiple	Определяет, разрешать ли несколько назначений контейнеров за один heartbeat	false
yarn.scheduler.fair.dynamic.max.assign	Если `assignmultiple=true`, параметр указывает, следует ли динамически определять количество ресурсов, выделяемых за один hearbeat. При значении `true` примерно половина нераспределенных ресурсов на узле выделяется контейнерам за один hearbeat	true
yarn.scheduler.fair.max.assign	Если `assignmultiple=true`, параметр указывает максимальное количество контейнеров, которое может быть назначено за один такт. Значение по умолчанию: `-1`, что не устанавливает ограничений	-1
yarn.scheduler.fair.locality.threshold.node	Для приложений, которые запрашивают контейнеры на определенных узлах, параметр определяет количество возможностей планирования с момента последнего назначения контейнера до размещения на другом узле. Валидное значение — число с плавающей запятой от `0` до `1`, пропорциональное размеру кластера и выражающее количество возможностей планирования, которые нужно пропустить. Значение по умолчанию `-1.0` означает не упускать возможности планирования	-1.0
yarn.scheduler.fair.locality.threshold.rack	Для приложений, которые запрашивают контейнеры на RAID-массивах, количество возможностей планирования с момента последнего назначения контейнера для ожидания перед принятием размещения на другой стойке. Выражается в виде числа с плавающей запятой от `0` до `1` пропорционального размеру кластера и представляющее количество возможностей планирования, которые необходимо упустить. Значение по умолчанию `-1.0` означает не упускать возможности планирования	-1.0
yarn.scheduler.fair.allow-undeclared-pools	При значении `true` могут создаваться новые очереди во время отправки приложения. Очередь приложения может быть указана как отправителем, так и параметром `user-as-default-queue`. При значении `false` каждый раз, когда приложение помещается в очередь, не указанную в файле распределения, вместо этого оно помещается в очередь `default`. По умолчанию `true`. Если в файле распределения задана политика размещения очереди, это свойство игнорируется	true
yarn.scheduler.fair.update-interval-ms	Интервал времени, в течение которого необходимо заблокировать планировщик, пересчитать fair-доли, спрос и проверить, нет ли кандидатов на вытеснение	500
yarn.scheduler.minimum-allocation-mb	Устанавливает для Resource Manager минимальное выделение ресурсов для каждого запроса контейнера (в МБ). Запросы памяти ниже этого значения вызовут исключение `InvalidResourceRequestException`	1024
yarn.scheduler.maximum-allocation-mb	Устанавливает для Resource Manager максимальное выделение ресурсов для каждого запроса контейнера (в МБ). Запросы памяти выше этого значения вызовут исключение `InvalidResourceRequestException`	4096
yarn.scheduler.minimum-allocation-vcores	Минимальное количество ресурсов (с точки зрения виртуальных ядер CPU), выделяемых Resource Manager для каждого запроса контейнера. Запросы ниже этого значения будут вызывать исключение `InvalidResourceRequestException`	1
yarn.scheduler.maximum-allocation-vcores	Максимальное количество ресурсов (с точки зрения виртуальных ядер CPU), выделяемых Resource Manager для каждого запроса контейнера. Запросы выше этого значения будут вызывать исключение `InvalidResourceRequestException`	2
yarn.timeline-service.enabled	На стороне сервера этот параметр указывает, включен ли сервис Timeline. На стороне клиента этот параметр может использоваться, чтобы указать, хочет ли клиент использовать сервис Timeline. Если этот параметр установлен на стороне клиента и безопасность включена, YARN-клиент пытается получить токены делегирования для Timeline-сервера	true
yarn.timeline-service.hostname	Имя хоста веб-приложения службы временной шкалы	—

yarn.timeline-service.http-cross-origin.enabled	Активирует совместное использование ресурсов между разными источниками (Cross Origin Support, CORS) для Timeline-сервера	true
yarn.webapp.ui2.enable	На стороне сервера этот параметр указывает, активен ли новый YARN UI v2	true
yarn.resourcemanager.proxy-user-privileges.enabled	Если установлено значение `true`, Resource Manager будет иметь привилегии прокси-пользователя. Если установлено значение `true`, Resource Manager может запрашивать новые токены делегирования hdfs от имени пользователя. Это необходимо для долговременных сервисов, поскольку срок действия токенов hdfs в конечном итоге истечет, а YARN требует новых действительных токенов для локализации и агрегации логов. Обратите внимание, что для этой опции соответствующий HDFS NameNode должен иметь Resource Manager, настроенный как прокси-пользователь, чтобы Resource Manager мог сам запрашивать новые токены от имени пользователя, когда токены истекли	false
yarn.resourcemanager.webapp.spnego-principal	Принципал Kerberos, который будет использоваться для фильтра SPNEGO для веб-интерфейса Resource Manager	HTTP/_HOST@REALM
yarn.resourcemanager.webapp.spnego-keytab-file	Kerberos keytab-файл, который будет использоваться для фильтра SPNEGO для веб-интерфейса Resource Manager	/etc/security/keytabs/HTTP.service.keytab
yarn.nodemanager.linux-container-executor.group	Группа UNIX, от имени которой должен работать linux-container-executor	yarn
yarn.resourcemanager.webapp.delegation-token-auth-filter.enabled	Флаг, позволяющий переопределить фильтр проверки подлинности Kerberos по умолчанию фильтром проверки подлинности Resource Manager, чтобы разрешить проверку подлинности с использованием токенов делегирования (возврат к Kerberos в случае отсутствия токенов). Применимо только в том случае, если тип HTTP-аутентификации задан `kerberos`	false
yarn.resourcemanager.principal	Принципал Kerberos для Resource Manager	yarn-resourcemanager/_HOST@REALM
yarn.resourcemanager.keytab	Kerberos keytab-файл для Resource Manager	/etc/security/keytabs/yarn-resourcemanager.service.keytab
yarn.resourcemanager.webapp.https.address	HTTPS-адрес веб-приложения Resource Manager. Если в качестве значения указан только хост, веб-приложение будет использовать случайный порт	${yarn.resourcemanager.hostname}:8090
yarn.nodemanager.principal	Принципал Kerberos для NodeManager	yarn-nodemanager/_HOST@REALM
yarn.nodemanager.keytab	Kerberos keytab-файл для NodeManager	/etc/security/keytabs/yarn-nodemanager.service.keytab
yarn.nodemanager.webapp.spnego-principal	Принципал Kerberos для использования SPNEGO-фильтра для веб-интерфейса NodeManager	HTTP/_HOST@REALM
yarn.nodemanager.webapp.spnego-keytab-file	Kerberos keytab-файл для использования SPNEGO-фильтра для веб-интерфейса NodeManager	/etc/security/keytabs/HTTP.service.keytab
yarn.nodemanager.webapp.https.address	HTTPS-адрес веб-приложения NodeManager	0.0.0.0:8044
yarn.timeline-service.http-authentication.type	Определяет тип аутентификации, которая используется для HTTP-endpoint Timeline Server. Допустимые значения: `simple`, `kerberos`, `#AUTHENTICATION_HANDLER_CLASSNAME#`	simple
yarn.timeline-service.http-authentication.simple.anonymous.allowed	Указывает, разрешены ли анонимные запросы от Timeline Server с использованием `simple` аутентификации	true

yarn.timeline-service.http-authentication.kerberos.keytab	Kerberos keytab-файл, который используется для Timeline Server (Collector/Reader) HTTP endpoint	/etc/security/keytabs/HTTP.service.keytab
yarn.timeline-service.http-authentication.kerberos.principal	Принципал Kerberos, который используется для Timeline Server (Collector/Reader) HTTP endpoint	HTTP/_HOST@REALM
yarn.timeline-service.principal	Принципал Kerberos для сервиса timeline reader. Принципал NodeManager будет использован для timeline collector, так как является вспомогательным сервисом внутри NodeManager	yarn/_HOST@REALM
yarn.timeline-service.keytab	Kerberos keytab-файл для timeline reader. Keytab-файл NodeManager будет использован для timeline collector, так как является вспомогательным сервисом внутри NodeManager	/etc/security/keytabs/yarn.service.keytab
yarn.timeline-service.delegation.key.update-interval	Интервал обновления ключей делегирования	86400000
yarn.timeline-service.delegation.token.renew-interval	Интервал обновления токенов делегирования	86400000
yarn.timeline-service.delegation.token.max-lifetime	Максимальное время жизни токена	86400000
yarn.timeline-service.client.best-effort	Определяет, считать ли ошибку при получении токена делегирования ошибкой приложения (`false`), или клиент должен попытаться опубликовать информацию без токена (`true`)	false
yarn.timeline-service.webapp.https.address	HTTPS-адрес веб-приложения сервиса Timeline	${yarn.timeline-service.hostname}:8190
yarn.http.policy	Параметр настраивает HTTP-endpoint для демонов Yarn. Доступны следующие значения: ▪ `HTTP_ONLY` — обеспечивает работу через HTTP; ▪ `HTTPS_ONLY` — обеспечивает работу через HTTPS.	HTTP_ONLY
yarn.nodemanager.container-executor.class	Имя container-executor Java-класса	org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor

container-executor.cfg

ВНИМАНИЕ

В AstraLinux UID пользователей могут начинаться со 100. Для корректной работы YARN на AstraLinux установите значение параметра `min.user.id` равным 100.

Параметр	Описание	Значение по умолчанию
banned.users	CC пользователей, разделенных запятой, которые не могут запускать приложения	bin
min.user.id	Исключает других super-пользователей	500

Enable CORS

Параметр	Описание	Значение по умолчанию
yarn.nodemanager.webapp.cross-origin.enabled	Обеспечивает поддержку кросс-источников для веб-сервисов NodeManager	true
yarn.resourcemanager.webapp.cross-origin.enabled	Обеспечивает поддержку кросс-источников для веб-сервисов ResourceManager	true
yarn_site.enable_cors.active	Включает CORS (Cross-Origin Resource Sharing)	true

yarn-env.sh

Параметр	Описание	Значение по умолчанию
YARN_RESOURCEMANAGER_OPTS	YARN ResourceManager heap memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap для ResourceManager	-Xms1G -Xmx8G
YARN_NODEMANAGER_OPTS	YARN NodeManager heap memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap для NodeManager	—
YARN_TIMELINESERVER_OPTS	YARN Timeline Server heap memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap для Timeline Server	-Xms700m -Xmx8G

Lists of decommissioned hosts

Параметр	Описание	Значение по умолчанию
DECOMMISSIONED	Список хостов в состоянии `DECOMMISSIONED`	—

ranger-yarn-policymgr-ssl.xml

Параметр	Описание	Значение по умолчанию
xasecure.policymgr.clientssl.keystore	Путь к keystore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.credential.file	Путь к файлу с паролями для keystore-файла	/etc/yarn/conf/ranger-yarn.jceks
xasecure.policymgr.clientssl.truststore.credential.file	Путь к truststore-файлу с учетными данными	/etc/yarn/conf/ranger-yarn.jceks
xasecure.policymgr.clientssl.truststore	Путь к truststore-файлу, который использует Ranger	—
xasecure.policymgr.clientssl.keystore.password	Пароль для доступа к keystore-файлу	—
xasecure.policymgr.clientssl.truststore.password	Пароль для доступа к truststore-файлу	—

mapred-env.sh

Параметр	Описание	Значение по умолчанию
HADOOP_JOB_HISTORYSERVER_OPTS	MapReduce History Server heap memory. Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap для History Server	-Xms700m -Xmx8G

Другие параметры

Параметр	Описание	Значение по умолчанию
GPU on YARN	Определяет, использовать ли GPU на YARN	false
capacity-scheduler.xml	Содержимое файла *capacity-scheduler.xml*, который используется CapacityScheduler	Default capacity-scheduler.xml
fair-scheduler.xml	Содержимое файла *fair-scheduler.xml*, который используется FairScheduler	Default fair-scheduler.xml
Custom mapred-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *mapred-site.xml*	—
Ranger plugin enabled	Определяет, активен ли Ranger-плагин	false
Custom yarn-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *yarn-site.xml*	—
Custom ranger-yarn-audit.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-yarn-audit.xml*	—
Custom ranger-yarn-security.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-yarn-security.xml*	—
Custom ranger-yarn-policymgr-ssl.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *ranger-yarn-policymgr-ssl.xml*	—
Custom mapred-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *mapred-env.sh*	—
Custom yarn-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *yarn-env.sh*	—
container-executor.cfg template	Шаблон для файла конфигурации *container-executor.cfg*	—

Компонент YARN NodeManager

Monitoring

Параметр	Описание	Значение по умолчанию
Java agent path	Путь к Java-агенту JMX Prometheus	/usr/lib/adh-utils/jmx/jmx_prometheus_javaagent.jar

Prometheus metrics port	Порт, на котором будут доступны метрики YARN NodeManager в формате Prometheus	9205
Mapping config path	Путь к файлу конфигурации мэппинга метрик	/etc/hadoop/conf/jmx_yarn_nodemanager_metric_config.yml
Mapping config	Файл конфигурации мэппинга метрик	yarn-mapping-config.yml

Компонент YARN ResourceManager

Monitoring

Параметр	Описание	Значение по умолчанию
Java agent path	Путь к Java-агенту JMX Prometheus	/usr/lib/adh-utils/jmx/jmx_prometheus_javaagent.jar
Prometheus metrics port	Порт, на котором будут доступны метрики YARN ResourceManager в формате Prometheus	9204
Mapping config path	Путь к файлу конфигурации мэппинга метрик	/etc/hadoop/conf/jmx_yarn_resourcemanager_metric_config.yml
Mapping config	Файл конфигурации мэппинга метрик	yarn-mapping-config.yml

Компонент YARN Timeline Server

Monitoring

Параметр	Описание	Значение по умолчанию
Java agent path	Путь к Java-агенту JMX Prometheus	/usr/lib/adh-utils/jmx/jmx_prometheus_javaagent.jar
Prometheus metrics port	Порт, на котором будут доступны метрики YARN Timeline Server в формате Prometheus	9206
Mapping config path	Путь к файлу конфигурации мэппинга метрик	/etc/hadoop/conf/jmx_yarn_timelineserver_metric_config.yml
Mapping config	Файл конфигурации мэппинга метрик	yarn-mapping-config.yml

Zeppelin

User-managed interpreters

Параметр	Описание	Значение по умолчанию
Allow user-managed interpreters	Позволяет использовать интерпретаторы Zeppelin, у которых объявлено свойство `user-managed=true`. Если опция включена, при рестарте Zeppelin с помощью ADCM кастомные свойства интерпретатора будут сохранены	true
Custom interpreter.json	Активирует конфигурацию для интерпретаторов, описанную в поле ADCM ***interpreter.json***	false

interpreter.json	Кастомная JSON-конфигурация интерпретаторов, которые должны быть доступны в веб-интерфейсе Zeppelin. Использование этой конфигурации перезаписывает настройки всех интерпретаторов (пользовательских и системных)	interpreters.json
interpreter.sh	Кастомное содержимое скрипта *interpreter.sh*. Данный скрипт вызывается при запуске Zeppelin и выполняет инициализацию окружения, необходимую для работы Zeppelin	interpreters.sh

zeppelin-site.xml

Параметр	Описание	Значение по умолчанию
zeppelin.dep.localrepo	Локальный репозиторий для загрузчика зависимостей	/srv/zeppelin/local-repo
zeppelin.server.addr	Адрес биндинга Zeppelin-сервера	0.0.0.0
zeppelin.server.port	Порт сервера	8180
zeppelin.server.kerberos.principal	Имя принципала, загружаемого из keytab-файла	—
zeppelin.server.kerberos.keytab	Путь к keytab-файлу	—
zeppelin.shell.auth.type	Устанавливает тип аутентификации. Допустимые значения: `SIMPLE` и `KERBEROS`	—
zeppelin.shell.principal	Имя принципала, загружаемого из keytab-файла	—
zeppelin.shell.keytab.location	Путь к keytab-файлу	—
zeppelin.jdbc.auth.type	Устанавливает тип аутентификации. Допустимые значения: `SIMPLE` и `KERBEROS`	—
zeppelin.jdbc.keytab.location	Путь к keytab-файлу	—
zeppelin.jdbc.principal	Имя принципала, загружаемого из keytab-файла	—
zeppelin.jdbc.auth.kerberos.proxy.enable	Если используется тип аутентификации `KERBEROS`, этот параметр включает/отключает проксирование с использованием логина пользователя для установки соединения	true
spark.yarn.keytab	Абсолютный путь к keytab-файлу для принципала. Этот keytab-файл будет скопирован на узел, на котором запущен YARN Application Master через безопасный распределенный кеш (Secure Distributed Cache), для периодического обновления логин-тикетов и токенов делегирования	—
spark.yarn.principal	Имя принципала, используемое для логина в KDC, запущенном на защищенном HDFS-кластере	—
zeppelin.livy.keytab	Путь к keytab-файлу	—
zeppelin.livy.principal	Имя принципала, загружаемого из keytab-файла	—
zeppelin.server.ssl.port	Номер порта, используемого для SSL-соединения	8180
zeppelin.ssl	Определяет, использовать ли SSL-соединение	false
zeppelin.ssl.keystore.path	Путь к keystore-файлу, который использует Zeppelin	—

zeppelin.ssl.keystore.password	Пароль для доступа к keystore-файлу	—
zeppelin.ssl.truststore.path	Путь к truststore-файлу, который использует Zeppelin	—
zeppelin.ssl.truststore.password	Пароль для доступа к truststore-файлу	—

Zeppelin server heap memory settings

Параметр	Описание	Значение по умолчанию
Zeppelin Server Heap Memory	Задает начальный (-Xms) и максимальный (-Xmx) размер Java heap для Zeppelin Server	-Xms700m -Xmx1024m

Shiro Simple username/password auth

Параметр	Описание	Значение по умолчанию
Users/password map	Сопоставление типа <username: password>. Например, `<myUser1: password1>`	—
sessionManager	Класс, отвечающий за создание, поддержку и очистку всех сессий приложения	org.apache.shiro.web.session.mgt.DefaultWebSessionManager
securityManager.sessionManager	Реализация компонента SessionManager для SecurityManager	$sessionManager
securityManager.sessionManager.globalSessionTimeout	Тайм-аут в миллисекундах, по истечении которого сессия считается истекшей и пользователю необходимо выполнить вход в систему	86400000
shiro.loginUrl	Устанавливает свойство `loginUrl` для стандартных фильтров Shiro	/api/login
cookie	Класс, отвечающий за обработку файлов cookie	org.apache.shiro.web.servlet.SimpleCookie
cookie.name	Имя сессионного cookie-файла, используемого Shiro. Этот параметр должен быть равен параметру `responseExcludeHeaders` в */usr/lib/knox/data/services/zeppelinui/{version}/service.xml*	JSESSIONID
cookie.httpOnly	Определяет, должны ли cookie Shiro иметь атрибут `HttpOnly`	true
cookie.secure	Определяет, должны ли cookie Shiro иметь атрибут `Secure`	false
sessionManager.sessionIdCookie	Реализация cookie для sessionManager	$cookie

Shiro LDAP auth

Параметр	Описание	Значение по умолчанию
ldapRealm	Расширяет провайдер Apache Shiro, чтобы разрешить поиск LDAP и предоставить членство в группе провайдеру авторизации	org.apache.zeppelin.realm.LdapRealm
ldapRealm.contextFactory.authenticationMechanism	Указывает тип аутентификации, используемый LDAP-сервисом	simple

ldapRealm.contextFactory.url	URL LDAP-сервера. Например, *ldap://ldap.example.com:389*	—
ldapRealm.userDnTemplate	Опциональный параметр. Knox использует это значение для составления UserDN для привязки аутентификации. UserDN, где токен `{0}` указывает на атрибут, который соответствует логину пользователя. Например, UserDnTemplate для Apache DS с Knox: `uid={0},ou=people,dc=hadoop,dc=apache,dc=org`	—
ldapRealm.pagingSize	Позволяет установить размер страницы LDAP	100
ldapRealm.authorizationEnabled	Включает авторизацию для Shiro ldapRealm	true
ldapRealm.contextFactory.systemAuthenticationMechanism	Определяет механизм аутентификации, используемый Shiro ldapRealm context factory. Допустимые значения: `simple` и `digest-md5`	—
ldapRealm.userLowerCase	Возвращает имена пользователей из LDAP в нижнем регистре	true
ldapRealm.memberAttributeValueTemplate	Атрибут для идентификации пользователя в группе. Например: `cn={0},ou=people,dc=hadoop,dc=apache,dc=org`	—
ldapRealm.searchBase	Начальный DN в LDAP DIT, с которого следует начать поиск. Поиск осуществляется только в дочерних деревьях указанного поддерева. Например: `dc=hadoop,dc=apache,dc=org`	—
ldapRealm.userSearchBase	База поиска для DN привязки пользователя. По умолчанию используется значение `ldapRealm.searchBase`, если значение не задано. Если задан параметр `ldapRealm.userSearchAttributeName`, также необходимо указать значение для `ldapRealm.searchBase` или `ldapRealm.userSearchBase`	—
ldapRealm.groupSearchBase	База поиска, используемая для поиска групп. По умолчанию используется значение `ldapRealm.searchBase`. Устанавливается только в том случае, если `ldapRealm.authorizationEnabled=true`	—
ldapRealm.groupObjectClass	Значение класса объекта, который идентифицирует записи группы в LDAP	groupofnames
ldapRealm.userSearchAttributeName	Атрибут, соответствующий логин-токену пользователя. Этот атрибут используется с результатами поиска для вычисления UserDN для привязки аутентификации	sAMAccountName
ldapRealm.memberAttribute	Значение атрибута, определяющего членство в группе. При значении `rememberer` найденные группы рассматриваются как динамические группы	member
ldapRealm.userSearchScope	Позволяет определить searchScopes. Возможные значения: `subtree`, `one`, `base`	subtree
ldapRealm.groupSearchScope	Позволяет определить groupSearchScope. Возможные значения: `subtree`, `one`, `base`	subtree
ldapRealm.contextFactory.systemUsername	Учетная запись службы LDAP, которую Zeppelin использует для поиска LDAP. При необходимости укажите полное имя пользователя учетной записи. Например: `uid=guest,ou=people,dc=hadoop,dc=apache,dc=org`. Для этой учетной записи требуется разрешение на чтение базового DN-поиска	—
ldapRealm.contextFactory.systemPassword	Устанавливает пароль для systemUsername. Этот пароль будет добавлен в хранилище ключей с учетными данными `hadoop`	—
ldapRealm.groupSearchEnableMatchingRuleInChain	Включает поддержку вложенных групп с помощью оператора LDAP_MATCHING_RULE_IN_CHAIN	true

ldapRealm.rolesByGroup	Опциональный параметр. Сопоставление физических групп с логическими ролями приложений. Например: `"LDN_USERS":"user_role", "NYK_USERS":"user_role", "HKG_USERS":"user_role", "GLOBAL_ADMIN":"admin_role"`	—
ldapRealm.allowedRolesForAuthentication	Необязательный список ролей, которым разрешено проходить аутентификацию. Если параметр не задан, всем группам разрешено проходить аутентификацию. Не влияет на разрешения для конкретных URL-адресов, которые будут продолжать работать, как указано в [urls]. Например: `admin_role,user_role`	—
ldapRealm.permissionsByRole	Опциональный параметр. Устанавливает права по ролям. Например: `'user_role = :ToDoItemsJdo::*, :ToDoItem::*; admin_role = *'`	—
securityManager.realms	Указывает список Apache Shiro Realms	$ldapRealm
sessionManager	Класс, отвечающий за создание, поддержку и очистку всех сессий приложения	org.apache.shiro.web.session.mgt.DefaultWebSessionManager
securityManager.sessionManager	Реализация компонента SessionManager для SecurityManager	$sessionManager
securityManager.sessionManager.globalSessionTimeout	Тайм-аут в миллисекундах, по истечении которого сессия считается истекшей и пользователю необходимо выполнить вход в систему	86400000
shiro.loginUrl	Устанавливает свойство `loginUrl` для стандартных фильтров Shiro	/api/login
cookie	Класс, отвечающий за обработку файлов cookie	org.apache.shiro.web.servlet.SimpleCookie
cookie.name	Имя сессионного cookie-файла, используемого Shiro. Этот параметр должен быть равен параметру `responseExcludeHeaders` в */usr/lib/knox/data/services/zeppelinui/{version}/service.xml*	JSESSIONID
cookie.httpOnly	Определяет, должны ли cookie Shiro иметь атрибут `HttpOnly`	true
cookie.secure	Определяет, должны ли cookie Shiro иметь атрибут `Secure`	false
sessionManager.sessionIdCookie	Реализация cookie для sessionManager	$cookie

Shiro Active Directory auth

Параметр	**Описание**	**Значение по умолчанию**
activeDirectoryRealm	Shiro realm для работы с Active Directory	org.apache.zeppelin.realm.ActiveDirectoryGroupRealm
activeDirectoryRealm.systemUsername	Имя пользователя для подключения к Active Directory	—
activeDirectoryRealm.systemPassword	Пароль для подключения к Active Directory	—
activeDirectoryRealm.searchBase	Базовый DN сервера Active Directory. Например: `CN=Users,DC=SOME_GROUP,DC=COMPANY,DC=COM`	—
activeDirectoryRealm.url	URL-адрес сервера Active Directory. Например: *ldap://ldap.example.com:389*	—

activeDirectoryRealm.groupRolesMap	Сопоставление групп Active Directory с ролями Apache Shiro. Например: `'CN=aGroupName,OU=groups,DC=SOME_GROUP,DC=COMPANY,DC=COM':'group1', 'CN=bGroupName,OU=groups,DC=SOME_GROUP,DC=COMPANY,DC=COM':'group2'`	—
activeDirectoryRealm.authorizationCachingEnabled	Включает/отключает кеширование решений об авторизации	false
securityManager.realms	Shiro Realm, используемый компонентом securityManager	$activeDirectoryRealm
sessionManager	Класс, отвечающий за создание, поддержку и очистку всех сессий приложения	org.apache.shiro.web.session.mgt.DefaultWebSessionManager
securityManager.sessionManager	Реализация компонента SessionManager для SecurityManager	$sessionManager
securityManager.sessionManager.globalSessionTimeout	Тайм-аут в миллисекундах, по истечении которого сессия считается истекшей и пользователю необходимо выполнить вход в систему	86400000
shiro.loginUrl	Устанавливает свойство `loginUrl` для стандартных фильтров Shiro	/api/login
cookie	Класс, отвечающий за обработку файлов cookie	org.apache.shiro.web.servlet.SimpleCookie
cookie.name	Имя сессионного cookie-файла, используемого Shiro. Этот параметр должен быть равен параметру `responseExcludeHeaders` в */usr/lib/knox/data/services/zeppelinui/{version}/service.xml*	JSESSIONID
cookie.httpOnly	Определяет, должны ли cookie Shiro иметь атрибут `HttpOnly`	true
cookie.secure	Определяет, должны ли cookie Shiro иметь атрибут `Secure`	false
sessionManager.sessionIdCookie	Реализация cookie для sessionManager	$cookie

Shiro SSO Knox

Параметр	**Описание**	**Значение по умолчанию**
knoxJwtRealm	Shiro realm для работы с Knox	org.apache.zeppelin.realm.ActiveDirectoryGroupRealm
knoxJwtRealm.providerUrl	URL-адрес сервера Knox. Например: *https://<knox-host>:8443/*	—
knoxJwtRealm.login	URL для выполнения входа в систему (login)	gateway/knoxsso/knoxauth/login.html
knoxJwtRealm.logout	URL для выполнения выхода из системы (logout)	gateway/knoxssout/api/v1/webssout
knoxJwtRealm.logoutAPI	Активирует действие выход из системы (logout), используя API	true
knoxJwtRealm.redirectParam	URL для редиректа клиента после выхода из системы	originalUrl
knoxJwtRealm.cookieName	Имя файла cookie, используемого Knox SSO	hadoop-jwt
knoxJwtRealm.publicKeyPath	Путь к открытому ключу (сертификату), используемому Knox	/usr/lib/zeppelin/gateway.cer

sessionManager	Класс, отвечающий за создание, поддержку и очистку всех сессий приложения	org.apache.shiro.web.session.mgt.DefaultWebSessionManager
securityManager.sessionManager	Реализация компонента SessionManager для SecurityManager	$sessionManager
securityManager.sessionManager.globalSessionTimeout	Тайм-аут в миллисекундах, по истечении которого сессия считается истекшей и пользователю необходимо выполнить вход в систему	86400000
shiro.loginUrl	Устанавливает свойство `loginUrl` для стандартных фильтров Shiro	/api/login
cookie	Класс, отвечающий за обработку файлов cookie	org.apache.shiro.web.servlet.SimpleCookie
cookie.httpOnly	Определяет, должны ли cookie Shiro иметь атрибут `HttpOnly`	true
cookie.secure	Определяет, должны ли cookie Shiro иметь атрибут `Secure`	false
sessionManager.sessionIdCookie	Реализация cookie для sessionManager	$cookie
cookie.name	Имя сессионного cookie, используемого Shiro. Этот параметр должен быть равен параметру `responseExcludeHeaders` в */usr/lib/knox/data/services/zeppelinui/{version}/service.xml*	WWW-AUTHENTICATE
knoxJwtRealm.groupPrincipalMapping	Определяет правила сопоставления принципалов и групп	group.principal.mapping
knoxJwtRealm.principalMapping	Определяет правила сопоставления принципалов	principal.mapping

Additional configuration Shiro.ini

Параметр	Описание	Значение по умолчанию
Additional main section in shiro.ini	Позволяет добавить пары ключ/значение в секцию `main` файла *shiro.ini*	—
Additional roles section in shiro.ini	Позволяет добавить пары ключ/значение в секцию `roles` файла *shiro.ini*	—
Additional urls section in shiro.ini	Позволяет добавить пары ключ/значение в секцию `urls` файла *shiro.ini*	—

Credential Encryption

Параметр	Описание	Значение по умолчанию
Encryption enable	Включает или выключает функцию шифрования учетных данных. Если опция включена, Zeppelin хранит учетные данные (пароли настроек, учетные данные для взаимодействия с другими сервисами) в зашифрованном виде	false
Credential provider path	Путь к хранилищу ключей JCEKS	jceks://file/etc/hive/conf/hive.jceks
Custom jceks	Укажите значение `true` при использовании кастомного файла JCEKS-хранилища. Укажите значение `false` для использования дефолтного файла хранилища, который генерируется автоматически	false

Другие параметры

Параметр	Описание	Значение по умолчанию
Custom zeppelin-site.xml	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *zeppelin-site.xml*	—
Custom zeppelin-env.sh	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *zeppelin-env.sh*	zeppelin-env.sh
Custom log4j.properties	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *log4j.properties*	log4j.properties

ZooKeeper

Main

Параметр	Описание	Значение по умолчанию
connect	Строка подключения ZooKeeper, используемая другими службами или кластерами. Генерируется автоматически	—
dataDir	Место, где ZooKeeper хранит снепшоты базы данных в памяти и, если не указано иное, журнал транзакций обновлений базы данных	/var/lib/zookeeper

zoo.cfg

Параметр	Описание	Значение по умолчанию
clientPort	Порт для прослушивания клиентских подключений, то есть порт, к которому пытаются подключиться клиенты	2181
admin.serverPort	Порт для встроенного сервера Jetty	5181
admin.enableServer	AdminServer — это встроенный сервер Jetty, который предоставляет HTTP-интерфейс для четырехбуквенных команд. Чтобы использовать системное свойство сервера, установите значение `false`	false
tickTime	Базовая единица времени, используемая в ZooKeeper для heartbeats (в миллисекундах). Минимальный тайм-аут сессии равен `tickTime * 2`	2000
initLimit	Тайм-аут, который ZooKeeper использует для ограничения длительности времени, в течение которого серверы ZooKeeper в quorum подключаются к лидеру	5
syncLimit	Определяет максимальное допустимое отклонение по дате между сервером и лидером	2
maxClientCnxns	Этот параметр ограничивает количество активных подключений с хоста с определенным IP-адресом к одному серверу ZooKeeper	0
autopurge.snapRetainCount	Когда функция автоматической очистки ZooKeeper включена, она сохраняет последние `autopurge.snapRetainCount` снэпшотов и соответствующие журналы транзакций в `dataDir` и `dataLogDir` каталогах соответственно, а остальные удаляет. Минимальное значение `3`	3

autopurge.purgeInterval	Интервал времени, в течение которого должна быть запущена задача очистки (в часах). Значением должно быть положительное целое число (`1` и больше), чтобы включить автоматическую очистку	24
Add key,value	В этой секции вы можете указать значения для кастомных параметров, которые не отображаются в интерфейсе ADCM, но могут присутствовать в конфигурационном файле *zoo.cfg*	—

SSl configuration

Параметр	**Описание**	**Значение по умолчанию**
sslQuorum	Включает шифрование в кворуме	false
serverCnxnFactory	Указывает реализацию `ServerCnxnFactory`. Чтобы использовать серверную связь на основе TLS, установите значение параметра `NettyServerCnxnFactory`	org.apache.zookeeper.server.NettyServerCnxnFactory
ssl.quorum.keyStore.location	Полный путь к файлу хранилища ключей сервера (например: */etc/zookeeper/ssl/keystore.jks*)	—
ssl.quorum.keyStore.password	Пароль, использованный при создании хранилища ключей	—
ssl.quorum.trustStore.location	Указывает полный путь к файлу хранилища доверенных сертификатов сервера (например: */etc/zookeeper/ssl/truststore.jks*)	—
ssl.quorum.trustStore.password	Пароль, используемый при создании хранилища доверенных сертификатов	—
ssl.protocol	Протокол TLS для общения с клиентом	TLSv1.2
ssl.quorum.protocol	Протокол TLS для общения в кворуме	TLSv1.2

Параметр	**Описание**	**Значение по умолчанию**
Myid matching	Сопоставление хостов кворума с внутренними идентификаторами ZooKeeper	—

zookeeper-env.sh

Параметр	**Описание**	**Значение по умолчанию**
ZOO_LOG_DIR	Каталог для хранения логов	/var/log/zookeeper
ZOOPIDFILE	Каталог, где хранится ID процесса ZooKeeper	/var/run/zookeeper/zookeeper_server.pid
SERVER_JVMFLAGS	Используется для установки различных JVM-параметров, например, для сборщика мусора	-Xmx1024m
JAVA	Путь к Java	$JAVA_HOME/bin/java
logback.xml template	Используется для установки уровня логирования и определяет, какие лог-appender следует использовать. Использование лог-appender `CONSOLE` направляет логи в стандартный вывод. Использование `ROLLINGFILE` создает файл *zookeeper.log*, затем этот файл ротируется, а затем срок его действия истекает	—

Monitoring

Prometheus settings

Группа	Параметр	Описание	Значение по умолчанию
—	scrape_interval	Интервал сбора метрик	1m
—	scrape_timeout	Время ожидания данных	10s
—	Password for grafana connection	Пароль пользователя Grafana (`admprom_grafana`) для подключения к Prometheus	—
—	Prometheus users to login/logout to Prometheus	Учетные данные пользователей для входа в веб-интерфейс Prometheus	—
Service parameters	config.file	Путь к конфигурационному файлу сервера Prometheus	/etc/admprom/prometheus/prometheus.yml
	storage.tsdb.path	Путь к базе данных сервера Prometheus	/var/lib/admprom/prometheus
	web.console.libraries	Путь к библиотекам управления консолями	/usr/share/admprom/prometheus/console_libraries
	web.console.templates	Путь к шаблонам консолей сервера Prometheus	/usr/share/admprom/prometheus/consoles
	web.config.file	Указывает, какой файл веб-конфигурации следует загрузить. Файл записывается в YAML-формате	/etc/admprom/prometheus/prometheus-auth.yml
	storage.tsdb.retention.time	Время хранения данных. Поддерживаемые единицы: `y`, `w`, `d`, `h`, `m`, `s`, `ms`	15d
	web.listen-address	Адрес для доступа к веб-интерфейсу Prometheus	0.0.0.0:11200

Grafana settings

Параметр	Описание	Значение по умолчанию
Grafana administrator’s password	Пароль администратора Grafana	—
Grafana listen port	Порт доступа к web-интерфейсу Grafana	11210

Node Exporter settings

Параметр	Описание	Значение по умолчанию
Listen port	Порт доступа к системным метрикам хоста ADH в формате Prometheus	11203
Metrics endpoint	Конечная точка, в которую Node Exporter экспортирует системные метрики в формате Prometheus	/metrics

SSL configuration

Параметр	Описание	Значение по умолчанию
[Prometheus] → Enable SSL	Определяет, используется ли SSL для Prometheus	false
[Prometheus] → Certificate file	Путь к файлу SSL-сертификата сервера Prometheus в формате PEM	/etc/admprom/prometheus/server.crt
[Prometheus] → Private key file	Путь к файлу приватного ключа SSL-сертификата сервера Prometheus	/etc/admprom/prometheus/server.key
[Prometheus] → Certificate authority file	Путь к файлу центра сертификации	/etc/admprom/prometheus/ca.crt
[Grafana] → Enable SSL	Определяет, используется ли SSL для Grafana	false
[Grafana] → Certificate file	Путь к файлу SSL-сертификата сервера Grafana в формате PEM	/etc/admprom/grafana/server.crt
[Grafana] → Private key file	Путь к файлу приватного ключа SSL-сертификата сервера Grafana	/etc/admprom/grafana/server.key
[Grafana] → Certificate authority file	Путь к файлу центра сертификации	/etc/admprom/grafana/ca.crt
[Node-exporter] → Enable SSL	Определяет, используется ли SSL для Node Exporter	false
[Node-exporter] → Certificate file	Путь к файлу SSL-сертификата сервера Node Exporter в формате PEM	/etc/ssl/server.crt
[Node-exporter] → Private key file	Путь к файлу закрытого ключа SSL-сертификата сервера Node Exporter	/etc/ssl/server.key
Set SSL rights for certs/key	Позволяет менять владельца и разрешения для SSL-сертификата и файлов ключей	false

Scrape config for HDFS NameNode (hdfs_namenode_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	hdfs-namenode
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for HDFS DataNode (hdfs_datanode_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	hdfs-datanode

scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for HDFS JournalNode (hdfs_journalnode_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	hdfs-journalnode
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for Hue (hue-scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	hue_exporter
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for Impala Daemon (impala_daemon_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	impala-daemon
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for Impala Statestore (impala_statestore_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	impala-statestore
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for Impala Catalog (impala_catalog_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	impala-catalog
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for Hive Metastore (hive_metastore_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	hive-metastore
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for Hive Server (hive_server_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	hive-server
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for Ozone (ozone_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	ozone_exporter
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for Kyuubi (kyuubi_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	kyuubi_exporter
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for YARN nodemanager (yarn_nodemanager_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	yarn-nodemanager
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for YARN resourcemanager (yarn_resourcemanager_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	yarn-resourcemanager
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Scrape config for YARN timelineserver (yarn_timelineserver_scraper.yml)

Параметр	Описание	Значение по умолчанию
job_name	Имя задачи, в рамках которой будут собираться метрики	yarn-timelineserver
scrape_interval	Как часто следует собирать метрики	—
scrape_timeout	Определяет, как долго отправлять повторные запросы на сбор метрик, если сервис не отвечает. Не может быть больше значения параметра `scrape_interval`	—

Метрики мониторинга ADH

Константин Алпашкин

Содержание

Назад к содержанию

- Типы метрик
- Визуализация метрик
  - Graphite
  - Grafana

В данной статье описаны метрики мониторинга, получаемые от ADH-кластера.

Типы метрик

Все метрики делятся на 2 группы: системные и сервисные. Обе группы метрик отображаются в интерфейсах Graphite/Grafana.

Системные метрики отражают общее состояние хостов кластера и обычно связаны с потреблением ресурсов. Ниже представлен список системных метрик.

Системные метрики

Группа метрик	Описание
cpu	Использование ЦПУ
diskspace	Использование диска
files	Файловая статистика
iostat	Производительность операций ввода/вывода
loadavg	Средние значения загрузки системы
memory	Использование памяти
netstat	Статистика сетевых подключений
network	Производительность сетевых интерфейсов
uptime	Время работы

Сервисные метрики отражают характеристики отдельных сервисов ADH и представлены ниже.

HBase

Компонент	Группа метрик	Имя метрики	Описание
RegionServer	IPC	numActiveHandler	Количество обработчиков RPC, активно обслуживающих запросы
		numCallsInGeneralQueue	Количество текущих запросов пользователей в очереди
		numOpenConnections	Количество открытых соединений на уровне RPC
	JvmMetrics	GcCount	Общее количество итераций GC
		GcCountConcurrentMarkSweep	Количество сборок мусора, выполненных с помощью алгоритма Concurrent Mark Sweep
		GcCountParNew	Количество сборок мусора, выполненных с помощью алгоритма ParNew
		GcTimeMillis	Общее время сборки мусора в мс
		GcTimeMillisConcurrentMarkSweep	Общее время сборки мусора (CMS) в мс
		GcTimeMillisParNew	Общее время сборки мусора (ParNew) в мс
		MemHeapCommittedM	Текущая выделенная память heap в МБ
		MemHeapMaxM	Максимальный размер памяти heap в МБ
		MemHeapUsedM	Используемая память heap в МБ
	Server	Append_95th_percentile	95-й перцентиль значений задержки операций append в RegionServer
		Append_median	Медиана значений задержки операций append в RegionServer
		Delete_95th_percentile	95-й перцентиль значений задержки операций delete в RegionServer
		Delete_median	Медиана значений задержки операций delete в RegionServer
		Get_95th_percentile	95-й перцентиль значений задержки операций get в RegionServer
		Get_median	Медиана значений задержки операций get в RegionServer
		Increment_95th_percentile	95-й перцентиль значений задержки операций increment в RegionServer
		Increment_median	Медиана значений задержки операций increment в RegionServer
		Put_95th_percentile	95-й перцентиль значений задержки операций put в RegionServer
		Put_median	Медиана значений задержки операций put в RegionServer
		ScanTime_95th_percentile	95-й перцентиль значений задержки операций scan в RegionServer
		ScanTime_median	Медиана значений задержки операций scan в RegionServer
		percentFilesLocal	Процент сохраненных данных, которые могут быть прочитаны с локальной DataNode

HDFS

Компонент	Группа метрик	Имя метрики	Описание
datanode	FSDatasetState	NumFailedVolumes	Количество неработающих дисков в кластере
	JvmMetrics	GcCount	Общее количество итераций GC
		GcCountPSMarkSweep	Количество сборок мусора, выполненных с помощью алгоритма PS MarkSweep
		GcCountPSScavenge	Количество сборок мусора, выполненных с помощью алгоритма PS Scavenge
		GcTimeMillis	Общее время сборки мусора в мс
		GcTimeMillisPSMarkSweep	Общее время сборки мусора (PS MarkSweep) в мс
		GcTimeMillisPSScavenge	Общее время сборки мусора (PS Scavenge) в мс
		MemHeapCommittedM	Текущая выделенная память heap в МБ
		MemHeapMaxM	Максимальный размер памяти heap в МБ
		MemHeapUsedM	Используемая память heap в МБ
	RpcActivityForPort	NumOpenConnections	Количество открытых соединений (RPC)
		RpcProcessingTimeAvgTime	Среднее время обработки запросов
		RpcQueueTimeAvgTime	Среднее время ожидания запроса в очереди
namenode	FSNamesystem	BlockCapacity	Объем доступного пространства для хранения блоков
		BlocksTotal	Общее количество блоков данных
		CapacityTotal	Общий объем доступного пространства для хранения данных
		CapacityUsed	Общий объем используемого пространства для хранения данных
		CorruptReplicatedBlocks	Количество поврежденных или повторно скопированных блоков данных в HDFS
		FilesTotal	Общее количество файлов в HDFS
		MissingBlocks	Количество недостающих блоков данных
		UnderReplicatedBlocks	Количество блоков данных, которые не имеют достаточного количества реплик в HDFS
	JvmMetrics	GcCount	Общее количество итераций GC
		GcCountPSMarkSweep	Количество сборок мусора, выполненных с помощью алгоритма PS MarkSweep
		GcCountPSScavenge	Количество сборок мусора, выполненных с помощью алгоритма PS Scavenge
		GcTimeMillis	Общее время сборки мусора в мс
		GcTimeMillisPSMarkSweep	Общее время сборки мусора (PS MarkSweep) в мс
		GcTimeMillisPSScavenge	Общее время сборки мусора (PS Scavenge) в мс
		MemHeapCommittedM	Текущая выделенная память heap в МБ
		MemHeapMaxM	Максимальный размер памяти heap в МБ
		MemHeapUsedM	Используемая память heap в МБ
	RpcActivityForPort	NumOpenConnections	Количество открытых соединений (RPC)
		RpcProcessingTimeAvgTime	Среднее время обработки запросов
		RpcQueueTimeAvgTime	Среднее время ожидания запроса в очереди

Hive

Компонент	Группа метрик	Имя метрики	Описание
hiveserver2	ClassLoading	LoadedClassCount	Количество загруженных классов
	GarbageCollector	PSMarkSweep	Время, затраченное на выполнение сборки мусора с помощью алгоритма PS MarkSweep
		PSScavenge	Время, затраченное на выполнение сборки мусора с помощью алгоритма PS Scavenge
	HS2	active_calls_api_Driver_execute	Метрики `active_calls_api_<method_name>` показывают количество вызовов соответствующего метода на текущий момент времени
		active_calls_api_Driver_run	
		active_calls_api_PostHook_org_apache_hadoop_hive_ql_stats_OperatorStatsReaderHook	
		active_calls_api_compile	
		active_calls_api_hs2_operation_INITIALIZED	Количество активных операций в HiveServer2 (HS2), которые находятся в соответствующем статусе
		active_calls_api_hs2_operation_PENDING	
		active_calls_api_hs2_operation_RUNNING	
		active_calls_api_hs2_sql_operation_PENDING	Количество активных SQL-операций в HiveServer2 (HS2), которые находятся в соответствующем статусе
		active_calls_api_hs2_sql_operation_RUNNING	
		active_calls_api_parse	Количество запросов, отправленных к HiveServer через API, которые были успешно распарсены и готовы к выполнению
		active_calls_api_releaseLocks	Метрики `active_calls_api_<method_name>` показывают количество вызовов соответствующего метода в текущий момент времени
		active_calls_api_runTasks	
		active_calls_api_semanticAnalyze	
		active_calls_api_waitCompile	
		active_calls_hs2_compiling_queries	Количество запросов, которые находятся в процессе компиляции
		active_calls_hs2_executing_queries	Количество запросов, которые находятся в процессе выполнения
		active_calls_hs2_submitted_queries	Количество запросов, которые были отправлены на выполнение
		api_Driver_execute	Метрики `api_<method_name>` показывают количество вызовов соответствующего метода
		api_Driver_run	
		api_PostHook_org_apache_hadoop_hive_ql_stats_OperatorStatsReaderHook	
		api_compile	
		api_hs2_operation_INITIALIZED	
		api_hs2_operation_PENDING	
		api_hs2_operation_RUNNING	
		api_hs2_sql_operation_PENDING	
		api_hs2_sql_operation_RUNNING	
		api_parse	
		api_releaseLocks	
		api_runTasks	
		api_semanticAnalyze	
		api_waitCompile	
		cumulative_connection_count	Общее количество соединений с HiveServer2 с момента запуска сервера
		exec_async_pool_size	Размер пула асинхронных потоков HiveServer2
		exec_async_queue_size	Размер очереди асинхронных операций HiveServer2
		hs2_active_sessions	Количество активных сессий на сервере HiveServer2
		hs2_compiling_queries	Количество запросов, находящихся в процессе компиляции на сервере HiveServer2
		hs2_completed_operation_CLOSED	Количество завершенных операций с соответствующим статусом
		hs2_completed_operation_FINISHED	
		hs2_completed_sql_operation_CLOSED	Количество завершенных SQL-операций с соответствующим статусом
		hs2_completed_sql_operation_FINISHED	
		hs2_executing_queries	Количество запросов, выполняющихся в данный момент на сервере HiveServer2
		hs2_open_sessions	Количество открытых сессий
		hs2_sql_operation_active_user	Текущее количество активных пользователей, выполняющих SQL-операции на сервере HiveServer2
		hs2_submitted_queries	Количество запросов, отправленных на сервер HiveServer2
		hs2_succeeded_queries	Количество запросов, успешно обработанных сервером HiveServer2
		buffers_direct_count	Метрики JVM
		buffers_direct_used	
		buffers_mapped_capacity	
		buffers_mapped_count	
		buffers_mapped_used	
		classLoading_loaded	
		classLoading_unloaded	
		gc_PS-MarkSweep_count	
		gc_PS-MarkSweep_time	
		gc_PS-Scavenge_count	
		gc_PS-Scavenge_time	
		jvm_pause_extraSleepTime	
		memory_heap_committed	
		memory_heap_init	
		memory_heap_max	
		memory_heap_usage	
		memory_heap_used	
		memory_non-heap_committed	
		memory_non-heap_init	
		memory_non-heap_max	
		memory_non-heap_usage	
		memory_non-heap_used	
		memory_pools_Code-Cache_usage	
		memory_pools_Compressed-Class-Space_usage	
		memory_pools_Metaspace_usage	
		memory_pools_PS-Eden-Space_usage	
		memory_pools_PS-Old-Gen_usage	
		memory_pools_PS-Survivor-Space_usage	
		memory_total_committed	
		memory_total_init	
		memory_total_max	
		memory_total_used	
		open_connections	
		open_operations	
		qc_current_size	
		qc_max_size	
		threads_blocked_count	
		threads_count	
		threads_daemon_count	
		threads_deadlock_count	
		threads_new_count	
		threads_runnable_count	
		threads_terminated_count	
		threads_timed_waiting_count	
		threads_waiting_count	
		waiting_compile_ops	
	Memory	HeapMemoryUsage_committed	Объем памяти, выделенной для heap
		HeapMemoryUsage_init	Начальный объем памяти для heap
		HeapMemoryUsage_max	Максимальный объем памяти для heap
		HeapMemoryUsage_used	Используемый объем памяти для heap
		NonHeapMemoryUsage_committed	Объем памяти, выделенной для non-heap областей в JVM
		NonHeapMemoryUsage_init	Начальный объем памяти, выделенной для non-heap областей
		NonHeapMemoryUsage_max	Максимальный объем памяти, выделенной для non-heap областей в JVM
		NonHeapMemoryUsage_used	Используемый объем памяти для non-heap областей в JVM
	MemoryPool	CodeCache	Размер кодового кеша (Code Cache)
		CompressedClassSpace	Размер области сжатых указателей (Compressed class space)
		Metaspace	Размер метаспейса (Metaspace)
		PSEdenSpace	Размер пространства Eden в Spark History Server с помощью алгоритма PS
		PSOldGen	Размер старшего поколения (Old Generation) в Spark History Server с помощью алгоритма PS
		PSSurvivorSpace	Размер Survivor-пространства в Spark History Server с помощью алгоритма PS
	OperatingSystem	ProcessCpuLoad	Загрузка процессора
	Threading	DaemonThreadCount	Количество демон-потоков
		PeakThreadCount	Максимальное количество потоков
		ThreadCount	Текущее количество потоков
		TotalStartedThreadCount	Общее количество потоков
metastore	ClassLoading	LoadedClassCount	Количество загруженных классов
	GarbageCollector	PSMarkSweep	Время, затраченное на выполнение сборки мусора с помощью алгоритма PS MarkSweep
		PSScavenge	Время, затраченное на выполнение сборки мусора с помощью алгоритма PS Scavenge
	Memory	HeapMemoryUsage_committed	Объем памяти, выделенной для heap
		HeapMemoryUsage_init	Начальный объем памяти для heap
		HeapMemoryUsage_max	Максимальный объем памяти для heap
		HeapMemoryUsage_used	Используемый объем памяти для heap
		NonHeapMemoryUsage_committed	Объем памяти, выделенной для non-heap областей в JVM
		NonHeapMemoryUsage_init	Начальный объем памяти, выделенной для non-heap областей
		NonHeapMemoryUsage_max	Максимальный объем памяти, выделенной для non-heap областей в JVM
		NonHeapMemoryUsage_used	Используемый объем памяти для non-heap областей в JVM
	MemoryPool	CodeCache	Размер кодового кеша (Code Cache)
		CompressedClassSpace	Размер области сжатых указателей (Compressed class space)
		Metaspace	Размер метаспейса (Metaspace)
		PSEdenSpace	Размер пространства Eden в Spark History Server с помощью алгоритма PS
		PSOldGen	Размер старшего поколения (Old Generation) в Spark History Server с помощью алгоритма PS
		PSSurvivorSpace	Размер Survivor-пространства в Spark History Server с помощью алгоритма PS
	OperatingSystem	ProcessCpuLoad	Загрузка процессора
	Threading	DaemonThreadCount	Количество демон-потоков
		PeakThreadCount	Максимальное количество потоков
		ThreadCount	Текущее количество потоков
		TotalStartedThreadCount	Общее количество потоков
	metastore	PS-MarkSweep_count	Количество итераций GC, выполненных с помощью алгоритма Concurrent Mark Sweep
		PS-MarkSweep_time	Время, затраченное на выполнение сборки мусора с помощью алгоритма PS MarkSweep
		PS-Scavenge_count	Количество итераций GC, выполненных с помощью алгоритма PS Scavenge
		PS-Scavenge_time	Время, затраченное на выполнение сборки мусора с помощью алгоритма PS Scavenge
		active_calls_create_table	Метрики `active_calls_api_<method_name>` показывают количество вызовов соответствующего метода на текущий момент времени
		active_calls_drop_table	
		active_calls_get_all_functions	
		active_calls_get_config_value	
		active_calls_get_database	
		active_calls_get_databases	
		active_calls_get_functions	
		active_calls_get_multi_table	
		active_calls_get_table	
		active_calls_get_tables	
		active_calls_get_tables_by_type	
		api_create_table	Метрики `api_<method_name>` показывают количество вызовов соответствующего метода
		api_create_table_with_environment_context	
		api_drop_table	
		api_drop_table_with_environment_context	
		api_flushCache	
		api_get_all_databases	
		api_get_all_functions	
		api_get_config_value	
		api_get_current_notificationEventId	
		api_get_database	
		api_get_databases	
		api_get_functions	
		api_get_multi_table	
		api_get_next_notification	
		api_get_table	
		api_get_table_objects_by_name_req	
		api_get_table_req	
		api_get_tables	
		api_get_tables_by_type	
		api_init	
		api_set_ugi	
		api_shutdown	
		blocked_count	Количество потоков, которые заблокированы на ожидание блокировки
		create_total_count_dbs	Количество созданных баз данных
		create_total_count_partitions	Количество созданных партиций
		create_total_count_tables	Количество созданных таблиц
		daemon_count	Количество демонов
		deadlock_count	Количество обнаруженных блокировок
		delete_total_count_dbs	Общее количество удаленных баз данных
		delete_total_count_partitions	Общее количество удаленных партиций
		delete_total_count_tables	Общее количество удаленных таблиц
		direct_count	Метрики JVM
		direct_used	
		directsql_errors	
		heap_committed	
		heap_init	
		heap_max	
		heap_usage	
		heap_used	
		jvm_pause_extraSleepTime	
		jvm_pause_info-threshold	
		jvm_pause_warn-threshold	
		loaded	
		mapped_capacity	
		mapped_count	
		mapped_used	
		new_count	
		non-heap_committed	
		non-heap_init	
		non-heap_max	
		non-heap_usage	
		non-heap_used	
		open_connections	
		pools_Code-Cache_usage	
		pools_Compressed-Class-Space_usage	
		pools_Metaspace_usage	
		pools_PS-Eden-Space_usage	
		pools_PS-Old-Gen_usage	
		pools_PS-Survivor-Space_usage	
		runnable_count	
		terminated_count	
		timed_waiting_count	
		total_committed	
		total_count_dbs	
		total_count_partitions	
		total_count_tables	
		total_init	
		total_max	
		total_used	
		unloaded	
		waiting_count	

Spark/Spark3

Компонент	Группа метрик	Имя метрики	Описание
historyserver	ClassLoading	LoadedClassCount	Количество загруженных классов
	GarbageCollector	PSMarkSweep	Время, затраченное на выполнение сборки мусора с помощью алгоритма PS MarkSweep
		PSScavenge	Время, затраченное на выполнение сборки мусора с помощью алгоритма PS Scavenge
	Memory	HeapMemoryUsage_committed	Объем памяти, выделенной для heap
		HeapMemoryUsage_init	Начальный объем памяти для heap
		HeapMemoryUsage_max	Максимальный объем памяти для heap
		HeapMemoryUsage_used	Используемый объем памяти для heap
		NonHeapMemoryUsage_committed	Объем памяти, выделенной для non-heap областей в JVM
		NonHeapMemoryUsage_init	Начальный объем памяти, выделенной для non-heap областей
		NonHeapMemoryUsage_max	Максимальный объем памяти, выделенной для non-heap областей в JVM
		NonHeapMemoryUsage_used	Используемый объем памяти для non-heap областей в JVM
	MemoryPool	CodeCache	Размер кодового кеша (Code Cache)
		CompressedClassSpace	Размер области сжатых указателей (Compressed class space)
		Metaspace	Размер метаспейса (Metaspace)
		PSEdenSpace	Размер пространства Eden в Spark History Server с помощью алгоритма PS
		PSOldGen	Размер старшего поколения (Old Generation) в Spark History Server с помощью алгоритма PS
		PSSurvivorSpace	Размер Survivor-пространства в Spark History Server с помощью алгоритма PS
	OperatingSystem	ProcessCpuLoad	Загрузка процессора
	Threading	DaemonThreadCount	Количество демон-потоков
		PeakThreadCount	Максимальное количество потоков
		ThreadCount	Текущее количество потоков
		TotalStartedThreadCount	Общее количество потоков

YARN

Компонент	Группа метрик	Имя метрики	Описание
historyserver	JvmMetrics	GcCount	Общее количество итераций GC
		GcCountPSMarkSweep	Количество сборок мусора, выполненных с помощью алгоритма PS MarkSweep
		GcCountPSScavenge	Количество сборок мусора, выполненных с помощью алгоритма PS Scavenge
		GcTimeMillis	Общее время сборки мусора в мс
		GcTimeMillisPSMarkSweep	Общее время сборки мусора (PS MarkSweep) в мс
		GcTimeMillisPSScavenge	Общее время сборки мусора (PS Scavenge) в мс
		MemHeapCommittedM	Текущая выделенная память heap в МБ
		MemHeapMaxM	Максимальный размер памяти heap в МБ
		MemHeapUsedM	Используемая память heap в МБ
nodemanager	JvmMetrics	GcCount	Общее количество итераций GC
		GcCountPSMarkSweep	Количество сборок мусора, выполненных с помощью алгоритма PS MarkSweep
		GcCountPSScavenge	Количество сборок мусора, выполненных с помощью алгоритма PS Scavenge
		GcTimeMillis	Общее время сборки мусора в мс
		GcTimeMillisPSMarkSweep	Общее время сборки мусора (PS MarkSweep) в мс
		GcTimeMillisPSScavenge	Общее время сборки мусора (PS Scavenge) в мс
		MemHeapCommittedM	Текущая выделенная память heap в МБ
		MemHeapMaxM	Максимальный размер памяти heap в МБ
		MemHeapUsedM	Используемая память heap в МБ
	NodeManagerMetrics	AllocatedContainers	Количество выделенных контейнеров
		AllocatedGB	Размер выделенной памяти в ГБ
		AllocatedVCores	Количество выделенных ядер
		AvailableGB	Размер доступной памяти в ГБ
		AvailableVCores	Количество доступных ядер
		BadLocalDirs	Количество директорий на локальном диске, которые не могут быть использованы для хранения данных задач из-за ошибок
		BadLogDirs	Количество директорий на локальном диске, которые не могут быть использованы для записи лог-файлов задач из-за ошибок
		ContainerLaunchDurationAvgTime	Среднее время, затраченное на запуск контейнера задачи
		ContainersCompleted	Количество контейнеров задач, которые были успешно завершены
		ContainersFailed	Количество контейнеров задач, которые завершились неудачно
		ContainersIniting	Количество контейнеров задач, которые находятся в процессе инициализации
		ContainersKilled	Количество контейнеров задач, которые были принудительно остановлены
		ContainersLaunched	Количество контейнеров задач, которые были успешно запущены
		ContainersRunning	Количество контейнеров задач, которые в данный момент выполняются
		GoodLocalDirsDiskUtilizationPerc	Процент использования дискового пространства в директориях на локальном диске, которые могут использоваться для хранения данных задач
		GoodLogDirsDiskUtilizationPerc	Процент использования дискового пространства в директориях на локальном диске, которые могут использоваться для записи лог-файлов задач
resourcemanager	JvmMetrics	GcCount	Общее количество итераций GC
		GcCountPSMarkSweep	Количество сборок мусора, выполненных с помощью алгоритма PS MarkSweep
		GcCountPSScavenge	Количество сборок мусора, выполненных с помощью алгоритма PS Scavenge
		GcTimeMillis	Общее время сборки мусора в мс
		GcTimeMillisPSMarkSweep	Общее время сборки мусора (PS MarkSweep) в мс
		GcTimeMillisPSScavenge	Общее время сборки мусора (PS Scavenge) в мс
		MemHeapCommittedM	Текущая выделенная память heap в МБ
		MemHeapMaxM	Максимальный размер памяти heap в МБ
		MemHeapUsedM	Используемая память heap в МБ
	QueueMetrics	AllocatedVCores	Количество выделенных ядер
		AppsFailed	Количество приложений, завершенных ошибкой
		AppsKilled	Количество приложений, "убитых" пользователем
		AppsPending	Количество приложений, ожидающих ресурсов
		AppsRunning	Количество запущенных приложений
		AppsSubmitted	Количество отправленных в очередь приложений
		AvailableVCores	Количество доступных ядер

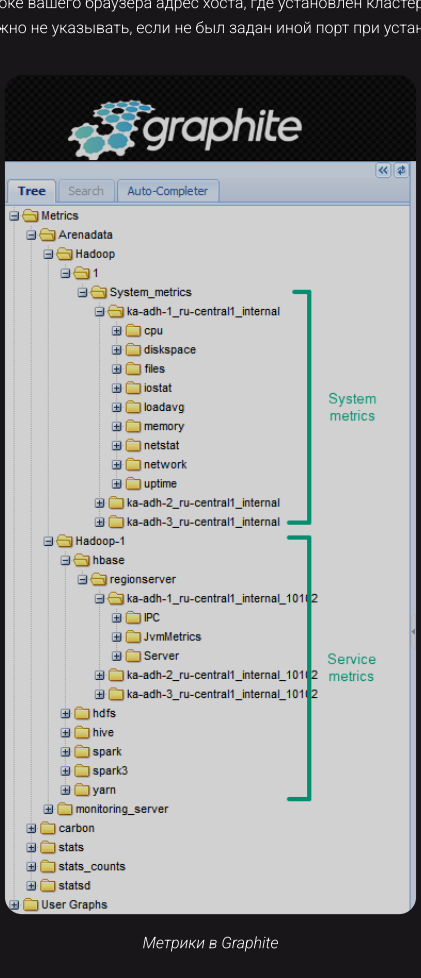
Визуализация метрик

Для просмотра метрик в виде графиков и схем используйте веб-интерфейсы Graphite и Grafana, доступные после установки кластера мониторинга.

Graphite

Graphite — это инструмент для мониторинга, который хранит числовые данные временного ряда (метрики) и может их визуализировать на графиках в веб-интерфейсе.

Для просмотра метрик в Graphite введите в адресной строке вашего браузера адрес хоста, где установлен кластер мониторинга. Например, *http://10.20.30.444:<port>*. По умолчанию Graphite использует порт 80. Данный порт можно не указывать, если не был задан иной порт при установке кластера мониторинга.



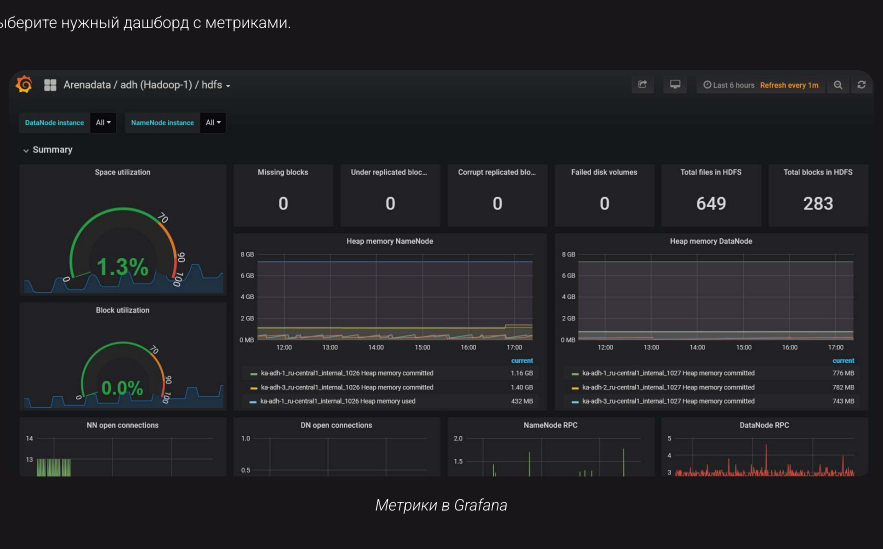
Метрики в Graphite

Grafana

Grafana умеет запрашивать данные из Graphite и визуализировать их.

Чтобы открыть интерфейс Grafana, в адресной строке браузера введите адрес хоста, где установлен кластер мониторинга. Например, *http://10.20.30.444:<port>*. По умолчанию, интерфейс Grafana доступен на 3000 порту.

Затем, кликните ***Home*** и выберите нужный дашборд с метриками.



Метрики в Grafana

Подготовка окружения

Альберт Багдасарян

Назад к содержанию

ADH представляет собой программную платформу для анализа распределенных данных с использованием обычных серверов. В данном разделе мы рекомендуем некоторые принципы надежного размещения этого программного обеспечения с использованием различных аппаратных и программных средств. Очень важно точно определить объем данных, тип и количество задач для кластеров ADH. Существуют метрики для измерения реальной рабочей нагрузки при работе над пилотным проектом. Вы можете масштабировать пилотный кластер, не прибегая к значительным изменениям в уже установленном оборудовании, программном обеспечении, стратегии размещения и сетевых соединениях.

Это руководство включает:

- Требования к ПО
- Требования к оборудованию
- Требования к файловой системе
- Сетевые порты ADH
- Типовые конфигурации

Требования к программному обеспечению

Альберт Багдасарян

Содержание

Назад к содержанию

Таблица требований

Установка Java из репозитория Arenadata

Права пользователя

Полный список требований к программному обеспечению приведен в таблице ниже. Помимо этого, для установки ADH необходимо, чтобы пользователь, устанавливающий его, имел полные права sudo.

Таблица требований

Компонент	Требования
Платформа	Intel x86_64
Операционная система	▪ Centos 7.9. ▪ RHEL 7.9. ▪ AltLinux 8.4 SP (для Enterprise-версии ADH, начиная с 2.1.6.b1). ▪ AltLinux 10 SP (для Enterprise-версии ADH, начиная с 4.1.0.b1). ▪ Astra Linux 1.7 "Орел" SE (для Enterprise-версии ADH, начиная с 3.2.4). Параметру `LC_TIME` должно быть присвоено значение `en_US.utf8`. ▪ Astra Linux 1.7 "Воронеж" (для Enterprise-версии ADH, начиная с 4.1.0.b1). ▪ Ubuntu 22.04.2 LTS. ▪ РЕД ОС 7.3 Сертифицированная редакция (для Enterprise-версии ADH).
Браузер	▪ Internet Explorer ▪ Firefox ▪ Google Chrome ▪ Safari
Программное обеспечение	▪ RPM + YUM / DPKG + APT ▪ scp ▪ curl ▪ unzip ▪ tar
Java	OpenJDK 8u252 или более поздняя версия для всех сервисов, кроме Trino (Java 23), Ozone (Java 21), Flink2 (Java 17) и Spark4 (Java 17)
Arenadata Cluster Manager (ADCM)	См. матрицу совместимости версий
Антивирусное ПО	Рекомендуется отключать антивирусные программы перед установкой и в процессе эксплуатации ADH. Если антивирус все же используется — пожалуйста, уточните настройки, необходимые для использования ADH, у поставщика вашего антивирусного ПО

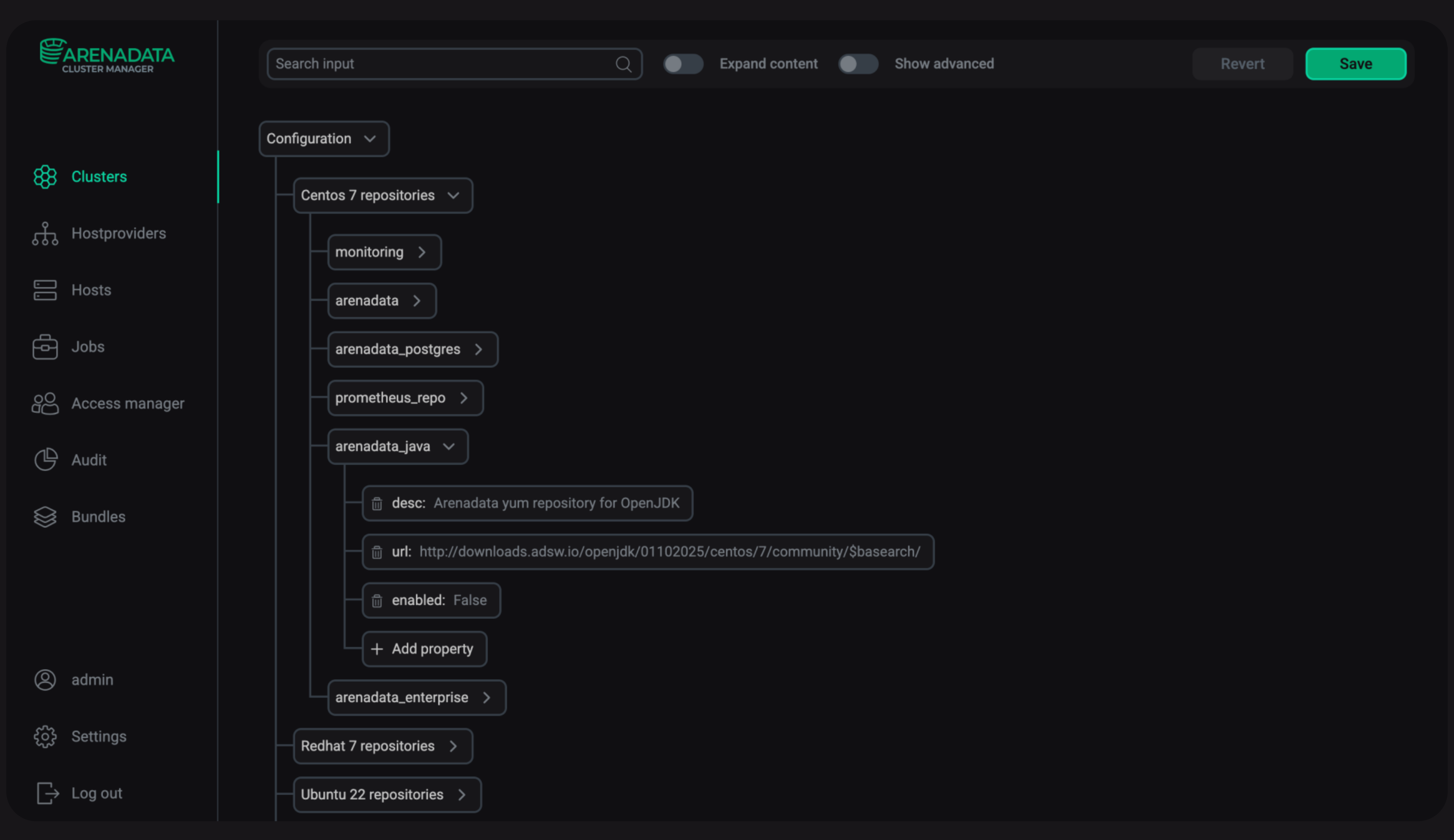
Установка Java из репозитория Arenadata

При установке или обновлении дистрибутива все необходимые версии Java устанавливаются на всех хостах по пути */usr/lib/jvm/*. Это определяется параметром ***ad-runtime-utils*** и управляется через настройки кластера и сервисов. Вы по-прежнему можете использовать версию Java, поставляемую с операционной системой, или установить Java самостоятельно.

Начиная с версии 4.0.0, вы можете устанавливать Java на хосты ADH-кластера из репозитория Arenadata. Соответствующая опция доступна в ADCM на странице настроек кластера и при активации устанавливает на хосты версию JDK, необходимую для работы определенных ADH-сервисов. Данную опцию необходимо активировать перед запуском кластерных действий, таких как ***Install***, ***Expand*** или ***Upgrade***.

Для установки Java из репозитория Arenadata:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Configuration*** и раскройте группу с репозиториями выбранной ОС (например, ***Centos 7 repositories***).
3. Выберите репозиторий ***arenadata_java*** и раскройте его параметры.
4. Укажите значение `true` для параметра ***enabled***.
5. Нажмите ***Apply*** и подтвердите изменения в конфигурации кластера, нажав ***Save***.
6. Выполните необходимое кластерное действие (***Install***, ***Expand*** или ***Upgrade***).



desc: Arenadata yum repository for OpenJDK

url: http://downloads.adsw.io/openjdk/01102025/centos/7/community/$basearch/

enabled: False

Настройка установки Java из репозитория Arenadata для Centos 7

После активации опции необходимая версия Java будет установлена из репозитория Arenadata независимо от того, что указано в переменной JAVA_HOME (группа ***Custom JDK settings***).

Начиная с ADH 4.1.0 вы можете выбрать версию Java для установки на хостах с помощью параметра ***ad-runtime-utils*** в конфигурации кластера. Он содержит файл конфигурации, который определяет, какие версии Java будут использоваться для всего кластера и какие версии будут установлены для отдельных сервисов ADH. Согласно требованиям, для всего кластера устанавливается Java 8, отдельно для Trino — Java 23, для Ozone — Java 21, а для Flink2 и Spark3 — Java 17.

Чтобы изменить версии Java или пути установки:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите в раздел ***Configuration*** и найдите параметр ***ad-runtime-utils***.
3. Отредактируйте файл конфигурации.
4. Нажмите ***Apply*** и подтвердите изменения конфигурации кластера, нажав ***Save***.
5. Выполните необходимое действие кластера (***Install***, ***Expand*** или ***Upgrade***).

ВНИМАНИЕ

Arenadata не является официальным поставщиком JDK, и эта функция предоставляется для обеспечения полностью автоматической установки в системах, не имеющих необходимых зависимостей. В производственных средах рекомендуется использовать JDK от официальных поставщиков.

Права пользователя

Чтобы пользователь обладал необходимыми правами, выполните следующее:

1. Предоставьте пользователю привилегии `root`:

CentOS 7/RHEL 7/РЕД ОС 7.3/Alt Linux SP 8

```
$ sudo usermod -aG wheel <user>
```

где `<user>` — имя пользователя.

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
$ sudo usermod -aG sudo <user>
```

где `<user>` — имя пользователя.

2. Убедитесь, что в файле */etc/sudoers* есть запись, указанная ниже в зависимости от используемой операционной системы, с помощью следующей команды:

```
$ sudo vi /etc/sudoers
```

Чтобы вернуться в терминал, нажмите `Esc`, введите `:q!` и нажмите `Enter`.

CentOS 7/RHEL 7/РЕД ОС 7.3

```
## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)       ALL
```

Ubuntu 22.04/Astra Linux SE 1.7 Орел

```
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Alt Linux SP 8

```
## Uncomment to allow members of group wheel to execute any command
WHEEL_USERS ALL=(ALL:ALL) ALL
```

Требования к оборудованию

Альберт Багдасарян

Содержание

Назад к содержанию

Электропитание и потребление

Компьютерная сеть

Планирование оборудования не может быть полноценным без прогнозирования рабочей нагрузки. Когда вы планируете кластер ADH, очень важно как можно точнее оценить объем данных, а также тип и количество задач. При проведении пилотного проекта применяйте различные метрики для оценки реальной нагрузки на кластер. Это позволит в дальнейшем масштабировать пилотную среду, избегая критических изменений в существующей инфраструктуре.

Количество узлов и их спецификация зависит от нескольких факторов:

- общий объем данных;
- политика хранения данных (по умолчанию коэффициент репликации равен 3);
- тип предполагаемой нагрузки (примеры в разделе Требования к оборудованию в зависимости от типа нагрузки);
- способы хранения данных (контейнерные данные, использование сжатия).

Обратите также внимание на обеспечение пилотного кластера.

Электропитание и потребление

Потребляемая мощность представляет собой важнейший фактор при разработке кластера. Прежде чем приобретать самые большие и быстродействующие физические серверы для реализации узлов кластера, проведите анализ мощности, потребляемой вашим установленным оборудованием. Мы наблюдаем существенную экономию стоимости и потребляемой энергии, если избегать приобретения самых быстрых процессоров, резервных источников питания и другого оборудования.

В настоящее время поставщики предлагают оборудование для облачных центров обработки данных, предусматривающего снижение цены, потребляемой мощности и веса. Компании Supermicro, Dell и HP поставляют подобные линейки продуктов для провайдеров облачных услуг. Поэтому при покупке оборудования для больших кластеров обращайте внимание на такие минимальные конфигурации "облачных серверов".

Для узла DataNode достаточно единственного источника питания, но для узла NameNode используйте вариант с резервированием. Решение с использованием общих источников питания для нескольких серверов повышает надежность без существенного увеличения стоимости.

Некоторые центры размещения оборудования (co-location sites) требуют оплату из расчета максимальной потребляемой мощности, не учитывая реальную потребленную энергию. В таких местах экономичность по мощности последних версий процессоров не очень заметна. Поэтому рекомендуем вам ознакомиться с ценообразованием в таких центрах заранее.

Компьютерная сеть

Это также одна из наиболее значимых систем, поскольку нагрузка в ADH очень изменчивая. Очень важны разумные затраты при обеспечении достаточной скорости взаимодействия между узлами. Типовым является подключение каждого узла к коммутатору в рамках одной стойки на 20 позиций с помощью двухканального интерфейса по 1 Гбит/с и межсоединение (interconnect) 2*10 Гбит/с всех стоек через пару центральных коммутаторов.

Хороший сетевой проект принимает в расчет возможность превышения допустимой пропускной способности в критических узлах сети под реальной нагрузкой. Общепринятые коэффициенты превышения возможности сети (oversubscription ratios) составляют приблизительно 4:1 на уровне доступа к серверу и 2:1 между уровнем доступа и уровнем агрегации или ядром сети. При высоких требованиях к производительности снижайте эти коэффициенты. Кроме того, между стойками возможно превышение на 1 Гбит/с.

Критически важно установить выделенные коммутаторы для кластера вместо попыток организовать виртуальные соединения (virtual connect) в уже существующих коммутаторах. В противном случае нагрузка от кластера ADH будет влиять на остальных пользователей этого оборудования. Также критически важно работать совместно с командой обслуживания сети при выборе коммутаторов, удовлетворяющих требованиям и со стороны ADH, и со стороны средств мониторинга.

Проектируйте сеть таким образом, чтобы оставалась возможность добавлять стойки с серверами для ADH. Исправление сетевых ошибок стоит дорого. Заявленная пропускная способность коммутатора аналогична возможности покрытия расстояния автомобилем (измеряемой в километрах на литр) в том, что вы вряд ли достигнете заявленного показателя. В коммутаторах глубокая буферизация (Deep buffering) предпочтительнее уменьшения задержки. Применение больших пакетов (Jumbo Frames) в пределах кластера повышает пропускную способность за счет использования более эффективных контрольных сумм и может также улучшить обеспечение целостности пакетов.

NameNode

Альберт Багдасарян

Содержание

Назад к содержанию

Средства хранения

Оперативная память

Процессоры

Подключение к сети

Для обеспечения высокой доступности кластера планируйте два узла NameNode: первичный (primary) и вторичный (secondary). Они являются критически важными частями кластера ADH. Необходимо обеспечить надежный доступ к этим узлам. Оба сервера хранят состояние HDFS в файле *fsimage* и журнал операций в файле *edits*.

NameNode выполняет следующие действия:

- операции с файлами в HDFS;
- разделение файлов на блоки, находящиеся в узлах DataNode;
- хранение метаданных о файлах и папках HDFS;
- хранение информации о расположении блоков в узлах DataNode;
- управление репликацией данных.

Вторичный узел NameNode используют в качестве резервного хранилища метаданных HDFS в файле *fsimage* и операций в файле *edits*. Поскольку он менее нагружен, он периодически обновляет файл *fsimage* с помощью журнала *edits*. При этом происходит очищение журнала, предотвращающее таким образом бесконтрольное увеличение последнего.

Средства хранения

В обоих узлах NameNode необходимо иметь надежное хранилище для метаданных и журнала операций. Обычно аппаратный массив RAID или надежное хранилище в сети оправданы для этой цели.

Для узлов NameNode, независимо от количества узлов DataNode, характеристики хранилища неизменны. Устанавливайте четыре дисковых устройства SAS объемом около 1 ТБ каждый вместе с контроллером RAID HDD, настроенным на RAID 1+0. Устройства SAS дороже SATA и обладают меньшей емкостью, но они быстрее и гораздо надежнее.

Объединение устройств SAS в массив RAID обеспечивает сервисы управления ADH достаточно надежным средством хранения с резервированием критически важных данных.

Оперативная память

Требования к памяти существенно зависят от размера кластера ADH. Оперативная память (RAM) является критическим фактором для обоих типов узлов, поскольку они используют RAM как для хранения метаданных HDFS, так и для обработки запросов. По этой причине используйте память с коррекцией ошибок (Error Correction Code, ECC). Обычно каждому узлу NameNode необходимо от 64 ГБ до 128 ГБ памяти.

Требование к объему памяти прямо зависит от количества файловых блоков в HDFS. Как правило, NameNode использует примерно 1 ГБ памяти на миллион блоков HDFS.

ПРИМЕЧАНИЕ

HDFS реплицирует не файлы, а блоки, образуя несколько копий (по умолчанию 3) каждого блока.

Процессоры

Рекомендуется использовать двухпроцессорные материнские платы, каждый процессор с восемью ядрами и тактовой частотой 2.5—3 ГГц. Обычно используют архитектуру Intel.

Подключение к сети

Высокоскоростное подключение к сети крайне необходимо для узлов NameNode. Рекомендуется использовать пару связанных (bonded) каналов по 10 Гбит/с. Она обеспечивает достаточную избыточность и увеличивает пропускную способность до 20 Гбит/с. Для кластеров небольших размеров (менее 50 узлов) достаточно соединения 1 Гбит/с.

DataNode

Альберт Багдасарян

Содержание

Назад к содержанию

Необходимость в RAID

Средства хранения

Оперативная память

Процессоры

Подключение к сети

Узлы DataNode являются главными средствами хранения данных и обеспечения вычислительными ресурсами. Многие полагают, что по этой причине для них необходимо наилучшее оборудование. Однако в основу Hadoop положен принцип использования этих узлов в качестве расходного материала. В этом смысле их производительность должна быть достаточной для выполнения задач кластера, но при невысокой стоимости, так чтобы при выходе из строя их можно было заменить на другие без значительных затрат. Разработчики Hadoop принимают во внимание частоту отказов в больших кластерах как одну из наиболее важных характеристик. В Hadoop решение проблемы отказа оборудования перенесено с оборудования на программное обеспечение.

Узел DataNode выполняет следующие операции:

- запись и чтение блоков данных по запросам от клиентов;
- выполнение задач MapReduce;
- посылка периодических контрольных пакетов (heartbeat) и отчетов о состоянии блоков (block reports) в узел NameNode;
- участие в процессе репликации блоков данных.

Необходимость в RAID

Hadoop обеспечивает резервирование на различных уровнях. Каждый узел DataNode хранит некоторые блоки файлов HDFS, которые имеют множество реплик на других узлах. Поэтому в случае отказа одиночного узла данные остаются доступными. Кластер допускает даже отказ большего количества узлов в зависимости от выбранной вами настройки. Более того, Hadoop позволяет указать распределение узлов по стойкам с тем, чтобы сохранять реплики данных в серверах, входящих в разные стойки. В таком случае существенно увеличивается вероятность сохранности данных даже при выходе из строя целой стойки (хотя строгая гарантия отсутствует). Такой подход исключает необходимость инвестировать в контроллеры RAID для узлов DataNode.

Средства хранения

Обычно один узел DataNode содержит 12-24 дисков с емкостью по 1 ТБ. Вместо объединения их в массив RAID лучше использовать их как простой набор дисков (Just a Bunch of Disks, JBOD). Это обеспечивает более высокую производительность и снижает расходы на оборудование. Не следует беспокоиться по поводу отказа отдельного диска, поскольку HDFS обеспечивает резервирование.

Допустимое количество узлов DataNode зависит от объема оперативной памяти NameNode. Помните, что узел NameNode задействует примерно 1 ГБ памяти на миллион блоков HDFS. По умолчанию размер блока HDFS составляет 128 МБ.

Обычно такие узлы организуют в виде серверов со стандартным форм-фактором (высотой 1U-2U и шириной 19 дюймов) в составе стоек или кабинетов.

Оперативная память

Кроме хранения данных, узел DataNode обрабатывает данные, в том числе, исполняет задачи MapReduce. Задания MapReduce разделены на множество задач, которые исполняют параллельно множество узлов DataNode. Чтобы задание получило согласованный логичный результат, все его задачи должны быть выполнены.

Процессоры

Рекомендуем материнские платы с двумя сокетами под процессоры, каждый процессор с восемью ядрами и тактовой частотой 2.5-3 ГГц. Обычно используют архитектуру Intel.

Подключение к сети

Как и для NameNode, высокоскоростное подключение к сети необходимо для узлов DataNode. Поэтому рекомендуем использовать пару связанных каналов 10 Гбит/с. Эта связанная пара обеспечивает достаточную избыточность и также увеличивает пропускную способность до 20 Гбит/с. Для кластеров небольших размеров (менее 50 узлов) достаточно соединения 1 Гбит/с.

YARN

Альберт Багдасарян

Назад к содержанию

YARN означает Yet Another Resource Negotiator. Основным назначением YARN является предоставление вычислительных ресурсов приложениям, обеспечивающее при этом разделение функций управления ресурсами и планирования заданий. Более подробно это объясняет раздел Архитектура YARN.

В основном рекомендации по оборудованию для менеджера ресурсов YARN такие же, как и для NameNode. В случае, если оба узла NameNode выйдут из строя, можно временно использовать узел менеджера ресурсов для активизации на нем NameNode.

Ноды PostgreSQL, используемые с Hive Metastore

Евгения Кузина

Назад к содержанию

Планирование размера внешней базы данных для Apache Hive Metastore требует тщательного учета различных факторов, таких как количество таблиц, партиций (partitions), пользователей и общая рабочая нагрузка. Несмотря на то, что определить точный размер внешней базы данных достаточно сложно, следующие рекомендации помогут вам принять обоснованное решение:

- **Оценка размера базы данных**
  - **Количество таблиц**. Для каждой таблицы требуется хранилище для схемы и метаданных, которое может быть от нескольких килобайт до нескольких мегабайт на таблицу, в зависимости от сложности схемы таблицы.
  - **Количество партиций**. Требования к объему памяти для хранения партиций зависят от типа данных ключа партиции и количества столбцов партиции, и в среднем может быть оценено примерно как 1 КБ на партицию.
  - **Информация о пользователях и разрешениях**. Для этого требуется небольшой объем памяти, обычно несколько килобайт на пользователя.
- **Темпы роста данных, сложившиеся исторически и планируемый рост**. Учитывайте исторические темпы роста ваших данных и ожидаемый будущий рост. Регулярно отслеживайте размер базы данных и масштабируйте ее по мере необходимости.
- **Репликация и резервное копирование**. Выделите дополнительный объём хранилища для резервного копирования и репликации, чтобы обеспечить избыточность и надежность данных. Рекомендуется оставлять 50% от общего размера базы данных для резервного копирования.
- **Нагрузка**. Оцените количество параллельных запросов и пользователей, которые будут обращаться к системе одновременно. Более высокая рабочая нагрузка может потребовать больше ресурсов и дополнительный объём хранилища данных для кеширования, планирования запросов и временного хранения.
- **Буфер для оптимизации производительности**. Оставьте дополнительное пространство для индексирования, кеширования и других операций оптимизации производительности. Обычно следует оставлять 20-30% от общего размера базы данных.

Хотя эти рекомендации могут помочь вам оценить размер внешней базы данных для Hive Metastore, крайне важно продолжать отслеживать использование базы данных и соответствующим образом корректировать её размер.

ВАЖНО

Системные требования, приведенные ниже, являются **минимальными**. Целевой сайзинг необходимо рассчитывать исходя из требований вашей организации.

Минимальные требования к оборудованию для хостов внешней базы данных перечислены в таблице ниже.

Требование	Небольшой кластер	Средний кластер	Большой кластер
Количество одновременных подключений (сессий пользователей)	5-10	20-50	50+
Процессор	64bit рекомендуется *, 4+ выделенных ядер	64bit, 4-8 ядер	64bit, 8+ ядер Процессоры с большим кешем L3 лучше работают с большим набором данных
RAM	8-16 ГБ	16-32 ГБ	32+ ГБ
Дисковое пространство (минимум один отдельный диск, SSD, RAID 1 или RAID 10)	50+ ГБ	100 ГБ	200+ ГБ

* Если вам нужно использовать 32-битную версию сервера, установите в `LDR_CNTRL` значение `MAXDATA=0xn0000000`, где 1 <= n <= 8, перед запуском сервера PostgreSQL. Попробуйте подобрать подходящее значение и параметры *postgresql.conf*, чтобы найти конфигурацию, работающую удовлетворительно. Для получения дополнительной информации обратитесь к статье Platform-Specific Notes.

Требования к оборудованию в зависимости от типа нагрузки

Альберт Багдасарян

Содержание

Назад к содержанию

- Паттерны нагрузки
  - Равномерность нагрузки
  - Интенсивное вычисление
  - Интенсивное хранение
  - Сбалансированная нагрузка
- Общие рекомендации

Рабочие нагрузки на ADH стоят в основе множества вопросов по управлению ресурсами и возникновению конфликтов. Конфликты могут быть разного рода, например:

- между долгосрочными заданиями, интенсивно использующими ресурсы, и краткосрочными интерактивными запросами;
- между нагрузкой, создаваемой ADH, и нагрузкой от других систем, установленных на узлах одного кластера.

Можно выделить три основных типа (шаблона) нагрузки:

- интенсивное вычисление (compute intensive);
- интенсивное хранение (storage intensive);
- сбалансированная нагрузка (balanced).

Бывают ситуации, когда сначала сложно отнести характер нагрузки к одному из перечисленных выше типов. Начальные действия в кластере обычно значительно отличаются от тех заданий, которые будут выполняться в производственной среде. Поэтому рекомендуется следовать рекомендациям по настройке на сбалансированную нагрузку в период развития пилотного кластера ADH. Затем можно пересмотреть настройки в зависимости от реальной нагрузки.

Паттерны нагрузки

Равномерность нагрузки

Узлы ADH подобны колесам транспортного средства, которые передвигают все остальное. Если колеса одинаковы, то и движение будет равномерным, без рывков. Если они разные, то становится сложным добиться равномерного движения. Поэтому рекомендуется обеспечить одинаковую конфигурацию всех узлов кластера с минимальными различиями между узлами одного типа, то есть между узлами DataNode и между узлами NameNode.

Характеристики следующих компонентов должны быть одинаковыми для всех узлов кластера:

- процессоры;
- оперативная память;
- сетевой интерфейс.

Подробная таблица с рекомендуемыми характеристиками оборудования находится в разделе Общие рекомендации.

Интенсивное вычисление

Этот тип нагрузки связан с работой процессоров и требует большого количества ядер процессора и большого объема оперативной памяти для хранения данных в процессе их обработки. Такой тип нагрузки характерен для процессов обработки естественной речи и других высоконагруженных систем (High-Performance Computing Cluster, HPCC). Для обеспечения высокой скорости вычислений необходимо устанавливать не менее 10 ядер процессоров в каждый узел.

Интенсивное хранение

При таком типе нагрузки рекомендуется больше инвестировать в дисковые устройства внутри каждого узла. Количество узлов и состав дисковых устройств определяется объемом данных для хранения и обработки. Обычно количество устройств хранения одинаково для всех узлов.

Для серверов с низкой нагрузкой (low density server) основной целью является обеспечение низкой стоимости при установке большого количества узлов. Восемь ядер процессора в сервере удовлетворяет этим требованиям и предоставляет достаточную мощность обработки. Каждая задача map или reduce выполняется одним ядром, но поскольку некоторое время уходит на ожидание выполнения операций ввода-вывода, имеет смысл планировать превышение нагрузки на ядра процессора (oversubscription). При наличии 8 ядер в узле можно запланировать создание около 12 слотов под задачи map и reduce.

Оптимальный объем жестокого диска 2-3 ТБ. В среднем устанавливают 12 дисков на узел.

Сбалансированная нагрузка

Логика выбора оборудования не зависит от того, обладает ли кластер высокой или низкой плотностью. Например, можно установить в узел 12 дисковых устройств по 2-3 ТБ или 24 устройства по 1 ТБ. Использование дисков с меньшим объемом предпочтительнее, так как это повышает пропускную способность ввода-вывода и отказоустойчивость. Для обеспечения достаточной вычислительной мощности используют 8 ядер процессора и 128-256 ГБ оперативной памяти в каждом узле.

Общие рекомендации

ВАЖНО

Системные требования, приведенные ниже, являются **минимальными**. Целевой сайзинг необходимо рассчитывать исходя из требований вашей организации.

Следующая таблица содержит рекомендации по подбору оборудования на основе типовых нагрузок кластера ADH.

Сервер	Типовая нагрузка	Хранилище данных	Процессор	Оперативная память	Сетевое оборудование
DataNode	Сбалансированная нагрузка	12 обычных дисков по 2-3 ТБ	8 ядер	128-256 ГБ	1 Гбит/с для подключения, 2x10 Гбит/с межсоединение
	Интенсивные вычисления	12 обычных дисков по 1-2 ТБ	10 ядер	128-256 ГБ	10 Гбит/с для подключения, 2x10 Гбит/с межсоединение
	Преимущественно хранение данных	12 обычных дисков по 4+ ТБ	8 ядер	128-256 ГБ	10 Гбит/с для подключения, 2x10 Гбит/с межсоединение
NameNode	Сбалансированная нагрузка	4+ дисков по 2-3 ТБ RAID 10	8 ядер	128-256 ГБ	10 Гбит/с для подключения, 2x10 Гбит/с межсоединение
Менеджер ресурсов YARN	Сбалансированная нагрузка	4+ дисков по 2-3 ТБ RAID 10	8 ядер	128-256 ГБ	10 Гбит/с для подключения, 2x10 Гбит/с межсоединение

Требования к файловой системе

Евгения Кузина

Содержание

Назад к содержанию

Поддерживаемые файловые системы

Использование опции noatime для повышения производительности

Ограничения использования параметров монтирования файловой системы

Опция umask

Поддерживаемые файловые системы

Файловая система Hadoop Distributed File System (HDFS) предназначена для работы поверх базовой файловой системы операционной системы. Поддерживаются следующие операционные системы:

- **ext3** — наиболее протестированная базовая файловая система для HDFS;
- **ext4** — является масштабируемым расширением ext3;
- **XFS** — файловая система по умолчанию в RHEL 7.

Если вы выбираете между ext3 и ext4, рекомендуется ext4.

Использование опции noatime для повышения производительности

Файловые системы Linux хранят метаданные, которые записывают время доступа к каждому файлу. Это означает, что каждая операция чтения также осуществляет запись на диск. Отключите эту функцию, чтобы ускорить чтение файлов. Для этого добавьте опцию монтирования `noatime` в каждую строку, которая определяет файловую систему в файле */etc/fstab*, например:

```
/dev/sdb1 /data1 xfs defaults,noatime 0
```

Используйте следующую команду, чтобы применить изменения без перезагрузки:

```
$ mount -o remount /data1
```

Ограничения использования параметров монтирования файловой системы

- Опция монтирования `sync` позволяет осуществлять запись синхронно. Использование синхронизации снижает производительность сервисов, записывающих данные на диски (например, HDFS и YARN). В ADH большинство операций записи реплицируются, поэтому синхронная запись на диск не является необходимой, она ресурсозатратна и не дает заметного улучшения стабильности работы кластера. Использовать `sync` не рекомендуется.
- Опции `nfs` и `nas` не поддерживаются при монтировании каталога данных DataNode.
- Монтирование */tmp* как файловой системы с опцией `noexec` не поддерживается. Этот способ используется для предотвращения выполнения хранящихся файлов.

Опция umask

UNIX-системы используют umask (user file-creation mode mask), чтобы устанавливать разрешения по умолчанию для создаваемых файлов и каталогов. В большинстве дистрибутивов Linux значение umask по умолчанию — `0022 (022)` или `0002 (002)`. Базовые права для каталога — `0777 (rwxrwxrwx)`, а для файла — `0666 (rw-rw-rw)`. Чтобы определить права доступа после применения umask, вычтите значение umask из базовых прав. Значение umask `0002` используется для обычного пользователя. При использовании этого значения права по умолчанию для каталога — `775`, а для файла — `664`. Для суперпользователя (root) маска по умолчанию — `0022`. При использовании этой маски права по умолчанию для каталога — `755`, а для файла — `644`.

Вы можете установить umask в файле *etc/bashrc* или */etc/profile*.

ADH поддерживает значения umask, перечисленные в таблице ниже.

Поддерживаемые значения umask

Значение umask	Файл				Каталог			
	Результат	Владелец	Группа	Остальные	Результат	Владелец	Группа	Остальные
0022 (рекомендовано)	644	rw-	r--	r--	755	rwx	r-x	r-x
0002	664	rw-	rw-	r--	775	rwx	rwx	r-x
0000	666	rw-	rw-	rw-	777	rwx	rwx	rwx

Сетевые порты ADH

Евгения Кузина

Содержание

Назад к содержанию

- Сопоставление сетевых портов ADH
  - Arenadata PostgreSQL
  - Airflow2
  - Flink
  - Flink2
  - HBase
  - HBaseThrift2 Server
  - HDFS
  - Hive
  - HUE
  - Impala
  - Kyuubi
  - Monitoring (сервис)
  - Monitoring (кластер)
  - Ozone
  - Solr
  - Spark3
  - SSM
  - Trino
  - YARN
  - Zeppelin
  - Zookeeper
- Порты JMX
- Порты, переопределённые Kerberos
- Клиентские порты
  - Примеры
- Порты ADCM
- Порты Arenadata Enterprise Tools
- Порты Arenadata Monitoring

Сопоставление сетевых портов ADH

В таблицах ниже перечислены сетевые порты компонентов и сервисов ADH. Столбец ***Конфигурирование*** содержит имя файла, в котором порт может быть переопределён.

Arenadata PostgreSQL

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Arenadata PostgreSQL	5432	TCP	Порт для подключения к PostgreSQL	port	postgresql.conf

Airflow2

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Airflow2 Broker	6379	TCP	Порт сервиса Redis	broker_url (redis_port в ADCM)	airflow.cfg
Airflow2 Flower	5555/5555	HTTP/HTTPS	Порт веб-интерфейса Flower	flower_port	airflow.cfg
Airflow2 Scheduler	5432	TCP	Порт сервиса PostgreSQL	sql_alchemy_conn, result_backend	airflow.cfg
Airflow2 Server	8080/8080	HTTP/HTTPS	Порт веб-интерфейса Airflow Server	web_server_port (server_port в ADCM)	airflow.cfg
Airflow2 Worker	8793/8793	HTTP/HTTPS	Порт для доступа к Celery Worker API	worker_log_server_port (worker_port в ADCM)	airflow.cfg

Flink

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Flink History Server	8082	HTTP/HTTPS	Порт веб-интерфейса Flink History Server	historyserver.web.port	flink-conf.yaml
Flink JobManager	8081/8081	HTTP/HTTPS	Порт веб-интерфейса Flink Job Manager	rest.port	Кастомный параметр в flink-conf.yaml
Flink JobManager	6123	TCP	Порт Flink Job Manager RPC (не в режиме HA)	jobmanager.rpc.port	flink-conf.yaml
Flink JobManager	Диапазон, сгенерированный Zookeeper (например, 50100)	TCP	Порт Flink Job Manager (в режиме HA)	high-availability.jobmanager.port	—
Flink SQL-Gateway	8083	TCP	Порт REST-эндпойнта SQL Gateway	sql-gateway.endpoint.rest.port	flink-conf.yaml
Flink TaskManager	Диапазон (например, 35122)	TCP	Порт данных (диапазон)	taskmanager.data.port	—
Flink TaskManager	Диапазон (например, 32390)	IPC	Порт IPC (диапазон)	taskmanager.rpc.port	—

Flink2

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Flink2 History Server	8092	HTTP/HTTPS	Порт веб-интерфейса History Server	historyserver.web.port	config.yaml
Flink2 SQL-Gateway	8093	TCP	Порт REST-эндпойнта SQL Gateway	sql-gateway.endpoint.rest.port	config.yaml

HBase

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
HBase Master Server	16000	IPC	Порт HBase Master IPC	hbase.master.port	Кастомный параметр в hbase-site.xml
HBase Master Server	16010/16010	HTTP/HTTPS	Порт веб-интерфейса HBase Master	hbase.master.info.port	Кастомный параметр в hbase-site.xml
HBase Master Server	9211	HTTP/HTTPS	Порт для метрик Prometheus	Prometheus metrics port	Monitoring
HBase Phoenix Query Server	8765/8765	HTTP/HTTPS	Порт для доступа к API	phoenix.queryserver.http.port	Кастомный параметр в hbase-site.xml
HBase Phoenix Query Server	9213	HTTP	Порт для метрик Prometheus	Prometheus metrics port	Monitoring
HBase REST Server	60080/60080	HTTP/HTTPS	Порт для доступа к API	hbase.rest.port	hbase-site.xml
HBase REST Server	8085	HTTP/HTTPS	Порт веб-интерфейса	hbase.rest.info.port	Кастомный параметр в hbase-site.xml
HBase REST Server	9215	HTTP	Порт для метрик Prometheus	Prometheus metrics port	Monitoring
HBase RegionServer	16030/16030	HTTP/HTTPS	Порт веб-интерфейса HBase RegionServer	hbase.regionserver.info.port	Кастомный параметр в hbase-site.xml
HBase RegionServer	16020	IPC	Порт HBase RegionServer IPC	hbase.regionserver.port	Кастомный параметр в hbase-site.xml
HBase RegionServer	9212	HTTP	Порт для метрик Prometheus	Prometheus metrics port	Monitoring

HBaseThrift2 Server

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
HBase Thrift2 Server	9090	TCP	Порт Thrift2 Server	—	—
HBase Thrift2 Server	9095/9095	HTTP/HTTPS	Порт веб-интерфейса Thrift2 Server	hbase.thrift.info.port	Кастомный параметр в hbase-site.xml
HBase Thrift2 Server	9214	HTTP	Порт для метрик Prometheus	Prometheus metrics port	Monitoring

HDFS

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
HDFS DataNode	9864/9865	HTTP/HTTPS	Порт веб-интерфейса DataNode	dfs.datanode.https.address	hdfs-site.xml
HDFS DataNode	9866	TCP	Data transfer	dfs.datanode.address	hdfs-site.xml
HDFS DataNode	9867	IPC	Операции с метаданными файловой системы	dfs.datanode.ipc.address	hdfs-site.xml
HDFS DataNode	9202	HTTP	Порт для отправки метрик мониторинга	Prometheus metrics port	Monitoring
HDFS HttpFS server	14000/14000	HTTP/HTTPS	Порт HTTPfs API	httpfs.http.port	httpfs-site.xml
HDFS JournalNode	8480/8481	HTTP/HTTPS	Порт веб-интерфейса JournalNode	dfs.journalnode.http.address, dfs.journalnode.https.address	hdfs-site.xml
HDFS JournalNode	8485	IPC	Операции с метаданными файловой системы	dfs.journalnode.rpc-address	hdfs-site.xml
HDFS JournalNode	9203	HTTP	Порт для отправки метрик мониторинга	Prometheus metrics port	Monitoring
HDFS NameNode	9870/9871	HTTP/HTTPS	Порт веб-интерфейса NameNode	dfs.namenode.http-address, dfs.namenode.https-address	hdfs-site.xml
HDFS NameNode	8020	IPC	Операции с метаданными файловой системы	Встроенный в dfs.namenode.rpc-address.<namespace>.<namenode_id>	hdfs-site.xml
HDFS NameNode	9201	HTTP	Порт для отправки метрик мониторинга	Prometheus metrics port	Monitoring
HDFS ZKFC	8019	IPC (TCP, UDP)	ZooKeeper FailoverController для NN HA	dfs.ha.zkfc.port	Кастомный параметр в hdfs-site.xml

Hive

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Hive Server	10000	TCP или HTTP	Порт Thrift/JDBC	hive.server2.thrift.port	hive-site.xml
Hive Server	10002/10002	HTTP/HTTPS	Порт веб-интерфейса Hive Server	hive.server2.webui.port	Кастомный параметр в hive-site.xml
Hive HiveServer2	9208	HTTP	Порт для отправки метрик мониторинга	Prometheus metrics port	Monitoring
Hive Metastore	9083	TCP	Порт сервиса Hive Metastore	hive.metastore.uris	hive-site.xml
Hive Metastore	9207	HTTP	Порт для отправки метрик мониторинга	Prometheus metrics port	Monitoring
Hive Tez UI	8089	HTTP/HTTPS	Порт веб-интерфейса Tez	Встроенный в tez.tez-ui.history-url.base	tez-site.xml

HUE

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
HUE Server	8000/8000	HTTP/HTTPS	Порт веб-интерфейса HUE Server	desktop.http_port	hue.ini
HUE Server	8000	HTTP	Порт для отправки метрик мониторинга	desktop.http_port	hue.ini

Impala

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Impala Catalog Service	23020	TCP	Только для внутреннего использования. Демон Catalog Server слушает этот порт для получения обновлений от демона StateStore	state_store_subscriber_port	/etc/impala/conf/catalogstore.conf
Impala Catalog Service	25020	HTTP/HTTPS	Порт, на котором работает веб-сервер Catalog Service. См. Обзор веб-интерфейсов Impala	webserver_port	/etc/impala/conf/catalogstore.conf
Impala Catalog Service	26000	TCP	Только для внутреннего использования. Порт, который использует Impala Catalog Service для связи с демонами Impala	catalog_service_port	/etc/impala/conf/catalogstore.conf
Impala Catalog Service	25020	HTTP	Порт для отправки метрик мониторинга	webserver_port	/etc/impala/conf/catalogstore.conf
Impala Daemon	21000	TCP	Порт, на котором демоны Impala обслуживают клиентские запросы Beeswax	beeswax_port	/etc/impala/conf/impalastore.conf
Impala Daemon	21050	TCP	Порт, на котором демоны Impala обслуживают клиентские запросы HiveServer2	hs2_port	/etc/impala/conf/impalastore.conf
Impala Daemon	22000	TCP	Только для внутреннего использования. Демоны Impala используют этот порт для связи друг с другом на основе Thrift	be_port	/etc/impala/conf/impalastore.conf
Impala Daemon	23030	TCP	Только для внутреннего использования. Демоны Impala прослушивают этот порт для получения обновлений от демона Statestore	state_store_subscriber_port	/etc/impala/conf/impalastore.conf
Impala Daemon	25000	HTTP/HTTPS	Порт, на котором работает веб-сервер демона Impala. См. Обзор веб-интерфейсов Impala	webserver_port	/etc/impala/conf/impalastore.conf
Impala Daemon	27000	TCP	Только для внутреннего использования. Демоны Impala используют этот порт для связи друг с другом на основе KRPC	krpc_port	/etc/impala/conf/impalastore.conf
Impala Daemon	28000	HTTP	Порт, который используют клиентские приложения для передачи команд и получения результатов по HTTP по протоколу HiveServer2	hs2_http_port	/etc/impala/conf/impalastore.conf
Impala Daemon	25000	HTTP	Порт для отправки метрик мониторинга	webserver_port	/etc/impala/conf/impalastore.conf
Impala Statestore	24000	TCP	Порт, на котором запущен компонент Impala Statestore	state_store_port	/etc/impala/conf/statestore.conf
Impala Statestore	25010	HTTP/HTTPS	Порт, на котором работает веб-сервер Statestore. См. Обзор веб-интерфейсов Impala	webserver_port	/etc/impala/conf/statestore.conf
Impala Statestore	25010	HTTP	Порт для отправки метрик мониторинга	webserver_port	/etc/impala/conf/statestore.conf

Kyuubi

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Kyuubi Server	10099	HTTP/HTTPS	Порт, на котором работает сервис REST	kyuubi.frontend.rest.bind.port	kyuubi-defaults.conf
Kyuubi Server	10009	Бинарный	Порт, на котором работает сервис Thrift через бинарный протокол	kyuubi.frontend.thrift.binary.bind.port	kyuubi-defaults.conf
Kyuubi Server	10010	HTTP/HTTPS	Порт, на котором работает сервис Thrift через HTTP	kyuubi.frontend.thrift.http.bind.port	kyuubi-defaults.conf
Kyuubi Server	10019	HTTP	Порт для отправки метрик мониторинга	kyuubi.metrics.prometheus.port	kyuubi-defaults.conf

Monitoring (сервис)

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Prometheus	11200	HTTP	Порт, используемый Prometheus для входящих запросов (UI, API, метрики)	web.listen-address	Prometheus settings
Grafana	11210	HTTP	Порт, на котором сервер Grafana ожидает входящие запросы	Grafana listen port	Grafana settings
Node exporter	11203	HTTP	Порт, на котором сервер Node exporter ожидает входящие запросы	Listen port	Node Exporter settings

Monitoring (кластер)

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Graphite	2015	TCP	Порт для отправки метрик	port	diamond.conf

Ozone

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Ozone Datanode	9882/9883	HTTP/HTTPS	HTTP-адрес веб-интерфейса Datanode	hdds.datanode.http-address/hdds.datanode.https-address	ozone-site.xml
Ozone Datanode	9855	RPC	Порт для Ratis DataStream на Datanode	hdds.container.ratis.datastream.port	ozone-site.xml
Ozone Manager	9862	RPC	Порт для компонента Ozone Manager	ozone.om.address	ozone-site.xml
Ozone Manager	9874/9875	HTTP/HTTPS	HTTP-адрес веб-интерфейса Ozone Manager	ozone.om.http-address/ozone.om.https-address	ozone-site.xml
Ozone Manager	9858	RPC	Порт для Ratis-коммуникации, используемый Ozone Manager	ozone.om.ratis.port	ozone-site.xml
Ozone Manager	9874	HTTP	Порт для отправки метрик мониторинга	ozone.om.http-address	ozone-site.xml
Ozone Recon	9888/9889	HTTP/HTTPS	HTTP-адрес веб-интерфейса Recon, используемый для просмотра метрик и отчетов кластера Ozone	ozone.recon.http-address/ozone.recon.https-address	ozone-site.xml
Ozone S3G	9878/9879	HTTP/HTTPS	HTTP-адрес шлюза S3, который предоставляет Ozone совместимый с S3 API-интерфейс	ozone.s3g.http-address/ozone.s3g.https-address	ozone-site.xml
Ozone Storage Container Manager	9876/9877	HTTP/HTTPS	HTTP-адрес веб-интерфейса SCM	ozone.scm.http-address/ozone.scm.https-address	ozone-site.xml
Ozone Storage Container Manager	9894	RPC	Порт для Ratis-коммуникации, используемый SCM для высокой доступности	ozone.scm.ratis.port	ozone-site.xml
Ozone Storage Container Manager	9876	HTTP	Порт для отправки метрик мониторинга	ozone.scm.http-address	ozone-site.xml

Solr

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Solr Server	8983/8985	HTTP/HTTPS	Порт для доступа к веб-интерфейсу и API Solr Server	hostPort	solr.xml
Solr prometheus exporter	9983	HTTP	HTTP-порт, прослушиваемый экспортером Solr	Option port	—

Spark3

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Spark3 Connect	15002	HTTP/HTTPS	Порт для gRPC-соединения с Spark3 Connect	spark.connect.grpc.binding.port	spark-defaults.conf
Spark3 History Server	18092	HTTP/HTTPS	Порт веб-интерфейса History Server	spark.history.ui.port, spark.ssl.historyServer.port	spark-defaults.conf
Spark3 Livy Server	8999	HTTP/HTTPS	Порт веб-интерфейса Livy Server	livy.server.port	livy.conf

SSM

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
SSM Agent	7044	RPC	Порт, использующийся SSM-агентами для общения с SSM-сервером	smart.agent.port	smart-site.xml
SSM Agent	7051	RPC	Порт, использующийся SSM-сервером для общения с SSM-агентами	smart.agent.master.port	smart-site.xml
SSM Server	7042	RPC	Порт, использующийся SSM-сервером для взаимодействия с NameNodes	smart.server.rpc.address	smart-site.xml
SSM Server	7045	HTTP/HTTPS	Порт веб-интерфейса SSM	smart.server.rpc.address	smart-site.xml

Trino

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Coordinator	18188	HTTP/HTTPS	HTTP-порт сервера	http-server.http.port	config.properties
Coordinator	9209	HTTP/HTTPS	Порт для метрик Prometheus	Prometheus metrics port	Monitoring
HAProxy	7001	HTTP/HTTPS	HTTP-порт сервера	http-server.http.port	config.properties
Worker	18189	HTTP/HTTPS	HTTP-порт сервера	http-server.http.port	config.properties
Worker	9210	HTTP/HTTPS	Порт для метрик Prometheus	Prometheus metrics port	Monitoring

YARN

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
MapReduce History Server	19888/19890	HTTP/HTTPS	Порт веб-интерфейса HS	mapreduce.jobhistory.webapp.address, mapreduce.jobhistory.webapp.https.address	mapred-site.xml
MapReduce History Server	10020	IPC	Порт сервера MapReduce JobHistory	mapreduce.jobhistory.address	mapred-site.xml
MapReduce History Server	10033	IPC	Порт административного интерфейса сервера History	mapreduce.jobhistory.admin.address	Кастомный параметр в mapred-site.xml
MapReduce History Server	13562	IPC	Порт MapReduce Shuffle	mapreduce.shuffle.port	Кастомный параметр в mapred-site.xml
YARN NodeManager	8042/8044	HTTP/HTTPS	Порт веб-интерфейса NM	8042 — yarn.nodemanager.webapp.address, 8044 — yarn.nodemanager.webapp.https.address	yarn-site.xml (только 8044)
YARN NodeManager	8040	IPC	Порт, на котором localizer IPC	yarn.nodemanager.localizer.address	Кастомный параметр в yarn-site.xml
YARN NodeManager	8041	IPC	Порт менеджера контейнеров в NM	yarn.nodemanager.address	yarn-site.xml
YARN NodeManager	7337	TCP	Порт сервиса Spark shuffle (Сервис выключен по умолчанию)	spark.shuffle.service.port	кастомный spark-defaults.conf
YARN NodeManager	7447	TCP	Порт сервиса Spark shuffle	—	—
YARN NodeManager	9205	HTTP	Порт для отправки метрик мониторинга	Prometheus metrics port	Monitoring
YARN ResourceManager	8088/8090	HTTP/HTTPS	Порт веб-интерфейса RM	yarn.resourcemanager.webapp.address, yarn.resourcemanager.webapp.https.address	yarn-site.xml (только 8090)
YARN ResourceManager	8030	HTTP	Порт интерфейса планировщика	yarn.resourcemanager.scheduler.address	Кастомный параметр в yarn-site.xml
YARN ResourceManager	8031	IPC	Порт трекера ресурсов	yarn.resourcemanager.resource-tracker.address	Кастомный параметр в yarn-site.xml
YARN ResourceManager	8032	IPC	Порт интерфейса диспетчера приложений в RM	yarn.resourcemanager.address	Кастомный параметр в yarn-site.xml
YARN ResourceManager	8033	IPC	Порт административного интерфейса RM	yarn.resourcemanager.admin.address	Кастомный параметр в yarn-site.xml
YARN ResourceManager	9204	HTTP	Порт для отправки метрик мониторинга	Prometheus metrics port	Monitoring
YARN Timeline Server	8188/8190	HTTP/HTTPS	Порт веб-интерфейса TS	yarn.timeline-service.webapp.https.address	yarn-site.xml
YARN Timeline Server	10200	IPC	Порт сервера TS	Встроенный в mapreduce.jobhistory.address	mapred-site.xml
YARN Timeline Server	9206	HTTP	Порт для отправки метрик мониторинга	Prometheus metrics port	Monitoring

Zeppelin

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Zeppelin Server	8180/8180	HTTP/HTTPS	Порт веб-интерфейса Zeppelin	zeppelin.server.port, zeppelin.server.ssl.port	zeppelin-site.xml
Zeppelin Server	8180	HTTP	Порт для метрик Prometheus	zeppelin.server.port	zeppelin-site.xml

Zookeeper

Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
Zookeeper Server	2888	TCP	Порт для координации кластера (только для внутреннего использования в кластере)	server.*	zoo.cfg
Zookeeper Server	3888	TCP	Порт для координации кластера (только для внутреннего использования в кластере)	server.*	zoo.cfg
Zookeeper Server	2181	TCP	Порт для клиентских подключений	clientPort (client_port в ADCM)	zoo.cfg
Zookeeper Server	5181	HTTP	HTTP-интерфейс для доступа к Zookeeper API	admin.serverPort (admin.serverPort в ADCM)	zoo.cfg
Zookeeper Server	7000	HTTP	Порт для метрик Prometheus	metricsProvider.httpPort	zoo.cfg

Порты JMX

В пользовательском интерфейсе ADCM определен только порт YARN RM. Другие порты JMX жестко закодированы в конфигурационных файлах среды сервисов.

Сервис	Компонент/Сервер	Порт	Протокол	Описание	Параметр	Конфигурирование
HBase	HBase Region	10102	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	hbase-regionserver-env.sh
HDFS	NameNode	1026	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	hadoop-env.sh
HDFS	DataNode	1027	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	hadoop-env.sh
Hive	Hive Server	1031	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	hive-server2-env.sh
Hive	Metastore	1032	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	hive-metastore-env.sh
Monitoring	jmxtrans	2101	TCP	Порт JMX для клиентских соединений	-Dcom.sun.management.jmxremote.port	/etc/jmxtrans/wrapper.conf
Spark	History Server	18083	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	spark-env.sh
Spark3	History Server	18093	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	spark-env.sh
YARN	Resource Manager	1028	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port (resource_manager_jmx_port в ADCM)	yarn-env.sh
YARN	NodeManager	1029	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	yarn-env.sh
YARN	MapReduce History Server	1030	TCP	Порт JMX (внутреннее использование с jmxtrans)	-Dcom.sun.management.jmxremote.port	mapred-env.sh

Порты, переопределённые Kerberos

Когда Kerberos включен, ADCM переопределяет порты, указанные в таблице ниже, значениями, определенными в конфигурационных файлах Ansible.

При отключении Kerberos ADCM возвращает для этих параметров значения, указанные по умолчанию.

Сервис	Компонент/Сервер	Порт	Протокол	Описание	Kerberos	Параметр
HDFS	DataNode	1022/50475	HTTP/HTTPS	Порт веб-интерфейса DataNode / защищённый порт	Включен	dfs.datanode.http.address, dfs.datanode.https.address
HDFS	DataNode	1019	TCP	Порт обмена данными	Включен	dfs.datanode.address

Клиентские порты

Клиентские порты генерируются в локальном диапазоне портов. Можно определить порт из этого диапазона, зная порт сервиса. Используйте команду lsof следующим образом:

```
$ sudo lsof -i -P -n | grep <известный номер порта>
```

Примеры

1. Клиент Diamond работает на локальном произвольном порту. Он подключается к Graphite, который прослушивает порт `2015`.

```
$ sudo lsof -i -P -n | grep 2015
```

Результат:

```
python    6000       root    2u  IPv4 4811652      0t0  TCP 10.92.16.232:34758->10.92.16.30:2015 (ESTABLISHED)
```

2. Клиент Hive работает на локальном произвольном порту. Он подключается к ADPG, который прослушивает порт `5432`.

```
$ sudo lsof -i -P -n | grep 5432
```

Результат:

```
java       1161       hive  783u  IPv6 2043472      0t0  TCP 10.92.41.49:58094->10.92.41.242:5432 (ESTABLISHED)
```

Примеры клиентских портов в локальном диапазоне портов

Компонент	Порт	Протокол	Описание
Diamond (python-процесс, запущенный под пользователем root)	34758 или другой случайный порт из локального диапазона портов	TCP	Локальный клиентский порт jmxtrans для отправки метрик кластеру мониторинга на порте 2015 (Graphite)
Hive	58094 или другой случайный порт из локального диапазона портов	TCP	Локальный клиентский порт для подключения к базе ADPG на порте 5432
HBase, Spark и т.д.	50382 или другой случайный порт из локального диапазона портов	TCP	Локальный клиентский порт для подключения к HDFS на порте 8020
Flink JobManager	43162 или другой случайный порт из локального диапазона портов	TCP	Клиент Flink подключается к Zookeeper: TCP 10.92.17.55:42844→10.92.17.55:2181 Внутреннее подключение Flink между нодами JobManager: TCP 10.92.17.55:60390→10.92.17.58:33769

Порты ADCM

Порт	Протокол	Описание
22	TCP	Порт подключения ADCM

Порты Arenadata Enterprise Tools

Порт	Протокол	Описание
443	TCP	Порт, используемый Docker Registry
81	TCP	Порт, используемый для доступа к репозиториям в случае offline-установки

Порты Arenadata Monitoring

Порт	Протокол	Описание
3000	HTTP	HTTP-порт доступа к web-интерфейсу Grafana. Значение по умолчанию может быть изменено при установке кластера мониторинга на странице конфигурации сервиса Grafana в разделе ***ip_and_ports***
80	TCP	TCP-порт доступа к web-интерфейсу Graphite. Значение по умолчанию может быть изменено при установке кластера мониторинга на странице конфигурации сервиса Graphite в разделе ***ip_and_ports***

Пилотный кластер

Альберт Багдасарян

Содержание

Назад к содержанию

Параметры

Оборудование

- Объем данных
- Количество узлов DataNode
- Процессор и память

Ниже приведен пример планирования пилотного кластера ADH на основе сбалансированного типа нагрузки (balanced workload pattern). Даже небольшой и простой кластер требует не менее трех узлов DataNode и один NameNode. Можно использовать один физический сервер (или виртуальную машину) или различные машины.

Более подробная информация о нагрузке приведена в руководстве Требования к оборудованию в зависимости от типа нагрузки.

Параметры

Для примера пилотного кластера будут использованы следующие условия:

- Объем данных — 500 ТБ.
- Коэффициент репликации — 3.
- Период сохранения данных — один год.
- Характер нагрузки — сбалансированный.
- Распределение по форматам данных:
  - Простой текст, AVRO, Parquet, Jason, ORC и другие — 20%;
  - Сжатые данные в формате GZIP и Snappy — 80%.

Оборудование

Спецификация на узлы DataNode зависит от объема хранимых и анализируемых данных.

Объем данных

В соответствии с коэффициентом репликации, равному трём, необходим объем $500\text{TB} \cdot 3 = 1500\text{TB}$ для хранения данных в течение одного года. Предположим, что 20% данных находятся в контейнерном формате, а 80% — архивированные данные в формате Parquet, сжатые с помощью Snappy. Эффективность сжатия составляет 70-80%. Для этого примера будет выбрано значение 80%. Расчет объема хранилища выглядит следующим образом:

$$\text{Total storage required for data} = \text{Total storage} \cdot \%\text{ in container storage} + \text{Total storage} \cdot \%\text{ in compressed format} \cdot \text{Expected compression ratio}$$

При заданных условиях требуемый объем равен $1500\text{TB} \cdot 0.2 + 1500\text{TB} \cdot 0.8 \cdot (1 - 0.8) = 540\text{TB}$

Кроме учтенных хранимых данных, необходимо также пространство для обработки данных и выполнения других задач. Поэтому следует предусмотреть дополнительное пространство. Например, в среднем каждый день кластер обрабатывает 10% всех хранимых данных, и процесс обработки создает в три раза больше временных данных. Таким образом, потребуется 30% дополнительного объема.

Требуемый объем хранилища для данных и различных видов обработки будет равен $540\text{TB} + 540\text{TB} \cdot 0.3 \cdot 3 = 702\text{TB}$.

Для узлов DataNode рекомендуются обычные дисковые накопители (JBOD). Файловая система требует для служебных целей около 20% от пространства хранения данных. Поэтому необходимо увеличить требования к пространству еще на 20%. Окончательное значение будет равно $702\text{TB} \cdot (1 + 0.2) \cong 842.5\text{TB}$, округленное до 845 ТБ.

Количество узлов DataNode

К примеру, узел DataNode содержит JBOD из 12 дисков, каждый объемом 4 ТБ. Для вычисления количества DataNode, необходимых для хранения 845 ТБ данных, следует использовать формулу $\frac{845\text{TB}}{4\text{TB} \cdot 12} = 18$.

ПРИМЕЧАНИЕ

Нет необходимости устанавливать весь кластер сразу. Например, можно начать с 25% требуемого количества узлов и масштабировать кластер до 100% по мере накопления данных.

Процессор и память

Согласно рекомендациям следующие характеристики удовлетворяют требованиям пилотного кластера:

- 8 ядер процессора;
- 128 ГБ оперативной памяти.

Настройка пользовательской Java

Сергей Тихомиров, Елена Костюченко

Содержание

Назад к содержанию

Установка Java

Изменение настроек Java в Trino

В этой статье описывается установка пользовательской версии Java для конкретного сервиса. В данном случае устанавливается JDK версии 23.0.2 для Trino.

Установка Java

1. Загрузите нужную версию Java на все хосты:

```
$ wget https://download.oracle.com/java/23/archive/jdk-23.0.2_linux-x64_bin.tar.gz
```

2. Создайте директорию к пользовательской версии Java на всех хостах:

```
$ mkdir -p /tmp/jvm-example
```

3. Разархивируйте пакет по созданному пути на всех хостах:

```
$ sudo tar -xzf jdk-23.0.2_linux-x64_bin.tar.gz -C /tmp/jvm-example
```

4. Проверьте установку Java, выполнив следующую команду, чтобы определить, используется ли правильная версия Java:

```
$ /tmp/jvm-example/jdk-23.0.2/bin/java --version
```

Вывод команды:

```
java 23.0.2 2025-01-21
Java(TM) SE Runtime Environment (build 23.0.2+7-58)
Java HotSpot(TM) 64-Bit Server VM (build 23.0.2+7-58, mixed mode, sharing)
```

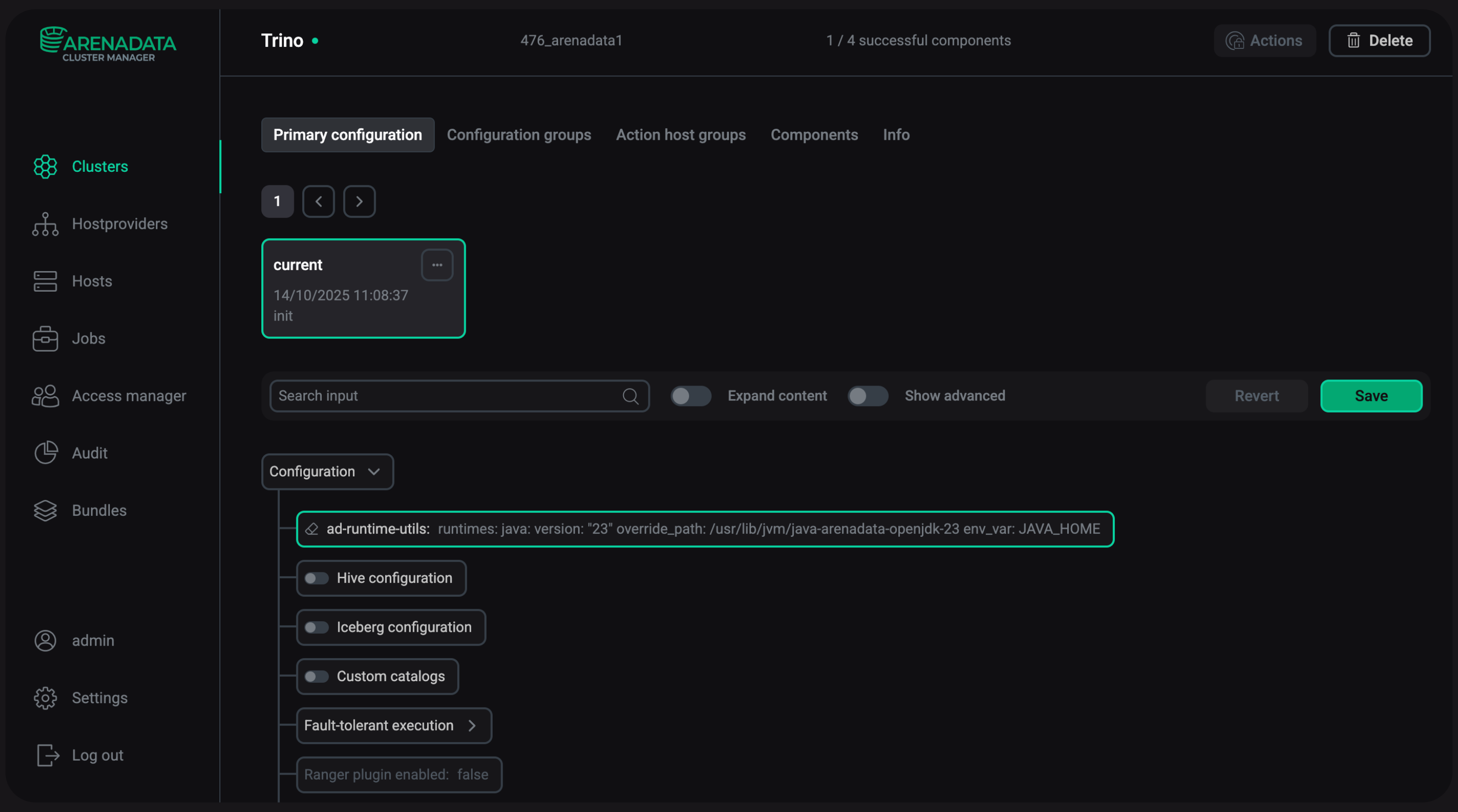
Изменение настроек Java в Trino

Перейдите в интерфейс ADCM и выполните следующие шаги:

1. На странице ***Clusters*** выберите нужный кластер.
2. Перейдите на вкладку ***Services*** и нажмите на ***Trino***.
3. Найдите и отредактируйте параметр ***ad-runtime-utils*** следующим образом:

```
runtimes:
  java:
    version: "23.0.2"
    override_path: /tmp/jvm-example
    env_var: JAVA_HOME
```

4. Подтвердите изменения в конфигурации, нажав ***Save***.
5. Нажмите ***Apply*** и подтвердите изменения в конфигурации кластера, нажав ***Save***.
6. В раскрывающемся меню ***Actions*** выберите ***Restart***, убедитесь, что для параметра ***Apply configs from ADCM*** установлено значение `true`, и нажмите ***Run***.


ARENADATA
CLUSTER MANAGER
Trino
476_arenadata1
1 / 4 successful components
Actions
Delete
Clusters
Hostproviders
Hosts
Jobs
Access manager
Audit
Bundles
admin
Settings
Log out
Primary configuration
Configuration groups
Action host groups
Components
Info
1
Search input
Expand content
Show advanced
Revert
Save
Configuration
Hive configuration
Iceberg configuration
Custom catalogs
Fault-tolerant execution

current
14/10/2025 11:08:37
init

ad-runtime-utils: runtimes: java: version: "23" override_path: /usr/lib/jvm/java-arenadata-openjdk-23 env_var: JAVA_HOME

Ranger plugin enabled: false

Параметр ad-runtime-utils в конфигурации Trino

Airflow

Константин Алпашкин, Дарья Барышева

Содержание

Назад к содержанию

Модель данных
- DAG
- Task
- Operator

Idempotence

Архитектура

Airflow — это платформа, позволяющая разрабатывать, планировать, запускать и отслеживать выполнение сложных процессов обработки данных (workflows). Airflow идеально подходит для описания ETL/ELT-процессов, но также может быть полезен для периодического запуска и мониторинга любых иных задач. Для описания процессов Airflow использует язык программирования Python.

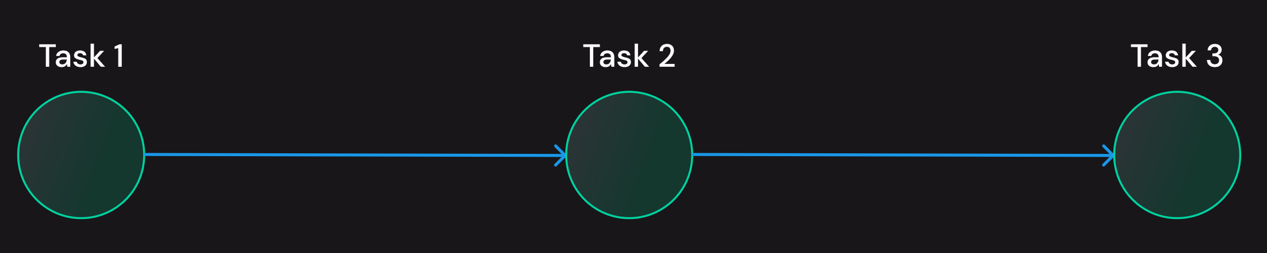
Модель данных

В основе модели данных Airflow лежат следующие объекты.

DAG

Это ключевой объект модели данных Airflow, представляющий семантически связанную группу задач, которые должны выполняться в строго определенном порядке — в соответствии с указанным расписанием. Визуально DAG выглядит как ориентированный ациклический граф (directed acyclic graph), то есть граф без циклических зависимостей.

Простой DAG, изображенный ниже, состоит из трех задач (Task). Обратите внимание, что DAG определяет порядок запуска задач, а также зависимости между ними. Дополнительно Airflow позволяет настроить расписание для обработки DAG, действия в случае неудачного запуска, количество повторных попыток запуска и многое другое. При этом DAG не анализирует, что именно он запускает — за настройку обработки данных отвечают непосредственно сами задачи (Task).



Простой DAG

Task

Задача (Task) — это отдельный узел DAG, который описывает операции, применяемые к данным. Такими операциями могут быть загрузка данных из внешних источников, агрегация и индексирование, удаление дубликатов и другие типовые ETL-операции.

Operator

Оператор (Operator) — это шаблон для выполнения задач. Если задачи описывают применяемые к данным действия, то операторы определяют конкретные действия на уровне кода: с использованием Python-функций, Bash-скриптов и так далее.

Airflow поддерживает множество встроенных операторов. Кроме этого, доступны операторы из community provider packages. Также есть возможность создать свой собственный Operator путем расширения базового класса `BaseOperator`.

Отдельный вид операторов — **Сенсоры (Sensors)** — позволяют обрабатывать реакцию на определенные события: наступление заданного времени, загрузку специфических данных, запуск других DAG/Task и так далее.

Примеры полезных Airflow-операторов приведены ниже.

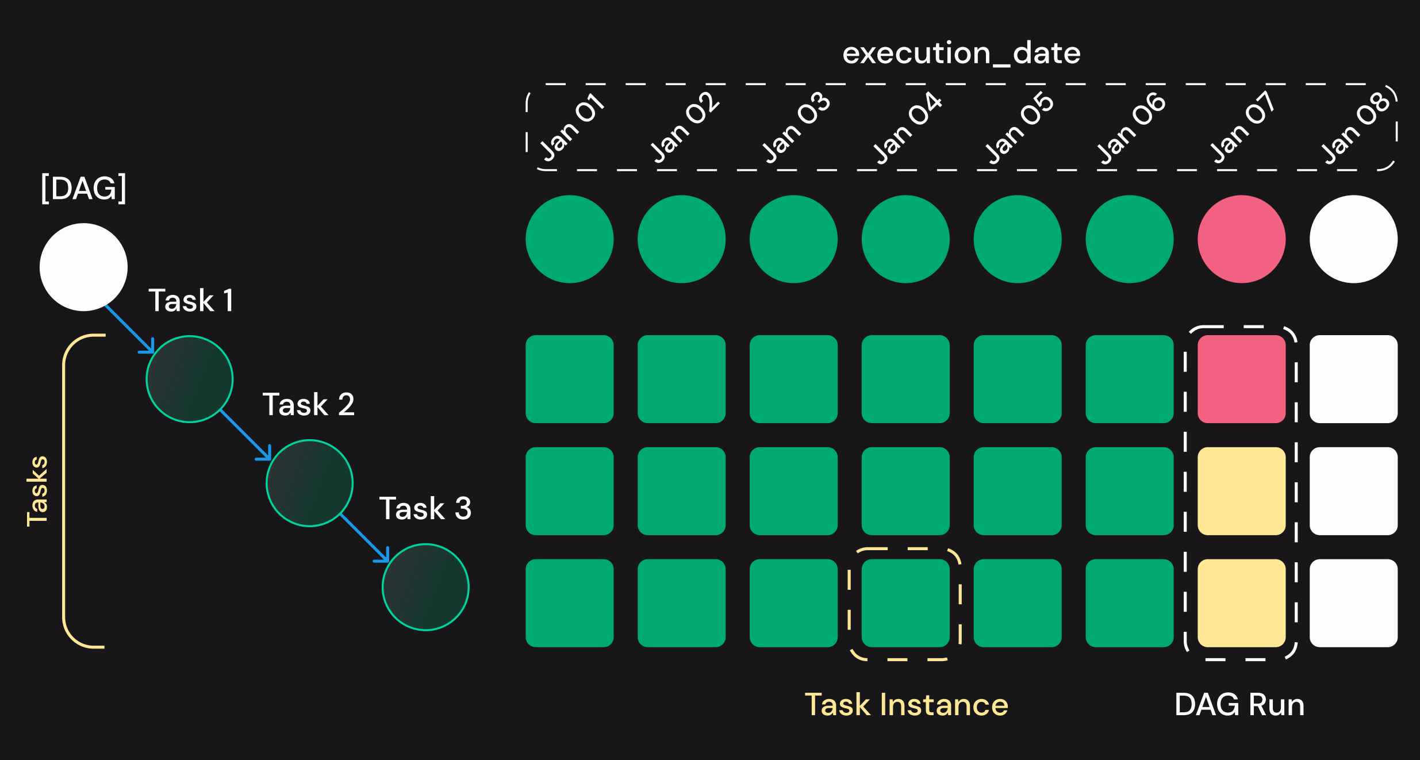
Операторы Airflow

Оператор	Описание
PythonOperator	Выполнение Python-кода
BranchPythonOperator	Выполнение программного кода в зависимости от заданных условий
BashOperator	Запуск Bash-скриптов
SimpleHttpOperator	Отправка HTTP-запросов
PostgresOperator	Отправка запросов в базу данных PostgreSQL
MySqlOperator	Отправка запросов в базу данных MySQL
SqlSensor	Проверка выполнения SQL-запроса
DockerOperator	Запуск Docker-контейнера для задачи (Task)
KubernetesPodOperator	Создание отдельного пода (Pod) Kubernetes для задачи (Task)
EmailOperator	Отправка электронной почты (Email)
DummyOperator	"Пустой" оператор, используемый для группировки задач (Task)

Idempotence

Одна из ключевых концепций, лежащих в основе AirFlow — это хранение информации о каждом запуске DAG в соответствии с настроенным расписанием. Например, если указать, что DAG должен запускаться начиная с 01.01.2022 00:00:00 один раз в сутки, AirFlow будет сохранять информацию о запусках DAG для следующих временных меток: 01.01.2022 00:00:00, 02.01.2022 00:00:00, 03.01.2022 00:00:00 и так далее. Временные метки, приведенные в нашем примере, называются **execution_date**, соответствующие им экземпляры DAG — **DAG Run**, а экземпляры задач (Task), связанные с каждым конкретным DAG Run — **Task Instance**.

Эта концепция крайне важна для поддержания идемпотентности данных (idempotence): запуск или перезапуск задачи с одинаковым набором данных для определенной даты в прошлом всегда приводит к одним и тем же результатам. Дополнительно появляется возможность запускать задачи одного DAG для разных временных меток одновременно.



Концепция Task Instance и Dag Run

Архитектура

Архитектура Airflow основана на следующих компонентах:

- **Web Server**. Предоставляет удобный пользовательский интерфейс, который позволяет запускать DAG и Task, отслеживать их расписание, статус выполнения и так далее.
- **Metadata Database**. Общая база данных, которую используют Scheduler, Executor и Web Server для хранения системных метаданных: глобальных переменных, настроек подключения к источникам данных, статусов запуска Task Instance и DAG Run и так далее. Так как в основе этого компонента лежит библиотека SQLAlchemy, требуется установка совместимой с ней базы данных. Например, MySQL или PostgreSQL.
- **Планировщик (Scheduler)**. Сервис, отвечающий за планирование в Airflow. Он управляет как запуском запланированных процессов по расписанию, так и непосредственной отправкой Task в Executor для выполнения.
  Регулярно (по умолчанию раз в минуту) планировщик анализирует результаты парсинга DAG и проверяет наличие задач, готовых к выполнению. Если необходимые для запуска задач условия соблюдены, планировщик инициализирует создание Task Instance. Для обработки активных задач Scheduler используется Executor, определенный в их настройках.

  Определенные версии Metadata Database (PostgreSQL 9.6+ и MySQL 8+) поддерживают наличие нескольких планировщиков. Их применение обеспечивает максимальную отказоустойчивость системы.
- **Executor**. Механизм, с помощью которого выполняются экземпляры задач (Task Instance). По умолчанию Executor работает внутри планировщика (в рамках одного и того же процесса). Но для production-сред рекомендуется использовать типы Executor, делегирующие выполнение задач в Worker.

  Типы Executor

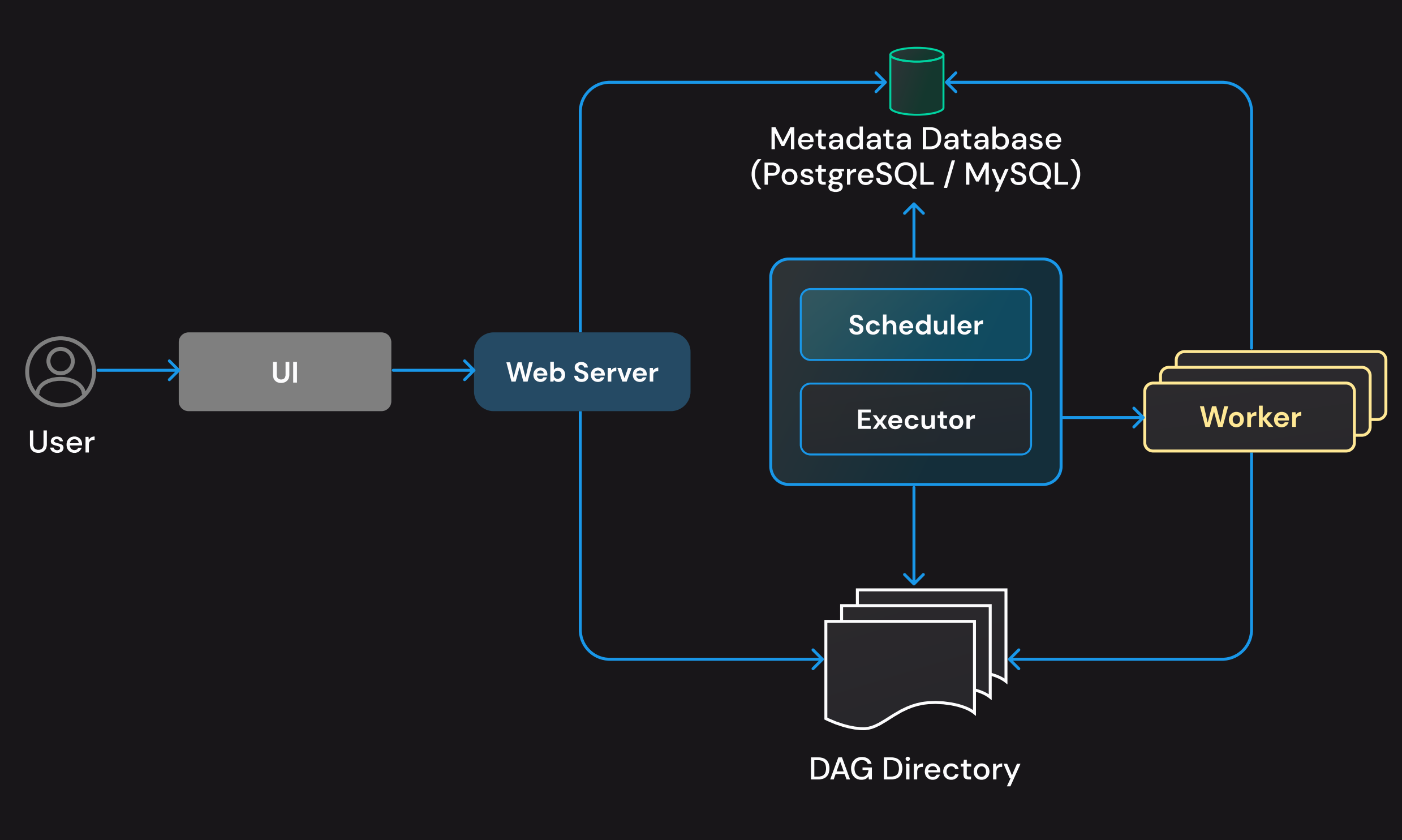
Тип	Описание
SequentialExecutor	Запускает задачи (Task) и приостанавливает планировщик (Scheduler) на время их выполнения. Этот тип Executor рекомендуется использовать только для тестирования — для production-среды он не подходит
LocalExecutor	Запускает дочерний процесс для каждой задачи (Task), что позволяет обрабатывать несколько задач одновременно. Это является отличной имитацией production-среды в тестовом окружении, но также не рекомендуется для реального использования из-за низкой отказоустойчивости: в случае сбоя на машине, на которой запущен Executor, задача не передается на другие узлы
CeleryExecutor	Поддерживает работу множества Worker на различных узлах. Использует библиотеку Celery и требует установки брокера сообщений. Например, Redis или RabbitMQ. При увеличении нагрузки достаточно добавить дополнительный Worker. При сбое любого Worker его задачи передаются работоспособным узлам
DaskExecutor	Аналогичен CeleryExecutor, но вместо Celery для параллельных вычислений использует библиотеку Dask
KubernetesExecutor	Запускает Worker на отдельном поде (Pod) в Kubernetes для каждого экземпляра задачи (Task Instance)
CeleryKubernetesExecutor	Позволяет запускать CeleryExecutor и KubernetesExecutor одновременно. Конкретный тип Executor выбирается в зависимости от текущего статуса очереди задач (Task)
DebugExecutor	Используется для запуска и отладки пайплайнов из IDE

- **Worker**. Отдельный процесс, в котором выполняются задачи (Task). Расположение Worker зависит от выбранного типа Executor.
- **DAG Directory**. Директория с DAG-файлами, доступная для Scheduler, Executor и Worker.

ВАЖНО

Все DAG в форме python-скриптов необходимо размещать в DAG Directory. Web UI предназначен только для мониторинга DAG, не для их добавления либо изменения.

Верхнеуровневая архитектурная схема Airflow изображена ниже. В зависимости от используемых типов Executor эта схема может содержать дополнительные компоненты: например, очередь сообщений (Message Queue) между Executor и Worker в случае CeleryExecutor. Но даже в таких частных случаях можно по-прежнему рассматривать Executor и Worker как единый логический компонент, управляющий выполнением задач (Task).



Архитектура Airflow

ПРИМЕЧАНИЕ

Для получения дополнительной информации по концепциям можно обратиться к документации Airflow.

Обзор работы Flink

Константин Алпашкин

Содержание

Назад к содержанию

- Архитектура
  - Клиент Flink
  - JobManager
  - TaskManager
- Концепции
- Flink History Server
  - History Server REST API

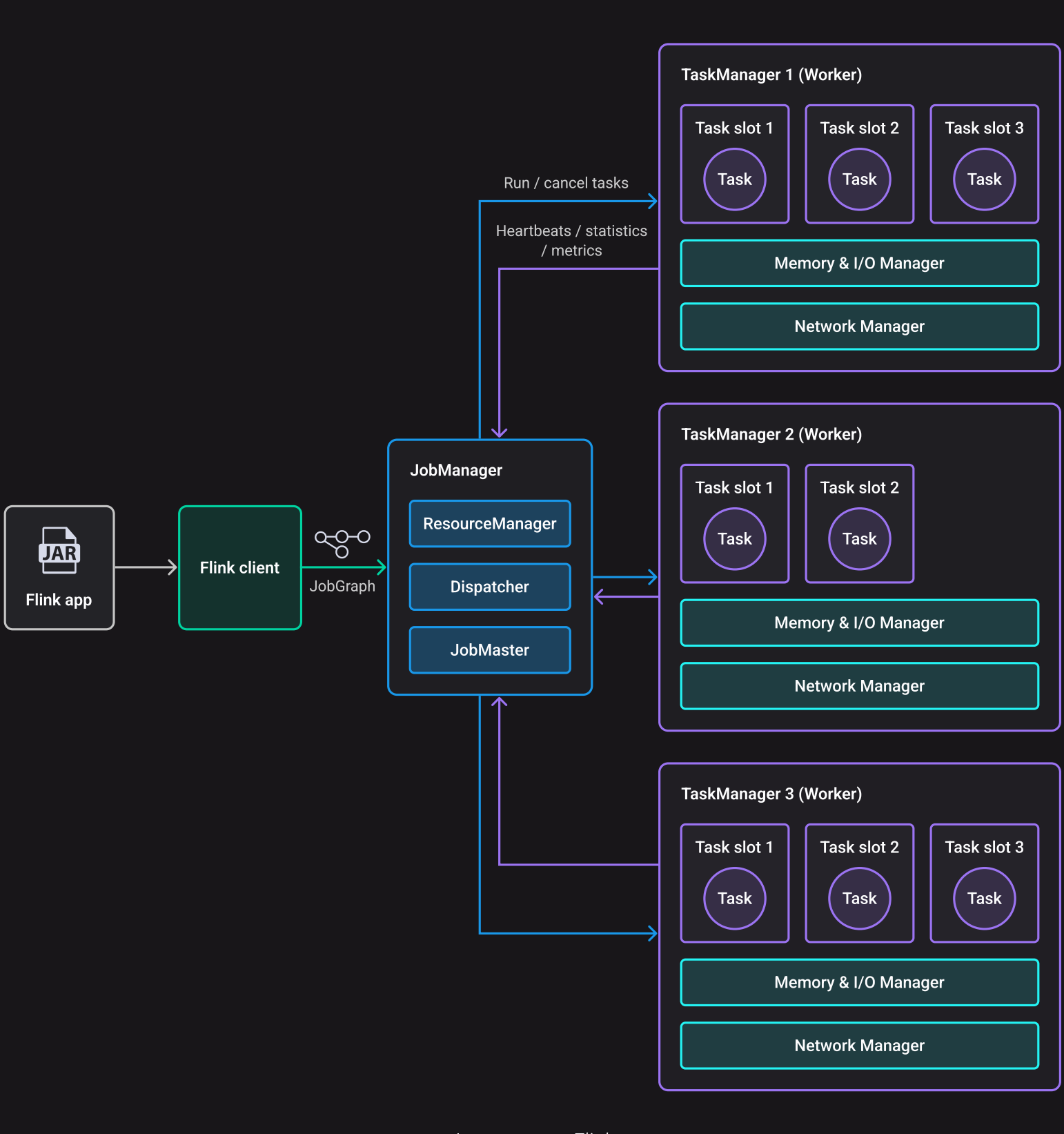
Сервис Flink — это процессинговый движок для потоковой и пакетной обработки данных. По назначению и использованию Flink похож на Spark — он также предлагает stateful/stateless-вычисления, вводит абстракцию SQL над данными, поддерживает режим высокой доступности и предлагает гибкие возможности кластеризации. Однако изначально Flink разрабатывался как фреймворк, ориентированный на работу именно с потоками данных. Flink может работать с ограниченными (bounded) и неограниченными (unbounded) потоками данных, что делает его универсальным выбором для различных приложений. Области применения Flink: аналитика в реальном времени, событийно-ориентированные приложения (event-driven applications), системы распознавания паттернов, аномалий, выявление мошеннических действий, высоконагруженные ETL-пайплайны и так далее.

Flink предоставляет следующие возможности:

- **Унифицированный API для потоковой и пакетной обработки**. Хоть Flink и был разработан в первую очередь для работы с потоками данных, он также может решать задачи пакетной обработки.
- **Обработка с учетом состояния (stateful processing)**. Flink поддерживает обработку данных с учетом состояния, периодически сохраняя состояние приложения на разных этапах работы. Этот подход позволяет Flink обеспечивать принцип обработки "ровно один раз" (exactly once), гарантируя, что данные будут обработаны только один раз, даже в случае сбоев или перезапусков.
- **Окна (windows) и временные операции**. Flink имеет встроенную поддержку различных типов окон, которые являются ключевым инструментом при работе с неограниченными потоками данных. Оконные функции позволяют группировать и агрегировать данные в определенном временном интервале. Flink также позволяет выполнять агрегации типа `SUM`, `AVG`, `COUNT` в контексте окна, отталкиваясь от времени события либо времени обработки.
- **Отказоустойчивость и восстановление**. Flink предоставляет гибкие возможности для обеспечения отказоустойчивости с помощью контрольных точек (checkpoints). Используя контрольные точки, система периодически сохраняет свое состояние с возможностью восстановления из контрольной точки в будущем. Схожий механизм точек сохранения (savepoints) позволяет вручную создавать снепшоты Flink-задач для обновления, приостановки или перезапуска задач с сохранением состояния.
- **SQL для потоков данных**. Flink вводит уровень SQL-абстракции для потоков данных, позволяя выполнять SQL-запросы над потоками, как с обычными таблицами. Flink поддерживает операции фильтрации, объединения, агрегации, а также оконные функции, используя синтаксис SQL.

Архитектура

Ниже представлена высокоуровневая архитектура Flink.



Архитектура Flink

Основные архитектурные компоненты описаны ниже.

Клиент Flink

Преобразует код приложения Flink в **логический граф** (Logical graph, также DataFlow graph или JobGraph) и отправляет его компоненту JobManager. Не является частью среды выполнения Flink. Существует несколько вариантов, которые могут использоваться в качестве клиента в кластере Flink:

- Flink CLI.
- REST-эндпойнт. Flink может получить JAR с задачами в теле REST-запроса.
- SQL-клиент. Позволяет запускать задачи Flink с помощью встроенного CLI-интерфейса для SQL.
- Клиентские приложения на Java/Python, которые программно запускают новые задачи через Flink API.

JobManager

Ключевой компонент, координирующий жизненный цикл приложений Flink. Он управляет планированием задач, отслеживает их состояние, координирует работу контрольных точек, занимается восстановлением системы и так далее. В кластере Flink JobManager существует в одном экземпляре и управляет одним или несколькими TaskManagers.

Процесс JobManager включает следующие компоненты:

- Resource Manager. Отвечает за распределение ресурсов в кластере Flink. Непосредственно управляет слотами задач (task slots) — единицами планирования ресурсов Flink.
- JobMaster. Создается для каждой задачи и отвечает за выполнение одного логического графа Dataflow. В кластере Flink одновременно могут выполняться несколько задач, у каждой из которых есть свой JobMaster.
- Dispatcher. Создает новый экземпляр JobMaster для каждой задачи. Обслуживает REST-эндпойнты для запуска приложений Flink по REST API и предоставляет веб-интерфейс Flink с информацией о выполненных задачах.

TaskManager

Процесс, который запускает одну или несколько подзадач (subtasks) в отдельных потоках. В кластере Flink содержится один или более экземпляров TaskManager, отвечающих за буферизацию данных и обмен данными друг с другом. Наименьшей единицей ресурсов для TaskManager является **слот задачи (task slot)**.

> **РЕКОМЕНДАЦИЯ**
>
> Задачи Flink можно запускать в режиме YARN-кластера, который автоматически создает контейнеры для компонентов JobManager/TaskManager на хостах под управлением YARN. Подробная информация о поддерживаемых режимах и примерах использования доступна в статье Запуск Flink в YARN.

Концепции

Далее описаны основные понятия, наиболее часто использующиеся в контексте Flink. Больше информации доступно в документации Flink.

- **Ограниченные и неограниченные потоки данных**

  Ограниченный (bounded) поток — это поток данных с определенным количеством записей, у которого есть начало и конец. Например, чтение записей из CSV-файла с помощью Flink DataStream API создает ограниченный поток данных. При работе с такими потоками все данные потока можно загрузить в систему (например, для предварительной сортировки) до выполнения вычислений.

  Неограниченный поток данных имеет начало, но может никогда не заканчиваться, непрерывно поставляя новые записи через произвольные промежутки времени. Примеры неограниченного потока данных: брокер Kafka, веб-сокет, отправляющий JSON-события, цифровой интерфейс датчика на производстве и так далее.

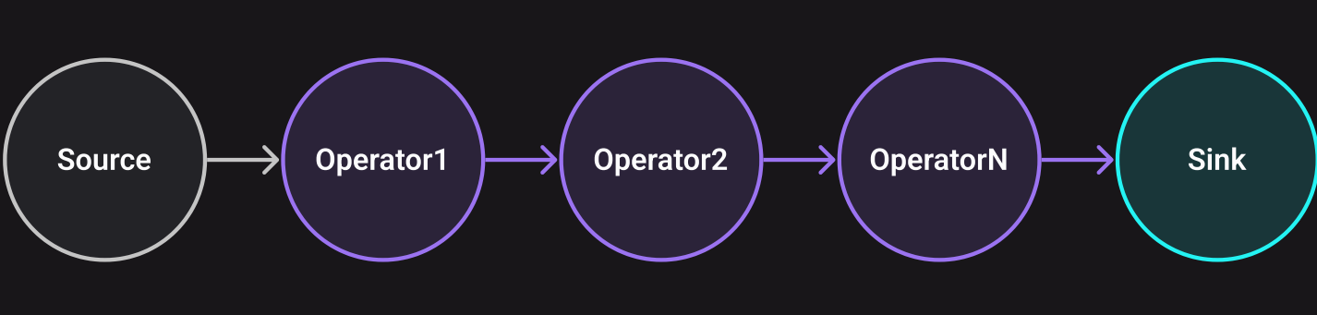
- **Данные состояния и обработка данных с учетом состояния**

  Данные состояния (state data) — это данные типа ключ/значение, которые Flink периодически сохраняет на различных этапах обработки, например, между двумя операторами агрегации. Использование данных состояния предполагает дополнительные затраты на обработку и хранение этих данных. Однако именно данные состояния делают возможными такие операции, как поиск закономерностей в потоке входных данных, агрегация событий в определенном временном интервале (минута/час/дата), восстановление обработки потока без потерь и так далее.

  В кластере Flink данные состояния обрабатываются системой state backend. Выбор реализации этой системы, наиболее подходящей для конкретного Flink-приложения, может помочь минимизировать затраты на доступ к данным состояния.

- **Источники и приемники данных**

  Типичное приложение Flink, работающее с потоками данных, имеет структуру, представленную ниже. Это упрощенное представление, поскольку "под капотом" в JVM Flink автоматически создает новые и при необходимости дублирует объекты источников/приемников/операторов в зависимости от многих факторов, таких как настройки параллелизма, активированные механизмы оптимизации и так далее.

  Источники и приемники данных

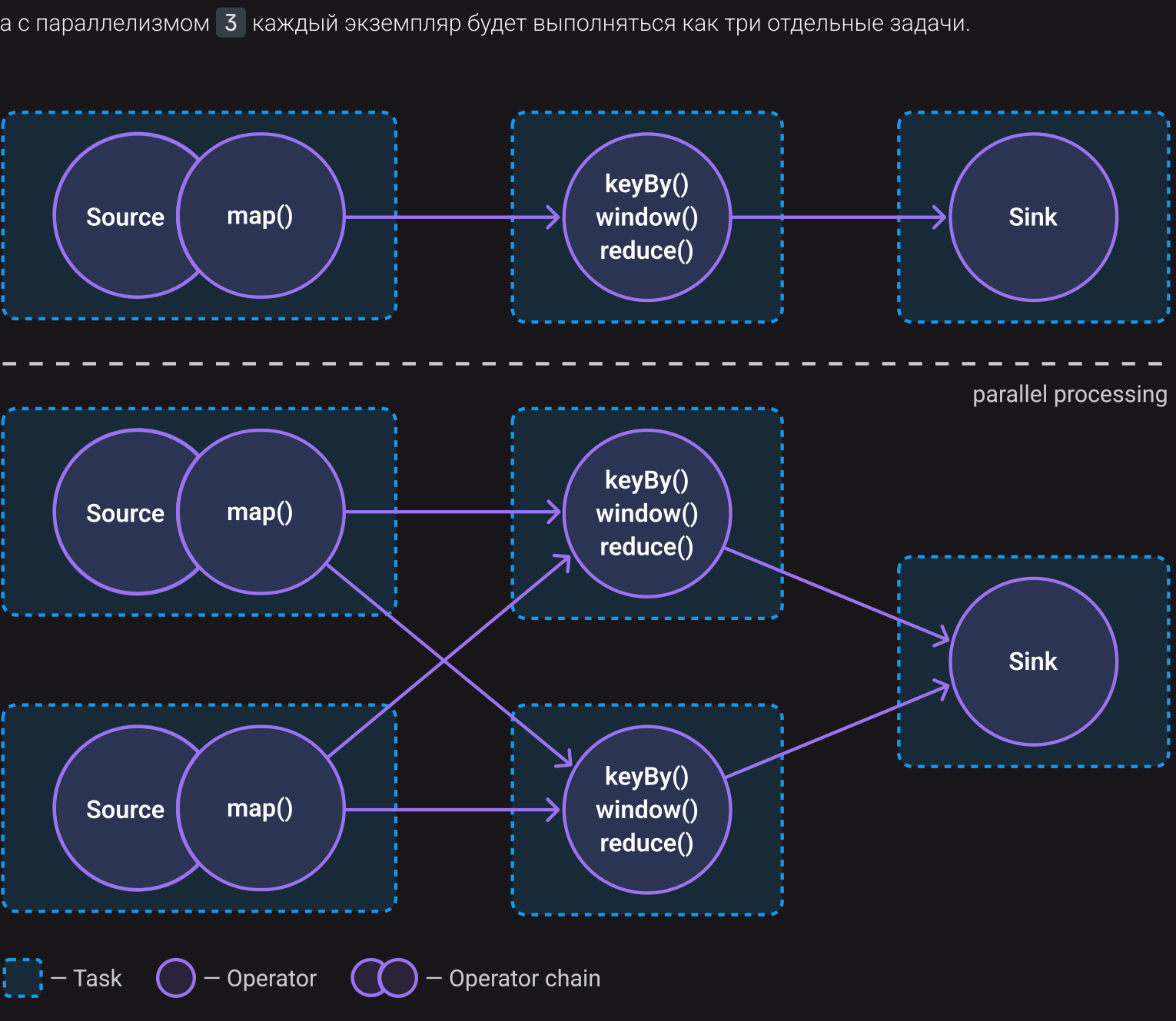
  Основные компоненты диаграммы:

  - Источник (source). Стартовая точка при выполнении задачи Flink. Источник получает данные из внешней системы, например Kafka-брокера, базы данных или сокета, и создает объект DataStream, который далее обрабатывается операторами. Сервис Flink поставляется со встроенными коннекторами для Kafka, JDBC, файловых систем и так далее.
  - Операторы выполняют трансформации над DataStreams.
  - Приемник (sink). Конечная точка обработки данных в приложении Flink. Приемники получают потоки данных в виде объектов DataStream и выводят содержащиеся в них данные в файлы, сокеты, Kafka-топики и другие внешние системы.

- **Задачи и операторы**

  Операторы (operators) получают на входе один или несколько DataStream, применяют к ним трансформации и на выходе отдают новый DataStream. Примеры операторов преобразования: функции `map()`, `flatMap()`, `keyBy()` и прочие.

  Задача (task) — это базовая единица работы Flink. Для каждого экземпляра оператора, работающего в параллельном режиме, требуется одна задача для выполнения работы. Например, у оператора с параллелизмом `3` каждый экземпляр будет выполняться как три отдельные задачи.

  Задачи и операторы

Flink History Server

Сервис Flink содержит компонент Flink History Server, который отображает статистику о завершенных приложениях Flink в веб-интерфейсе.

> **ПРИМЕЧАНИЕ**
>
> Компонент Flink History Server доступен начиная с версии ADH 4.0.0.

Статистика о завершенных задачах Flink генерируется компонентом JobManager и сохраняется в конфигурируемой локации (по умолчанию — */apps/flink/completed-jobs* в HDFS). Flink History Server периодически сканирует данную локацию и отображает информацию о завершенных задачах в веб-интерфейсе. Директории, в которых хранятся архивы, а также интервал сканирования можно настроить с помощью параметров конфигурации сервиса Flink.

History Server REST API

Помимо пользовательского интерфейса, компонент Flink History Server также предоставляет REST API для получения информации о выполненных задачах. REST-эндпойнты не требуют аутентификации, принимают только GET-запросы и доступны по следующему URL:

```
GET http://<history_server_host>:<port>/<endpoint_path> HTTP/1.1
```

где:

- `<history_server_host>` — имя или IP-адрес хоста, на котором установлен компонент Flink History Server.
- `<port>` — номер порта, на котором Flink ожидает REST-запросы. По умолчанию используется порт `8081`. Порт можно изменить с помощью параметра `historyserver.web.port` сервиса Flink в ADCM.

Доступны следующие эндпойнты:

GET [BASE_URL]/config

Возвращает общую информацию о кластере Flink.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.internal:8082/config'
```

Ответ сервера:

```
{
    "refresh-interval": 10000,
    "timezone-name": "Coordinated Universal Time",
    "timezone-offset": 0,
    "flink-version": "1.20.1",
    "flink-revision": "ae77a60 @ 2025-04-28T10:41:27+02:00",
    "features": {
        "web-submit": false,
        "web-cancel": false,
        "web-rescale": false,
        "web-history": true
    }
}
```

GET [BASE_URL]/jobs/overview

Возвращает список выполненных Flink-задач.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.internal:8082/jobs/overview'
```

Ответ сервера:

```
{
    "jobs": [
        {
            "jid": "547cc23f3e85744bffbfcafe7c06471a",
            "name": "Flink Java Job at Sun Jun 15 23:25:44 UTC 2025",
            "start-time": 1750029945589,
            "end-time": 1750029946616,
            "duration": 1027,
            "state": "FINISHED",
            "last-modification": 1750029946616,
            "tasks": {
                "running": 0,
                "canceling": 0,
                "canceled": 0,
                "total": 3,
                "created": 0,
                "scheduled": 0,
                "deploying": 0,
                "reconciling": 0,
                "finished": 3,
                "initializing": 0,
                "failed": 0
            }
        }
    ]
}
```

GET [BASE_URL]/jobs/<job_id>

Возвращает подробную информацию о конкретной задаче Flink, включая план выполнения, вершины логического графа (vertices), подзадачи и так далее.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.internal:8082/jobs/547cc23f3e85744bffbfcafe7c06471a'
```

Ответ сервера:

```
{
    "jid": "547cc23f3e85744bffbfcafe7c06471a",
    "name": "Flink Java Job at Sun Jun 15 23:25:44 UTC 2025",
    "isStoppable": false,
    "state": "FINISHED",
    "job-type": "BATCH",
    "start-time": 1750029945589,
    "end-time": 1750029946616,
    "duration": 1027,
    "maxParallelism": -1,
    "now": 1750029946692,
    "timestamps": {
        "FAILED": 0,
        "SUSPENDED": 0,
        "CANCELLING": 0,
        "INITIALIZING": 1750029945589,
        "FAILING": 0,
        "CANCELED": 0,
        "RUNNING": 1750029946033,
        "FINISHED": 1750029946616,
        "CREATED": 1750029945752,
        "RESTARTING": 0,
        "RECONCILING": 0
    },
    "vertices": [
        ...
    ],
    "status-counts": {
        "INITIALIZING": 0,
        "CANCELING": 0,
        "DEPLOYING": 0,
        "FAILED": 0,
        "SCHEDULED": 0,
        "RUNNING": 0,
        "RECONCILING": 0,
        "FINISHED": 3,
        "CREATED": 0,
        "CANCELED": 0
    },
    "plan": {
        ...
    }
}
```

GET [BASE_URL]/jobs/<job_id>/config

Возвращает конфигурацию определенной задачи Flink.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.internal:8082/jobs/547cc23f3e85744bffbfcafe7c06471a/config'
```

Ответ сервера:

```
{
    "jid": "547cc23f3e85744bffbfcafe7c06471a",
    "name": "Flink Java Job at Sun Jun 15 23:25:44 UTC 2025",
    "execution-config": {
        "execution-mode": "PIPELINED",
        "restart-strategy": "Cluster level default restart strategy",
        "job-parallelism": 1,
        "object-reuse-mode": false,
        "user-config": {}
    }
}
```

GET [BASE_URL]/jobs/<job_id>/vertices/<vertex_id>

Возвращает информацию о конкретной вершине Flink-задачи.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.
internal:8082/jobs/547cc23f3e85744bffbfcafe7c06471a/vertices/33b1d60e4c0d4656355755296d6794e0'
```

Ответ сервера:

```
{
    "id": "33b1d60e4c0d4656355755296d6794e0",
    "name": "CHAIN DataSource (at getDefaultTextLineDataSet(WordCountData.java:70) (org.apache.flink.api.java.io.
CollectionInputFormat)) -> FlatMap (FlatMap at main(WordCount.java:97)) -> Combine (SUM(1), at main(WordCount.java:100)",
    "parallelism": 1,
    "maxParallelism": 1,
    "now": 1750029946787,
    "subtasks": [
        {
            ...
        }
    ],
    "aggregated": {
        ...
    }
}
```

GET [BASE_URL]/jobs/<job_id>/exceptions

Возвращает список исключений, возникших при выполнении задачи.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.internal:8082/jobs/57a9afc1904b14f808c80378ee20ffeb/exceptions'
```

Ответ сервера:

```
{
    "root-exception": null,
    "timestamp": null,
    "all-exceptions": [],
    "truncated": false,
    "exceptionHistory": {
        "entries": [],
        "truncated": false
    }
}
```

GET [BASE_URL]/jobs/<job_id>/accumulators

Возвращает список объектов accumulator, задействованных для выполнения определенной задачи.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.internal:8082/jobs/57a9afc1904b14f808c80378ee20ffeb/accumulators'
```

Ответ сервера:

```
{
    "job-accumulators": [],
    "user-task-accumulators": [
        {
            "name": "85d49c256c7a7e0370ebba1e88e8f199",
            "type": "SerializedListAccumulator",
            "value": "[[B@7b294b0f, [B@2a90f4d2, [B@2a86b8be, ..."
        }
    ],
    "serialized-user-task-accumulators": {
        "85d49c256c7a7e0370ebba1e88e8f199": "rO0ABXNyACVvcmcuYXBhY2hlLmZsaW5rLnV..."
    }
}
```

GET [BASE_URL]/jobs/<job_id>/vertices/<vertex_id>/subtasktimes

Возвращает список подзадач Flink, созданных для выполнения задачи.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.
internal:8082/jobs/547cc23f3e85744bffbfcafe7c06471a/vertices/33b1d60e4c0d4656355755296d6794e0/subtasktimes'
```

Ответ сервера:

```
{
    "id": "33b1d60e4c0d4656355755296d6794e0",
    "name": "CHAIN DataSource (at getDefaultTextLineDataSet(WordCountData.java:70) (org.apache.flink.api.java.io.
CollectionInputFormat)) -> FlatMap (FlatMap at main(WordCount.java:97)) -> Combine (SUM(1), at main(WordCount.java:100)",
    "now": 1750029946723,
    "subtasks": [
        {
            "subtask": 0,
            "host": "ka-adh-3",
            "endpoint": "ka-adh-3.ru-central1.internal:38443",
            "duration": 565,
            "timestamps": {
                "INITIALIZING": 1750029946391,
                "CANCELING": 0,
                "DEPLOYING": 1750029946222,
                "FAILED": 0,
                "SCHEDULED": 1750029946037,
                "RUNNING": 1750029946393,
                "RECONCILING": 0,
                "FINISHED": 1750029946602,
                "CREATED": 1750029945835,
                "CANCELED": 0
            }
        }
    ]
}
```

GET [BASE_URL]/jobs/<job_id>/plan

Возвращает план выполнения (execution plan) задачи Flink.

Пример запроса:

```
$ curl 'http://ka-adh-3.ru-central1.internal:8082/jobs/57a9afc1904b14f808c80378ee20ffeb/plan'
```

Ответ сервера:

```
{
    "plan": {
        "jid": "57a9afc1904b14f808c80378ee20ffeb",
        "name": "Flink Java Job at Sun Jun 15 23:49:15 UTC 2025",
        "type": "BATCH",
        "nodes": [
            {
                "id": "5b6d423820643b538f78784054364b75",
                "parallelism": 1,
                "operator": "Data Sink",
                "operator_strategy": "(none)",
                "description": "DataSink (collect())",
                "inputs": [
                    {
                        "num": 0,
                        "id": "fa8699328ec101fd063912c9734556b7",
                        "ship_strategy": "Forward",
                        "exchange": "pipelined"
                    }
                ],
                "optimizer_properties": {
                    ...
                }
            },
            {
                "id": "fa8699328ec101fd063912c9734556b7",
                "parallelism": 1,
                "operator": "GroupReduce",
                "operator_strategy": "Sorted Group Reduce",
                "description": "Reduce (SUM(1), at main(WordCount.java:100)",
                "inputs": [
                    {
                        "num": 0,
                        "id": "9d782237f8c637c037444a774f60e6e7",
                        "ship_strategy": "Hash Partition on [0]",
                        "local_strategy": "Sort (combining) on [0:ASC]",
                        "exchange": "pipelined"
                    }
                ],
                "optimizer_properties": {
                    ...
                }
            },
            {
                "id": "9d782237f8c637c037444a774f60e6e7",
                "parallelism": 1,
                "operator": "Data Source -> FlatMap -> GroupCombine",
                "operator_strategy": "(none)<br/> -&gt; FlatMap<br/> -&gt; Sorted Combine",
                "description": "DataSource (at getDefaultTextLineDataSet(WordCountData.java:70) (org.apache.flink.api.java.io.
CollectionInputFormat))<br/> -&gt; FlatMap (FlatMap at main(WordCount.java:97))<br/> -&gt; Combine (SUM(1), at main(WordCount.
java:100)",
                "optimizer_properties": {
                    ...
                }
            }
        ]
    }
}
```

Архитектура HBase

Константин Алпашкин

Содержание

Назад к содержанию

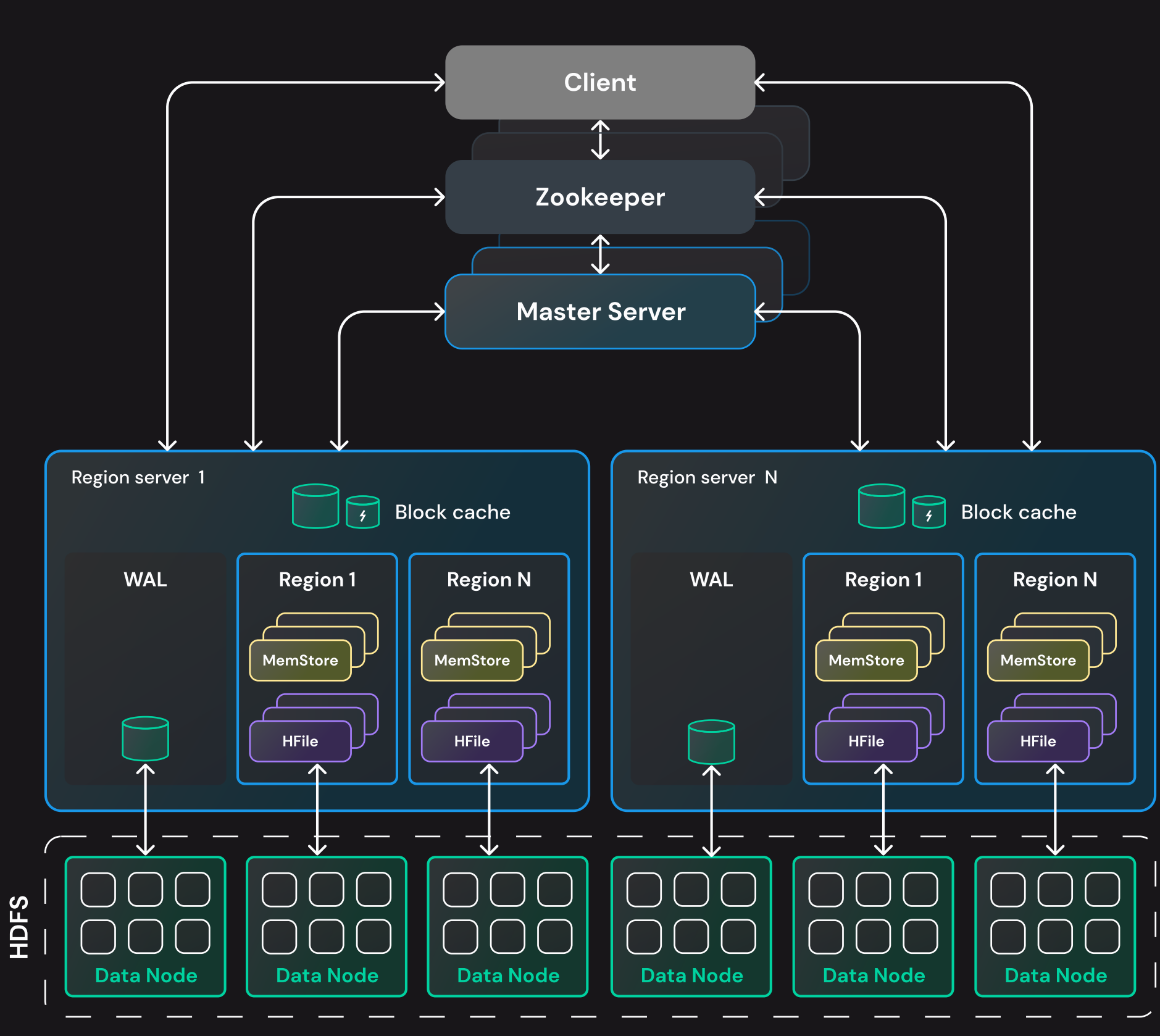
- Компоненты
  - Region Server
  - Master server
  - ZooKeeper
- Операции с данными
  - Препроцессинг
  - Запись
  - Чтение
- Повышение производительности
  - Region split
  - HFile compaction
- Восстановление данных

Компоненты

В основе архитектуры HBase лежат следующие компоненты:

- **Region Server**. Обслуживает один или несколько регионов (Regions). Регион — это диапазон записей, которые хранятся вместе. Каждый регион обслуживается только одним Region-сервером. Region-сервер также называют HRegionServer. Region-сервер содержит несколько компонентов, некоторые из них работают поверх HDFS, используя последний как хранилище данных.
- **Master server**. Основной сервер, отвечающий за управление HBase-кластером. По своим функциям Master-сервер схож с NameNode в HDFS. Master-сервер управляет распределением регионов между Region-серверами, осуществляет регистрацию регионов, и так далее. Master-сервер также называется HMaster. Вы можете развернуть несколько Master-серверов в кластере: один активный и один или более резервных (standby).
- ZooKeeper. Специальный сервис для управления конфигурациями и синхронизации сервисов. Используется для координации действий между сервисами HBase.

Высокоуровневая архитектура HBase представлена на следующей схеме.

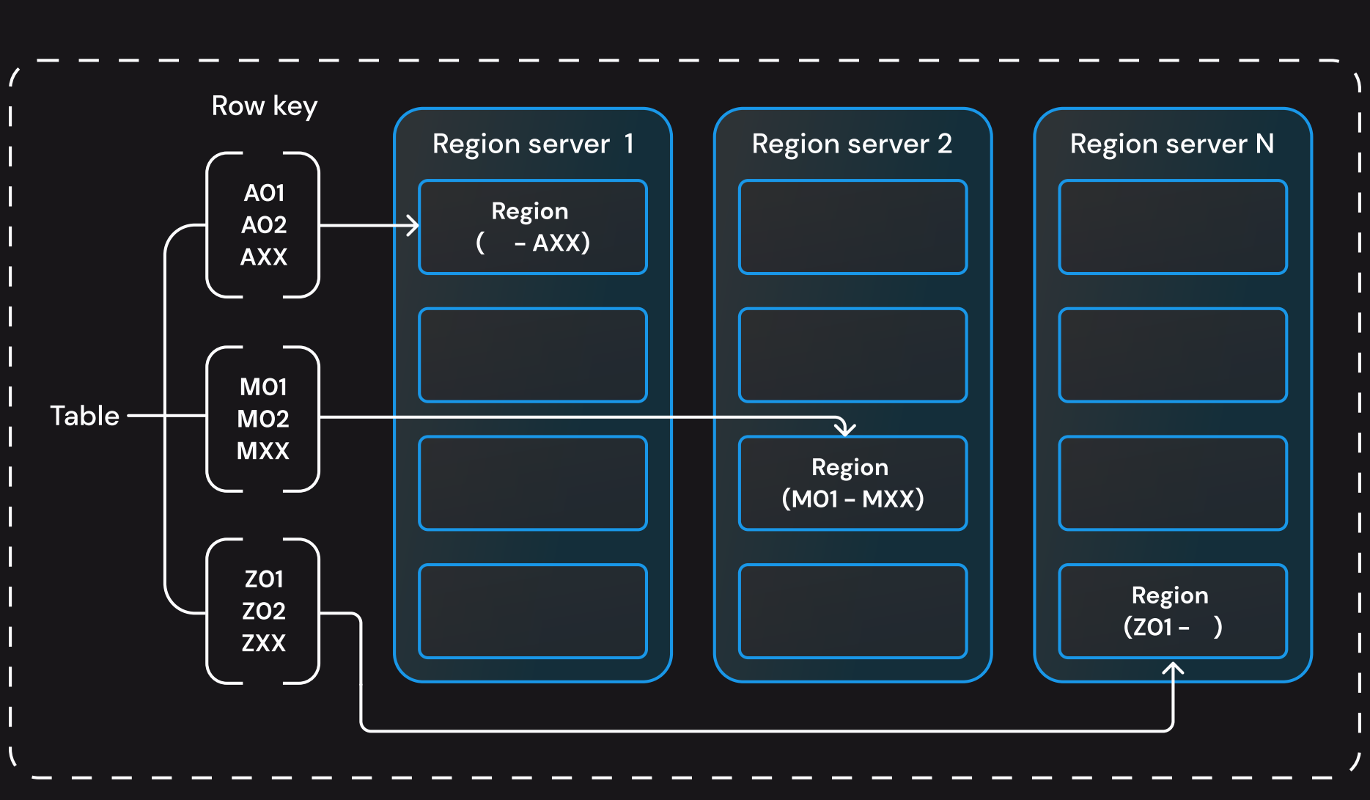


Архитектура HBase

Region Server

Region-сервер обслуживает несколько регионов (Regions), которые работают поверх HDFS. Каждый регион содержит строки, где ключи из одного диапазона принадлежат данному региону. Строки отсортированы в алфавитном порядке по ключам. Каждый регион имеет размер по умолчанию 256 МБ, а каждый Region-сервер может содержать примерно 1000 регионов.

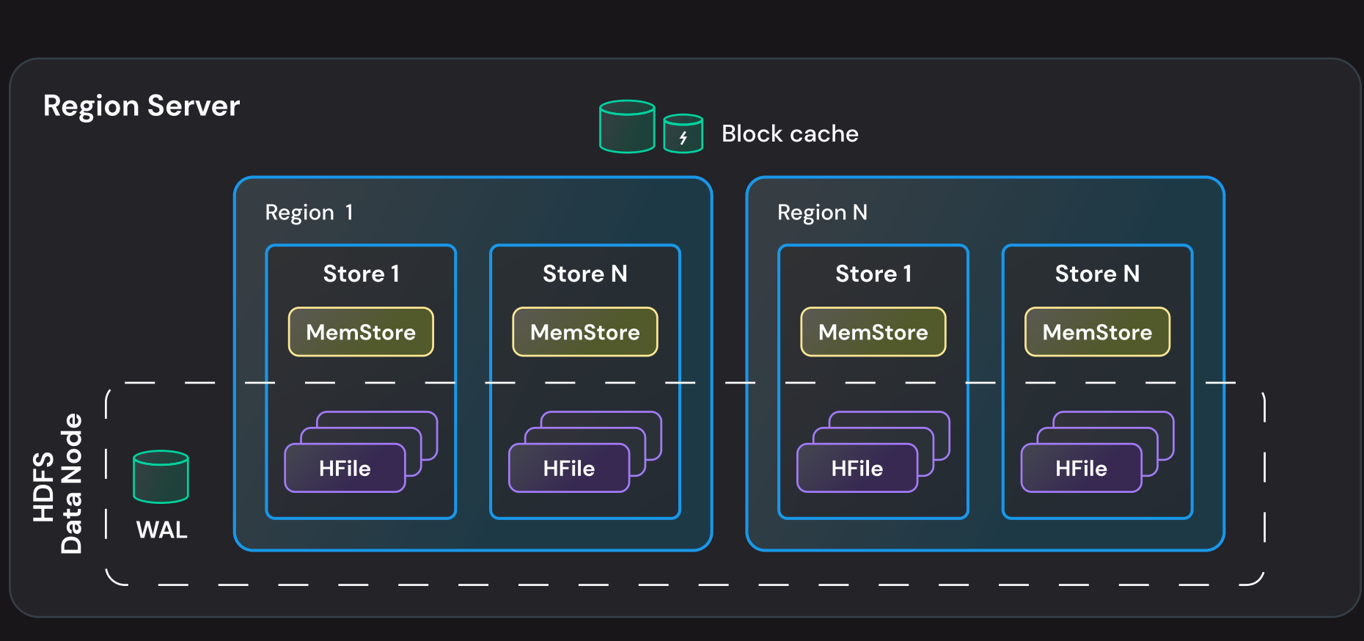
> **ПРИМЕЧАНИЕ**
>
> Каждый регион определяется диапазоном ключей строк. Первый регион таблицы имеет диапазон, который начинается с пустого значения. Последний регион таблицы имеет диапазон, заканчивающийся пустым значением.



Пример распределения ключей между регионами

Region-сервер управляет несколькими регионами и выполняет операции чтения/записи. Каждый Region-сервер включает следующие компоненты:

- **HFiles** — основное хранилище данных в HBase. Данные физически хранятся на HDFS в специальном формате HFile и отсортированы по RowKey. Одной паре <Region, Column family> соответствует как минимум один HFIle.
- **MemStore** — буфер на запись в памяти. Так как данные хранятся в HFile в отсортированном порядке, обновление HFile для каждой операции записи довольно дорого. Вместо этого данные при записи попадают в специальную область памяти MemStore, где накапливаются некоторое время. При наполнении MemStore до критического значения данные записываются в новый HFile. Данные сортируются в лексикографическом порядке перед сохранением в HFile.
  Один MemStore соответствует одной группе колонок (column family). По этой причине один регион может содержать несколько MemStore-буферов, так как каждый регион может содержать несколько групп колонок. Совокупность MemStore-буферов и всех HFiles, которые содержат данные одной группы колонок, называют хранилищем (Store).
- **BlockCache** — кеш на чтение, который хранит часто читаемые данные в памяти, тем самым позволяя ускорить операции чтения. Данные, которые не используются долгое время, удаляются из BlockCache.
- **Write Ahead Log (WAL)** — специальный лог-файл (также именуемый HLog), хранящийся в HDFS и прикрепленный к каждому Region-серверу. Данный файл используется, чтобы предотвратить возможные потери данных при сбоях MemStore. Все входящие операции записываются в данный файл перед выполнением. WAL-файл содержит данные, которые еще не были сохранены в перманентное хранилище, что позволяет восстановить данные после сбоя.



Компоненты Region Server

Master server

Master server занимается мониторингом всех Region-серверов и следит за всеми изменениями метаданных в HBase-кластере. Его основные функции описаны ниже:

- Управляет распределением регионов по Region-серверам и переопределяет их при восстановлении данных или во время процессов балансирования нагрузки.
- Координирует работу Region-серверов аналогично тому, как NameNode управляет DataNode в HDFS.
- Выполняет DDL-операции, позволяя создавать, удалять и обновлять таблицы.
- Выполняет мониторинг всех Region-серверов в кластере, а также обеспечивает восстановление данных после сбоя с помощью ZooKeeper.

ZooKeeper

ZooKeeper координирует взаимодействие компонентов в HBase-кластере и выполняет следующие функции:

- Проверяет доступность Region-серверов, получая от них heartbeat-сигналы. Если Region-сервер не посылает heartbeat-сигнал в ожидаемый интервал времени, Master-сервер получает уведомление и запускает автоматический процесс восстановления данных.
- Обеспечивает резервные Master-сервера информацией о состоянии активного Master-сервера. Если активный Master-сервер не посылает heartbeat-сигнал для ZooKeeper, один из резервных Master-серверов получает соответствующее уведомление и становится активным.
- Хранит расположение таблицы-каталога hbase:meta (прежнее название — `.META.`), где хранится список всех регионов в системе. Эта таблица хранит информацию о регионах в виде пар ключ/значение, где ключ — это комбинация полей <имя таблицы, начальный ключ региона, ID региона>, а значение — полная информация о данном регионе, включая путь к Region-серверу, где содержится данный регион.

Операции с данными

Препроцессинг

Когда клиент отправляет запрос на запись/чтение к HBase, клиенту необходимо получить адрес Region-сервера, который содержит регион с необходимыми данными. Все операции с данными в HBase начинаются со следующих шагов.

1. Клиент получает расположение таблицы hbase:meta от ZooKeeper и сохраняет ее в собственный кеш.
2. Из таблицы `hbase:meta` клиент запрашивает локацию Region-сервера, который содержит блок данных с определенными ключами строк. Эту информацию клиент также сохраняет в кеш.
3. Клиент запрашивает необходимую информацию из полученного Region-сервера.

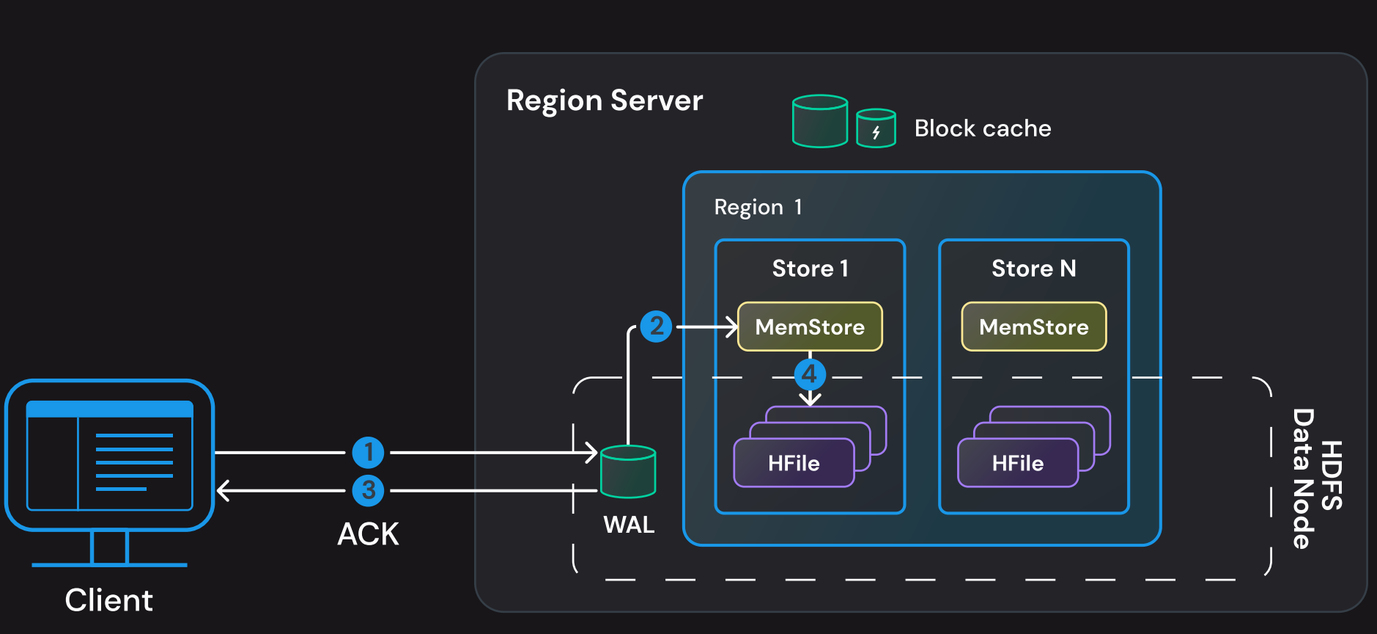
Для дальнейших обращений, клиент использует собственный кеш для получения местонахождения таблицы `hbase:meta` и пути к Region-серверу, используя ключи строк, считанные ранее. Клиент не обращается к таблице `hbase:meta` до тех пор, пока в таблице не обнаружены недостающие данные из-за перемещения или удаления региона. В этом случае клиент запрашивает информацию из таблицы `hbase:meta` и обновляет кеш.

Запись

После определения необходимого Region-сервера, типичные шаги при операциях записи в HBase выглядят следующим образом:

1. HBase записывает данные в WAL (Write Ahead Log), функция которого — обеспечение отказоустойчивости.
2. Данные копируются в MemStore, который служит оперативной памятью для HDFS DataNode. Выбор MemStore зависит от группы колонок, указанной в запросе.
3. После помещения данных в WAL и MemStore, клиент получает подтверждение (ACK) о выполнении задачи.
4. Далее при заполнении MemStore, HBase записывает все данные в HFile в сортированном виде.

> **ПРИМЕЧАНИЕ**
>
> Клиентские приложения не взаимодействуют напрямую с HFile, которые хранятся в HDFS.



Типичная операция записи

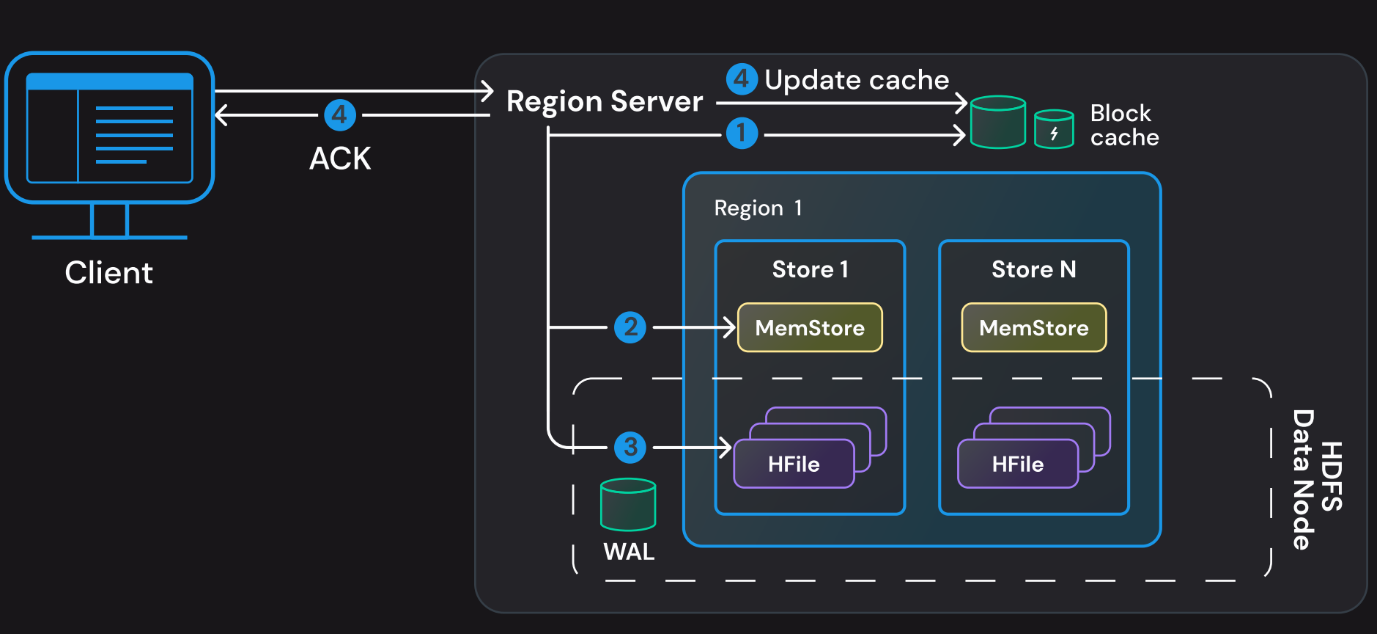
Чтение

После определения необходимого Region-сервера, типичные шаги при операциях чтения из HBase выглядят следующим образом:

1. Выполняется поиск необходимых данных в BlockCache — пространство, где хранятся все недавно считанные пары ключ/значение.
2. Если данные не найдены в BlockCache, поиск выполняется в MemStore, который хранит недавно записанные файлы, но которые еще не были записаны в HFiles.
3. Если данные не найдены в MemStore, поиск выполняется в HFiles, которые хранятся в HDFS.

> **ПРИМЕЧАНИЕ**
>
> Так как каждый HFile содержит MemStore-снепшот, созданный в момент записи в HFile, данные всей запрашиваемой строки могут храниться в нескольких HFile. Таким образом, чтобы собрать данные всей строки, HBase считывает все HFile, которые могут содержать данные из запрашиваемой строки. Использование Bloom-фильтров помогает упростить этот процесс.

4. После того, как поиск выполнен (вне зависимости, на каком из этапов был найден результат), данные записываются в BlockCache и возвращаются клиенту вместе с подтверждением (ACK) о выполнении задачи.



Типичная операция чтения

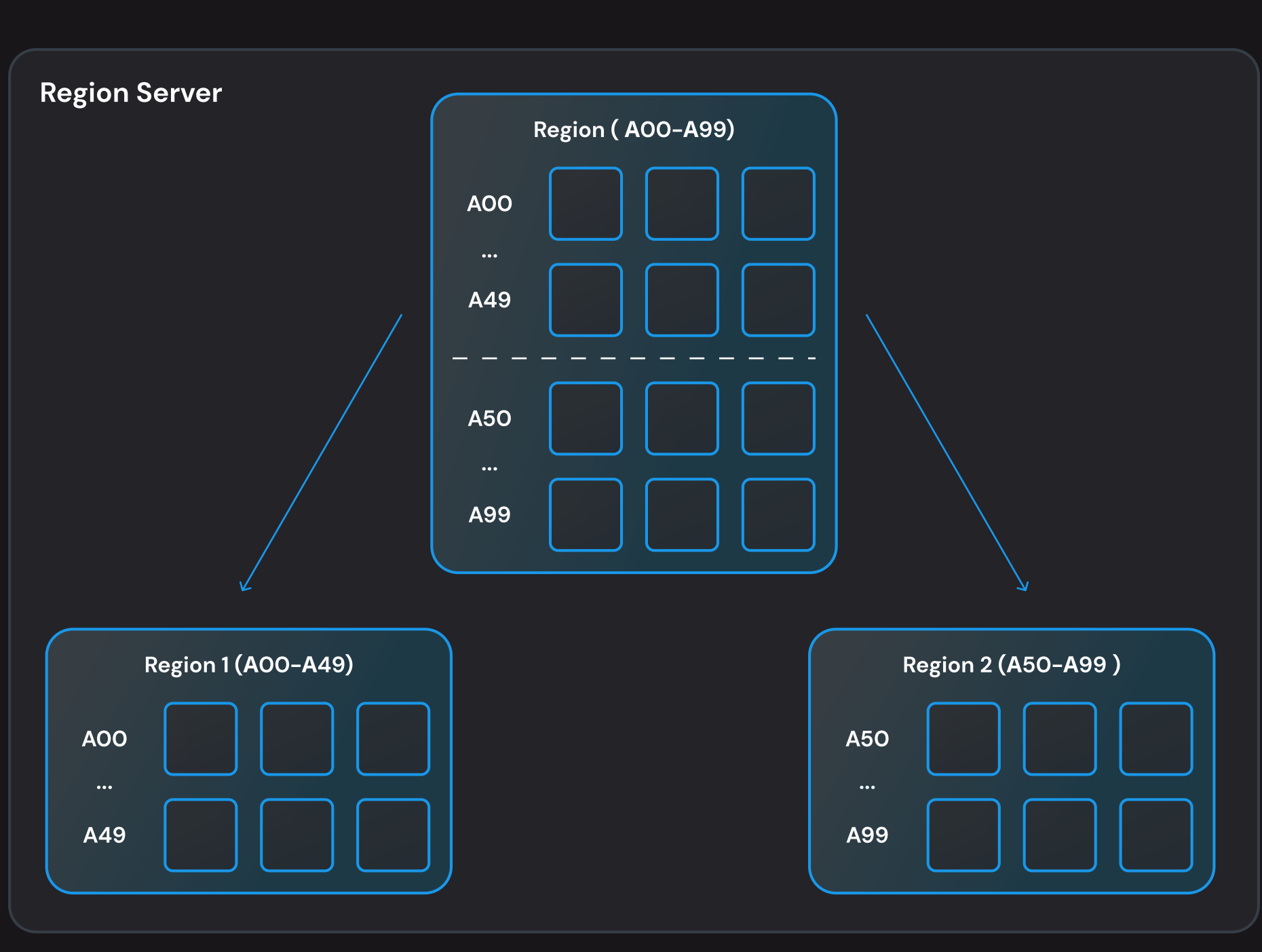
Повышение производительности

Region split

Когда данные в регионе вырастают до некоторого критического размера, HBase выполняет разделения региона (Region split) — специальную процедуру, которая разделяет исходный регион на два дочерних. Затем об изменениях сообщается в Master-сервер.

Новые регионы управляются тем же Region-сервером до тех пор, пока Master-сервер не присвоит им новый Region-сервер (для балансировки нагрузки).

> **РЕКОМЕНДАЦИЯ**
>
> Чтобы избежать частых разделений регионов, можно установить границы регионов и увеличить их максимальный размер.

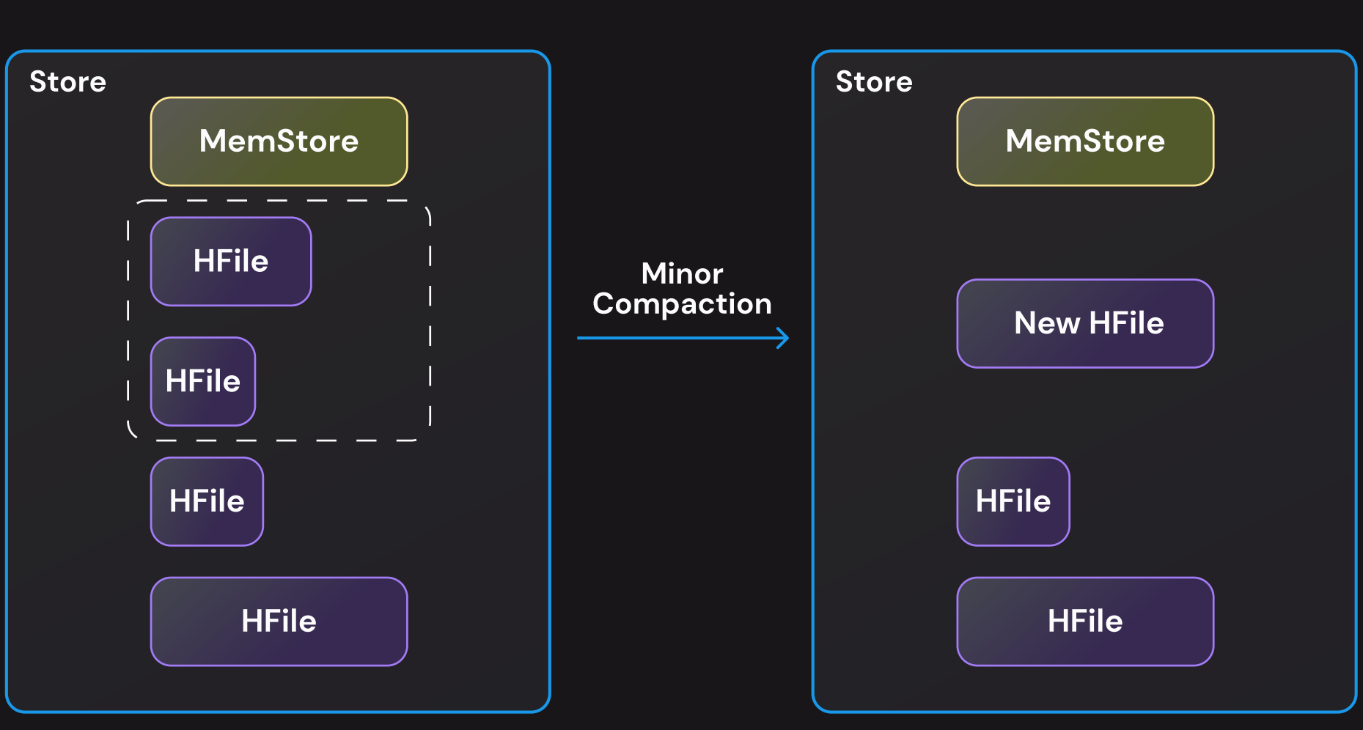


Разделение регионов

HFile compaction

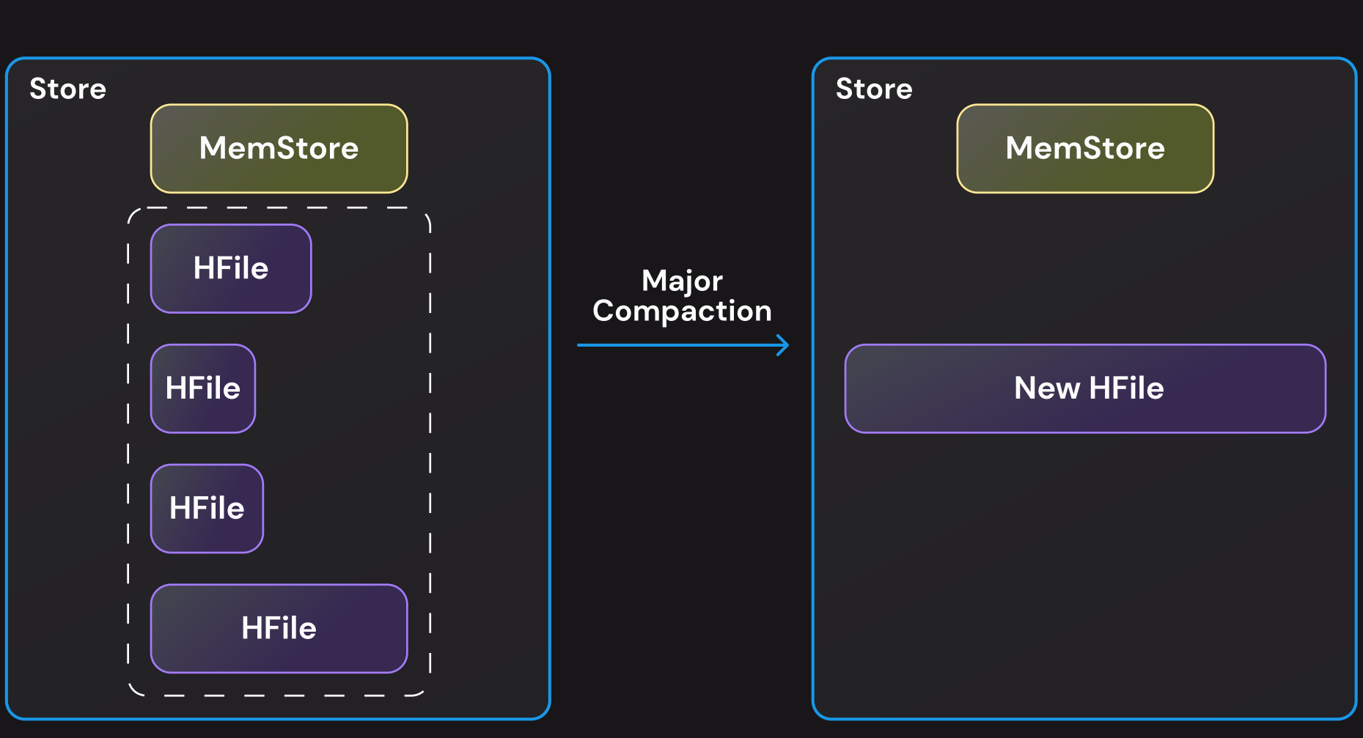
Поскольку данные в одном регионе могут храниться в нескольких HFile, HBase периодически объединяет их вместе, чтобы уменьшить количество обращений к диску, необходимое для одной операции чтения, и ускорить работу. Данная операция называется **сжатием (compaction)**. Существует 2 типа сжатия:

- **Частичное сжатие (Minor compaction)** — HBase выбирает несколько HFile и записывает их содержимое в один большой HFile. Это помогает оптимизировать использование дискового пространства. Данный процесс стартует автоматически и выполняется в фоновом режиме с низким приоритетом по сравнению с другими HBase-процессами. Для частичного сжатия HFile выбираются эвристическим методом.



Частичное сжатие

- **Полное сжатие (Major compaction)** — HBase переписывает все HFile определенного Region-хранилища в один большой HFile. При этом также выполняется физическое удаление устаревших данных и данных, отмеченным флагом `tombstone`, что положительно сказывается на производительности при чтении.
  Полное сжатие запускается вручную или при срабатывании определенных условий (например, по таймеру). Этот тип имеет высокий приоритет и может значительно замедлить работу кластера, так как диски ввода-вывода и сетевой трафик могут быть перегружены. Таким образом, этот тип сжатия следует планировать во время низких пиковых нагрузок.



Полное сжатие

Восстановление данных

HBase поддерживает автоматическое восстановление данных после различных сбоев. Типичный процесс восстановления включает следующие шаги:

1. Если какой-либо из Region-серверов выходит из строя, ZooKeeper уведомляет об этом Master-сервер.
2. Master-сервер распределяет все регионы вышедшего из строя Region-сервера между активными Region-серверами. При этом Master-сервер также распределяет WAL вышедшего из строя Region-сервера, который хранится в HDFS. Это необходимо для восстановления данных с вышедших из строя MemStore (которые более недоступны).
3. Каждый Region-сервер обрабатывает WAL-файл, чтобы восстановить MemStore для каждой группы колонок из вышедшего из строя Region-сервера. Так как все данные записываются в WAL-файл в хронологическом порядке, обработка записей позволяет воссоздать все необходимые изменения, сохраненные в MemStores.
4. Регионы продолжают работать с активными Region-серверами.

Модель данных HBase

Константин Алпашкин

Содержание

Назад к содержанию

Основные объекты

Принцип работы

Пример

HBase — это колоночно-ориентированная база данных. И хотя HBase хранит данные в таблицах со строками и столбцами (как это делают реляционные базы данных), не стоит рассматривать эти таблицы как наборы строк с фиксированной схемой. Вместо этого таблицы HBase стоит рассматривать как многомерные структуры данных типа ключ/значение.

Основные объекты

Для начала стоит рассмотреть терминологию HBase:

- **Таблица (Table)**. Набор из нескольких строк, которые могут иметь один или несколько столбцов, принадлежащих фиксированным семействам столбцов. Каждая схема таблицы определяет только семейство столбцов.
- **Строка (Row)**. Состоит из ключа строки и столбцов со значениями, связанными с ними. Строки сортируются в алфавитном порядке по ключу строки и хранятся в таком же виде.
- **Ключ строки (Row key)**. Уникальный идентификатор строки, используемый в качестве первичного ключа. Этот ключ используется для доступа к таблицам HBase.

РЕКОМЕНДАЦИЯ

Тщательно продумывайте дизайн ваших ключей. Поскольку строки сортируются по ключу, основная цель состоит в том, чтобы хранить связанные строки рядом друг с другом. Однако следует уделять должное внимание равномерности распределения — с целью избежать ситуации, когда одна часть ключей строк используется намного чаще, чем другие.

- **Столбец (Column)**. Состоит из семейства столбцов и квалификатора столбца, разделенных двоеточием, например: `cf:column1`, `cf:column2` и так далее.
- **Семейство столбцов (Column family)**. Логическое и физическое объединение столбцов и их значений — обычно выполняется для улучшения производительности. Семейства столбцов фиксируются при создании таблицы и не могут быть изменены "на лету". Тем не менее, каждая строка таблицы может ничего не хранить в семействе столбцов.
  Семейство столбцов может иметь набор свойств хранения, таких как: следует ли кешировать значения в памяти, методы сжатия данных или кодирования строк, время жизни (TTL) и максимальное количество хранимых версий для значений и так далее.

ВНИМАНИЕ

Не рекомендуется иметь более трех семейств столбцов для каждой таблицы.

- **Квалификатор столбца (Column qualifier)**. Идентификатор столбца, необходимый для предоставления индекса для части данных в семействе столбцов. Хотя семейства столбцов фиксированы, квалификаторы можно менять "на лету", и они могут сильно различаться между строками одной и той же таблицы.
- **Ячейка (Cell)**. Комбинация строки, семейства столбцов и квалификатора столбца. Содержит значение данных и метку времени, представляющую текущую версию значения.
- **Метка времени (Timestamp)**. Идентификатор версии значения ячейки. По умолчанию представляет время на Region-сервере, когда значение ячейки было записано в HBase. Вы также можете указать другое значение метки времени.
- **Значение (Value)**. Часть данных, хранящихся в HBase в определенной комбинации: <Таблица, Ключ строки, Семейство столбцов, Квалификатор столбца, Метка времени>.

Модель данных HBase можно представить в виде следующей структуры данных типа ключ/значение: <Table, Row key, Column family, Column qualifier, Timestamp> → Value

Принцип работы

На логическом уровне данные в HBase организованы в таблицы, индексированные по первичному ключу строки. Вы можете хранить неограниченное количество столбцов для каждого ключа строки. Столбцы организованы в группы, называемые семействами столбцов. Как правило, каждое семейство столбцов объединяет столбцы с одинаковым шаблоном использования и хранения.

Столбцы используются для хранения фрагментов данных, называемых значениями. Каждый столбец определяется именем семейства столбцов и собственным уникальным именем, называемым квалификатором столбца. Столбец может хранить несколько версий одного значения. Разные версии значений имеют разные метки времени.

Допустим, имеется таблица с двумя семействами столбцов и двумя строками, идентифицированными ключами строк. Для каждой комбинации ключа строки и семейства столбцов вы можете определить разные квалификаторы столбцов. Вы также можете добавлять новые квалификаторы столбцов "на лету".

Примерный логический вид такой таблицы показан ниже (где `ts` — это сокращение "timestamp").

Логическое представление

Ключ строки	Семейство столбцов 1		Семейство столбцов 2	
	Квалификатор столбца 1	Квалификатор столбца 2	Квалификатор столбца 3	Квалификатор столбца 4
Ключ строки 1	ts1:value1	ts3:value3 ts2:value2	ts6:value6	—
Ключ строки 2	ts5:value5 ts4:value4	—	—	ts8:value8 ts7:value7

В таблице, показанной выше, можно увидеть разреженные данные, так как заполнение столбцов для каждого ключа строки необязательно. Вам не нужно заполнять каждое семейство столбцов для ваших строк. Отсутствие столбца не вызывает нагрузки при сохранении пустых значений, так как на физическом уровне данные хранятся иначе, чем на логическом.

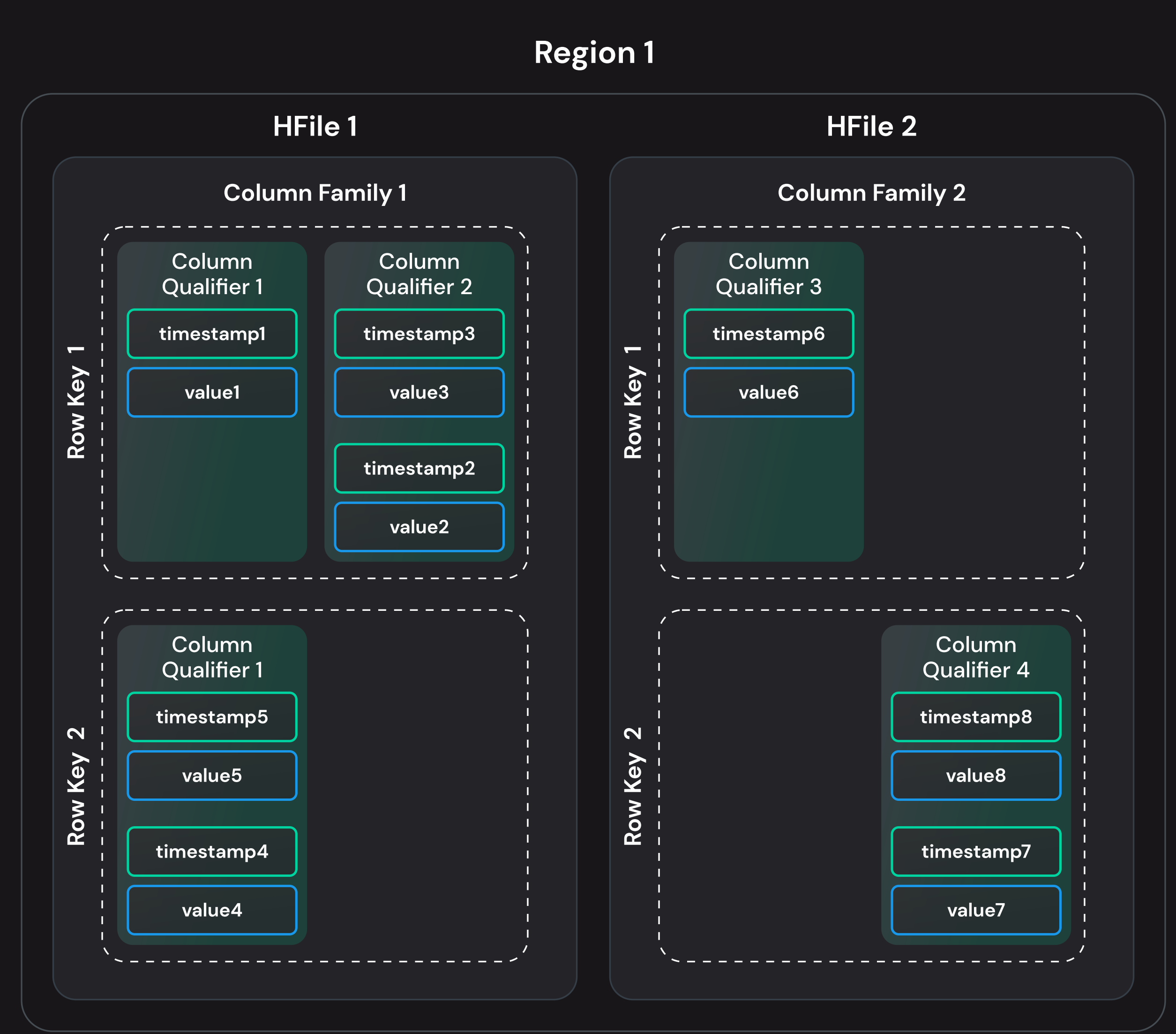
На физическом уровне все значения данных в HBase хранятся в отсортированном виде по ключу строки. Блоки ближайших ключей строк образуют регионы, подробную информацию о которых вы можете найти в статье Архитектура HBase. Данные, соответствующие разным семействам столбцов, хранятся отдельно — в разных HFile. Каждое семейство столбцов использует как минимум один HFile. Это позволяет при необходимости читать данные только из нужного семейства столбцов. Столбцы, принадлежащие одному семейству и одному и тому же ключу строки, также сохраняются в виде списка, отсортированного в алфавитном порядке.

Любой столбец может отсутствовать или присутствовать для каждого ключа строки, так как пустые значения не сохраняются в базе данных.

При удалении определенного значения столбца на физическом уровне оно не удаляется в этот же момент. Вместо этого оно помечается специальным флагом `tombstone`. Физическое удаление данных происходит позже, при выполнении операции Major compaction.

Помимо ручного удаления старых версий, вы можете настроить автоматическое удаление. На уровне семейства столбцов можно определить такие параметры, как время жизни (TTL) и максимальное количество хранимых версий. Если разница между отметкой времени для конкретной версии и текущим временем больше, чем TTL, такая запись помечается для удаления. Если количество версий для определенного столбца превышает максимальное значение хранимых версий, запись также помечается на удаление.

Условное физическое представление таблицы показано ниже. На рисунке видно, что все данные хранятся в одном регионе, а разные семейства столбцов принадлежат разным HFile внутри этого региона.



Физическое представление

Пример

Данный раздел описывает модель данных HBase на конкретном примере.

Допустим, необходимо хранить текстовые статьи. Каждая статья имеет базовые атрибуты, которые меняются очень редко (`author`, `header` и другие), и теги, описывающие содержание статьи и помогающие выполнять текстовый поиск по всем статьям (`arch`, `concepts`, `tutorials` и другие). Теги изменяются чаще.

РЕКОМЕНДАЦИЯ

Поскольку данные, соответствующие разным семействам столбцов, хранятся отдельно, необходимо распределить их по частоте использования.

Для данного примера можно создать таблицу `articles` с двумя семействами столбцов:

- `basic` — для хранения данных, которые изменяются редко;
- `tags` — для хранения тегов.

Следующий JSON иллюстрирует возможную модель данных для данного примера.

Не стоит рассматривать код ниже как точную иллюстрацию того, как данные хранятся на физическом уровне. Это всего лишь макет, используемый для понимания основных объектов модели данных HBase. JSON больше подходит для описания отображений, используемых в HBase, чем обычная двумерная таблица.

```
{
    "article1": { ①
        "basic": { ②
            "author": { ③
                1637054560096: "Test author" ④
            },
            "header": {
                1637056832082: "Test article. Version 3",
                1637055836875: "Test article. Version 2",
                1637054560118: "Test article"
            }
        },
        "tags": {
            "arch": {
                1637054560141: true
            },
            "concepts": {
                1637054560160: true
            },
            "tutorials": {
                1637054564066: true
            }
        }
    },
    "article2": {
        "basic": {
            "author": {
                1637054576501: "Test author2"
            },
            "header": {
                1637054576516: "Test article2"
            }
        },
        "tags": {
            "ref": {
                1637054577512: true
            }
        }
    }
}
```

① Верхний уровень пар ключ/значение представляет строки таблицы: `article1` и `article2` — это ключи строк, уникальные для каждой строки.

② Второй уровень описывает семейства столбцов, которые одинаковы для каждой строки таблицы: `basic` и `tags`.

③ Третий уровень описывает квалификаторы столбцов: `author` и `header` — принадлежат семейству столбцов `basic`; `arch`, `concepts`, `tutorials` и `ref` — принадлежат семейству столбцов `tags`. Квалификаторы столбцов не фиксированы, их можно добавлять "на лету". Их набор может отличаться для разных строк.

④ Нижний уровень описывает ячейки — пары временных меток и значений. Благодаря временным меткам поддерживается управление версиями значений.

Таким образом, чтобы получить определенное значение данных, хранимое в HBase, необходимо знать следующий путь: *Table > Row Key > Column Family > Column Qualifier > Timestamp*. Без определения временных меток вы получаете самые последние данные. Например, если вы попытаетесь получить значение строки с ключом `article1` и столбцом `basic:header`, вы получите самое последнее значение `Test article. Version 3`. Но если вы дополнительно укажете временную метку (например, `1637054560118`), вы получите значение `Test article`.

```
hbase(main):005:0> get 'articles', 'article1', {COLUMN => 'basic:header', TIMESTAMP => 1637054560118}
COLUMN                CELL
 basic:header         timestamp=1637054560118, value=Test article
1 row(s)
Took 0.0202 seconds
hbase(main):006:0> get 'articles', 'article1', 'basic:header'
COLUMN                CELL
 basic:header         timestamp=1637056832082, value=Test article. Version 3
1 row(s)
Took 0.0054 seconds
```

Основные команды, применимые к модели данных, показанной в этом примере, описаны в статье Начало работы с HBase shell.

Архитектура HDFS

Альберт Багдасарян

Содержание

Назад к содержанию

Компоненты

Архитектура

Операции с файлами

Репликация

- Принципы хранения
- Принципы работы

HDFS (Hadoop Distributed File System) — это распределенная файловая система, обеспечивающая высокую отказоустойчивость на оборудовании с низкой стоимостью. Она обеспечивает высокую скорость доступа к данным приложений и в наибольшей степени пригодна для больших наборов данных.

Все файлы и каталоги в HDFS представлены в виде индексных дескрипторов (inodes), содержащих различные атрибуты, такие как привилегии, время последнего изменения, ограничение на дисковое пространство, ограничение на пространство namespace (например, количество inodes) и время последнего доступа.

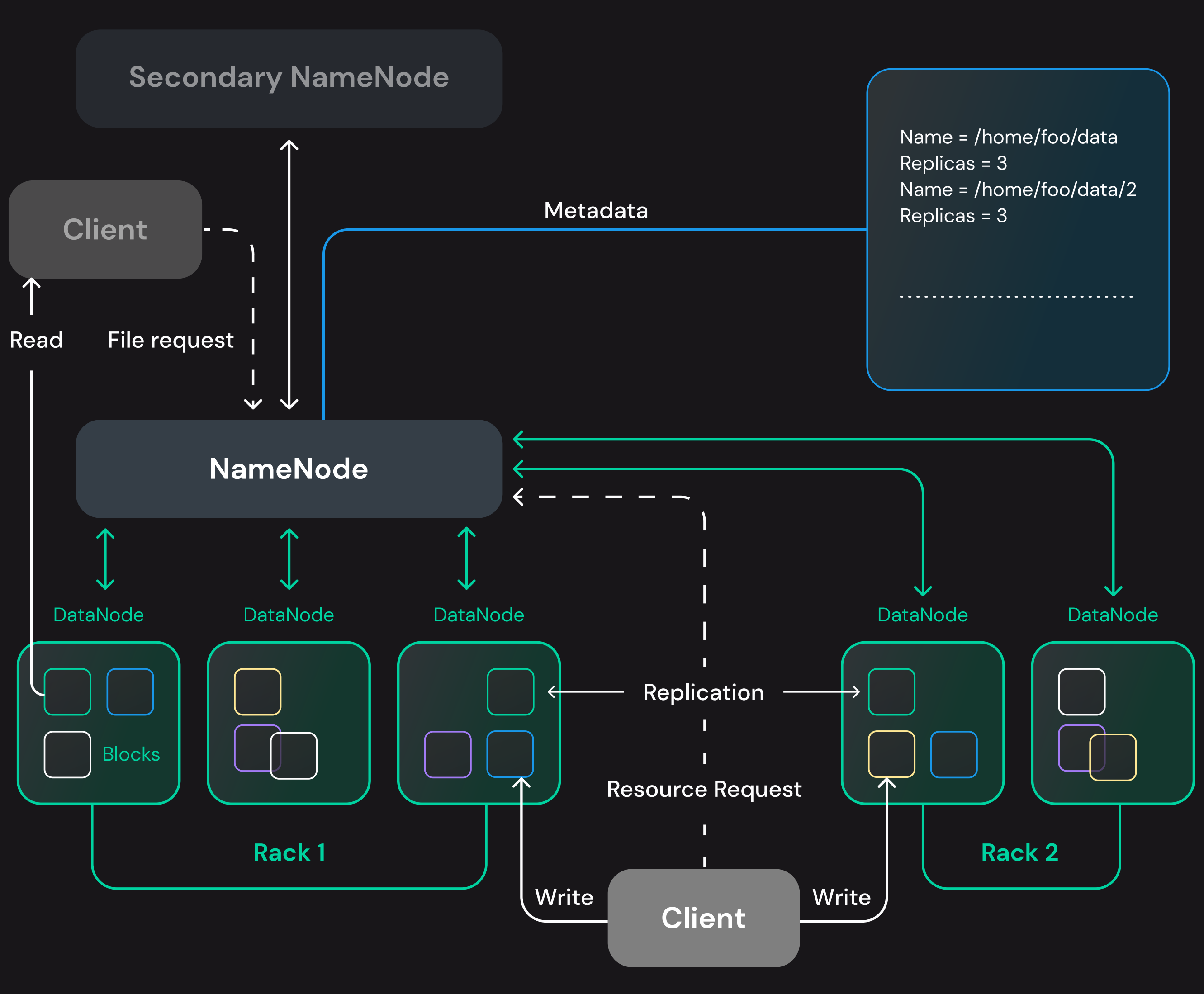
Компоненты

Функции HDFS обеспечивают следующие компоненты:

- **NameNode** является центральной нодой управления системой. Она обслуживает пространство имен (namespace) в HDFS, хранящееся в ее оперативной памяти и содержащее полную информацию о каталогах и файлах. NameNode также управляет распределением блоков файлов среди DataNodes.
- **DataNodes** — это большинство нод кластера, подчиняющихся NameNode и используемых для хранения данных и их обработки. Они исполняют запросы от клиентов на чтение и запись блоков данных.
- **Secondary NameNode** — это вспомогательный сервер, который периодически создает полный образ текущего содержимого пространства имен, то есть его контрольные точки (checkpoint). Он делает это путем обработки журнала транзакций с его последующей очисткой. Вспомогательный сервер обеспечивает доступ к checkpoint для NameNode, чтобы последняя могла в любое время обновить содержимое namespace в своей оперативной памяти. В случае сбоя на NameNode ее можно запустить заново, используя последний checkpoint и журнал транзакций.

Архитектура

NameNode является основной нодой, ответственной за HDFS namespace и управление доступом клиентов к файлам. DataNodes управляют хранилищами данных.



Компоненты HDFS

HDFS предоставляет пользователям пространство имен файловой системы и обеспечивает хранение данных в виде файлов. Каждый файл HDFS состоит из одного или нескольких блоков, распределенных среди DataNodes. NameNode выполняет операции с файлами в пределах пространства имен, включая их открытие и закрытие, а также переименование файлов и каталогов. NameNode также управляет расположением файловых блоков в DataNodes.

DataNodes выполняют запросы клиентов HDFS на чтение и запись блоков данных. Они также обрабатывают запросы от NameNode на создание, удаление и репликацию блоков.

HDFS поддерживает традиционную организацию файловой системы. Приложение или пользователь может создавать каталоги и затем сохранять в них файлы. Иерархия пространства имен аналогична многим другим файловым системам, то есть можно создавать, удалять, переименовывать или перемещать файлы.

NameNode сохраняет все изменения в пространстве имен или свойствах файлов. Приложение может потребовать создание определенного количества копий блоков файла в HDFS. Это количество называется фактором репликации (replication factor) данного файла.

Операции с файлами

В отличие от других файловых систем, HDFS не позволяет производить запись в файл. Файлы HDFS записываются только один раз, и только один процесс может производить эту запись в каждый момент времени. Поскольку HDFS используется для больших данных, эта файловая система ориентирована на файлы больших размеров (>10 ГБ). Файл состоит из блоков, размер которых больше, чем в обычных файловых системах.

Блок HDFS представляет собой файл типа BLOB (с размером 128 МБ по умолчанию) в файловой системе, над которой строится HDFS. Размер блока можно увеличить до 256 МБ.

> **ВАЖНО**
>
> Файлы, сжатые с использованием определенных кодеков (например, gzip, zlib), не разбиваются на блоки (**non-splittable**). Поэтому для их обработки всегда будет использован один Mapper. Если вы хотите, чтобы один файл обрабатывался параллельно несколькими Mapper — используйте **splittable**-форматы, например bzip2.

В Hadoop можно выполнять следующие операции с файлами:

- запись
- чтение
- удаление
- репликация

Процесс создания (записи) файла HDFS состоит из следующих шагов:

1. Клиент разделяет исходные данные на части размером, равным размеру блока.
2. Клиент подключается к NameNode и требует начать процедуру записи файла, указав фактор репликации для этого файла.
3. NameNode возвращает список DataNodes, где следует разместить все реплики первого блока данных.
4. Клиент посылает запрос на запись первого блока в первую ноду по списку. Если не удается установить соединение, клиент выбирает следующую ноду из списка и так далее.
5. Первая нода записывает блок и передает его копию в следующую ноду по списку. Эта нода записывает блок и передает его копию следующей ноде и так далее.
6. После завершения записи блока DataNodes посылают сообщение об успешном завершении операции по цепочке в обратном направлении к клиенту.
7. После получения первого подтверждения об успешной записи клиент уведомляет NameNode о записи блока и получает список нод для записи второго блока и так далее.

Клиент переходит к записи следующего блока, если удается записать предыдущий блок хотя бы на одной ноде. Репликация блоков происходит в фоновом режиме по событийному (**eventual**) принципу путем рассылки блоков между нодами для создания требуемого количества реплик.

Репликация

HDFS обеспечивает надежный способ хранения больших объемов данных в распределенной среде. Система реплицирует блоки файлов для обеспечения отказоустойчивости на уровне блоков. Каждый блок имеет несколько копий в HDFS.

Принципы хранения

HDFS разделяет файлы на блоки и сохраняет каждый блок в трех DataNodes (по умолчанию фактор репликации равен 3). Например, для сохранения файла размером 128 МБ в HDFS вам потребуется пространство общим объемом 384 МБ (3 * 128 МБ), потому что единственный блок данного файла должен быть записан в виде трех копий на разных DataNodes.

Обратите внимание на то, что одна нода DataNode не может хранить более одной копии того же блока. Обычно HDFS сохраняет две копии в нодах, расположенных в одной стойке, и еще одну копию на ноде вне этой стойки. Рекомендуется задавать фактор репликации равным не менее трем, чтобы даже при выходе из строя целой стойки одна копия оставалась целой.

> **ПРИМЕЧАНИЕ**
>
> - Топология (распределение по стойкам) настраивается вручную. Для получения дополнительной информации обратитесь к статье Rack awareness.
> - Можно задать фактор репликации для каждого файла HDFS отдельно.

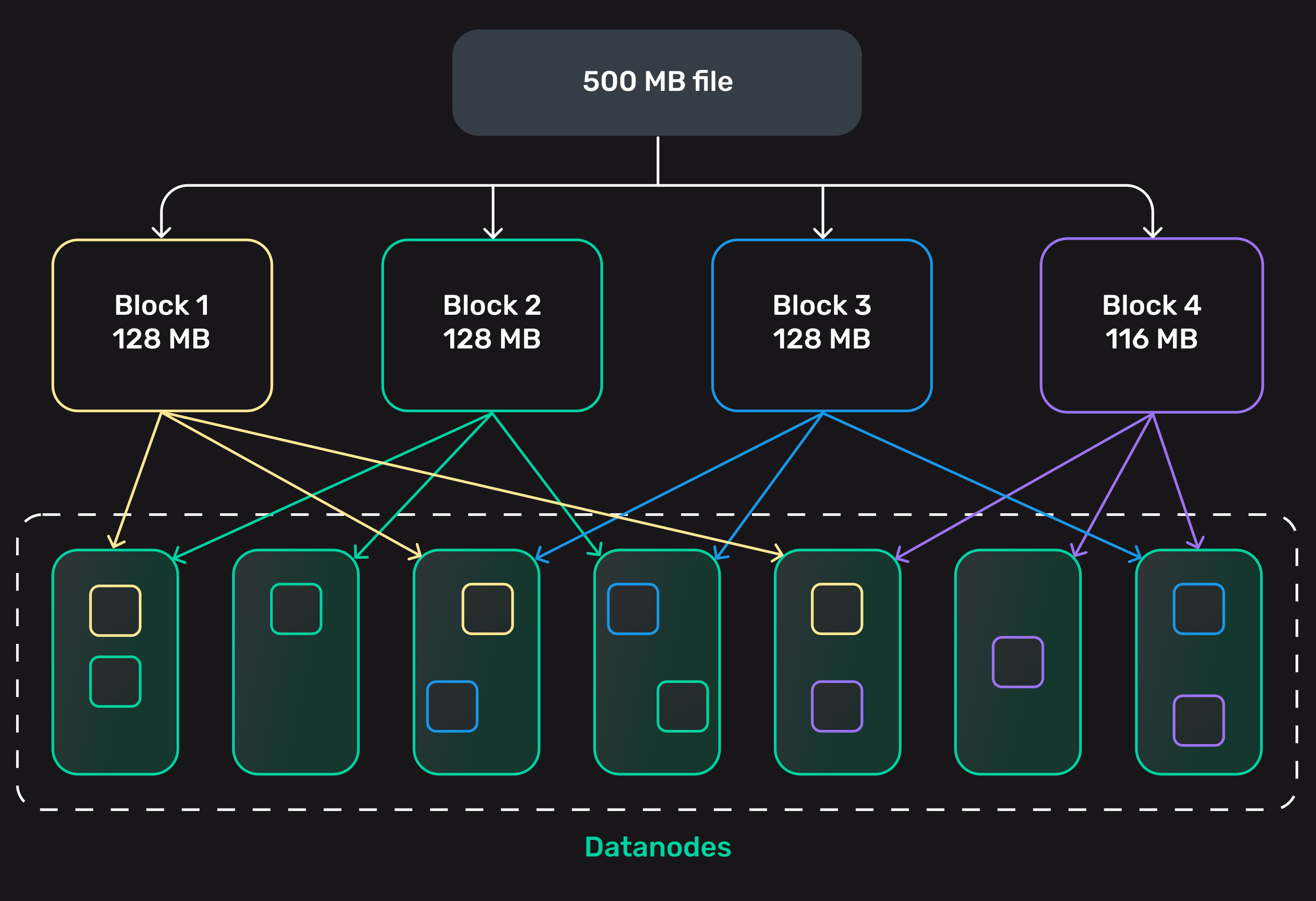
Разделение файлов HDFS на блоки имеет следующие преимущества:

- Очень большие файлы хорошо умещаются на дисках малого объема.
- Остается меньше неиспользуемого пространства на диске, потому что любой диск может уместить большое количество блоков объемом 128 МБ.
- Это оптимизирует передачу данных путем распределения этого процесса на множество нод. Например, если файл состоит из десяти блоков, хранящихся на десяти нодах, то при чтении клиентом этого файла нагрузка распределяется на все десять нод.

Принципы работы

По умолчанию фактор репликации равен трем, а размер блока — 128 МБ. Когда клиент создает файл в HDFS, он сначала разделяет файл на части по 128 МБ и затем записывает их в блоки HDFS. Размер каждого блока, за исключением последнего, будет 128 МБ. Размер последнего блока может быть 128 МБ или меньше в зависимости от общего объема записываемых данных. Размер блока по умолчанию можно настраивать.

Рассмотрим пример записи 500 МБ данных в файл HDFS. Эти данные будут разделены на 3 блока по 128 МБ и четвертый блок размером 116 МБ, как представлено на следующей схеме.



Распределение блоков HDFS

Все блоки распределяются между нодами в соответствии с фактором репликации. Каждая копия блока хранится в одной ноде, которая не может иметь еще одну копию того же блока.

Процесс чтения данных файла HDFS похож на получение данных через сеть torrent. Файл, содержащий некоторый фильм, разбивается на несколько частей, которые передаются из разных компьютеров параллельно. Это помогает значительно ускорить получение этого файла.

Можно задавать фактор репликации как для всей системы в целом (значение по умолчанию), так и для каждого файла или каталога отдельно. Для незначимых файлов можно этот фактор делать меньше, а для наиболее значимых файлов увеличивать.

Если какая-то DataNode становится недоступной или в ней происходит сбой, то NameNode требует от тех нод, которые хранят копии ставших недоступными блоков, начать репликацию этих блоков на другие ноды для обеспечения требуемых факторов репликации соответствующих файлов.

Масштабируемость

Альберт Багдасарян

Назад к содержанию

Для поддержки возрастающей нагрузки кластер Hyperwave легко масштабировать путем добавления серверов или даже стоек с серверами и увеличения объема оперативной памяти в узлах. После подобных изменений сначала значительно возрастает несбалансированность нагрузки, однако увеличение хранилища и вычислительной мощности стоит этих временных явлений. Поскольку масштабируемость серьезно зависит от возможностей узлов NameNode, обратите на них свое внимание в первую очередь.

Воспользуйтесь следующими рекомендациями при масштабировании кластера:

1. Убедитесь, что в непосредственной близости от оборудования кластера есть еще свободное пространство для размещения необходимого количества дополнительных стоек с серверами.
2. Планируйте изменения в сетевой структуре для подключения дополнительных серверов.
3. Рассмотрите возможность добавления дисков и оперативной памяти в существующие серверы, а также установки дополнительных процессоров в свободные разъемы. Это позволит расширить возможности кластера без установки дополнительных серверов и изменений в сетевой структуре.
4. Обновление оборудования в работающем кластере требует значительного времени и трудовых ресурсов. Рекомендуем расширять его путем установки серверов последовательно по одному.
5. В списках процессоров у поставщиков нужные модели не всегда доступны. Если планируете установить второй процессор в сервер, узнайте у продавца, когда понизят цены на требуемую модель и приобретите процессор по более выгодной цене. Снижение цены может произойти в течение 18 месяцев после начала их выпуска.
6. Скорее всего, необходимо также рассмотреть необходимость увеличения оперативной памяти в мастер-серверах.

Обзор работы TEZ

Константин Алпашкин

Содержание

Назад к содержанию

Интерфейс Tez

ADH-сервис Tez основан на проекте Apache TEZ. Вы можете использовать данный сервис для создания высокопроизводительных batch/интерактивных приложений, управляемых YARN. Модель обработки данных в Tez представлена в виде ориентированного ациклического графа (DAG), где вершины графа представляют логику приложения, а ребра — отображают движение данных. Фактически Tez может обработать любой произвольный граф.

В ADH Tez встроен в сервис Hive. При установке Hive сервис Tez используется по умолчанию в качестве механизма выполнения SQL-запросов.

Все настройки сервиса доступны в ADCM-разделе Hive. Раздел с XML-параметрами Tez находится на странице настроек Hive.

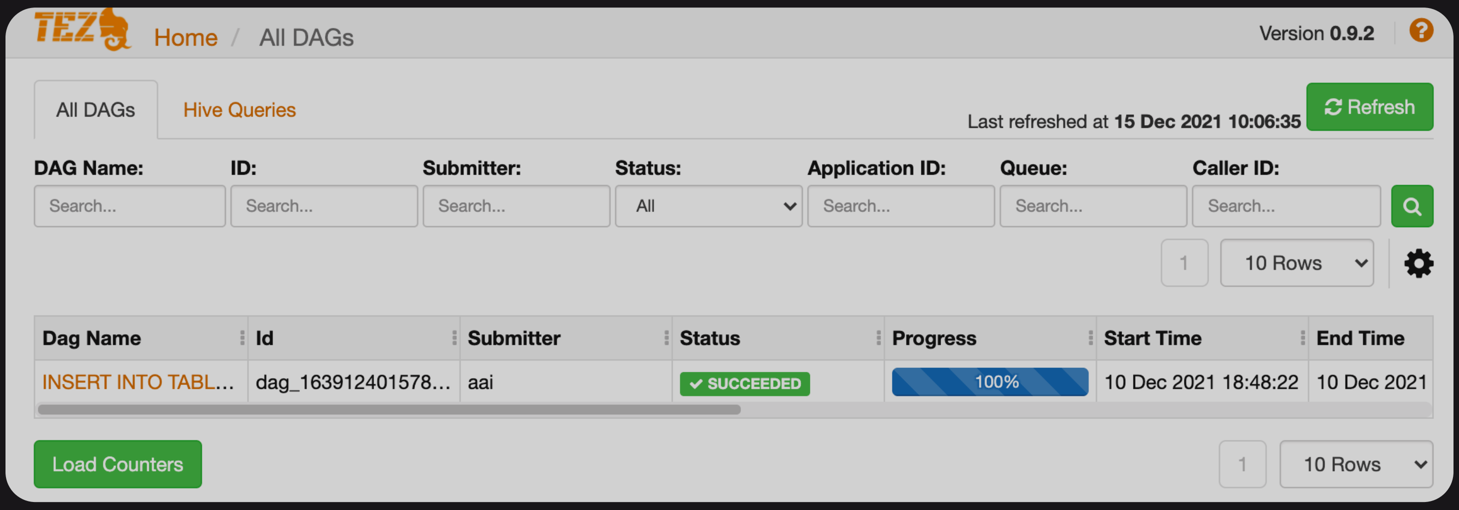
Интерфейс Tez

Ссылку на интерфейс Tez вы можете найти в ADCM на странице Hive. Обычно, она включает полное имя домена (FQDN) и номер порта, который по умолчанию равен `9999`. Tez взаимодействует с YARN Application Timeline Server и отображает текущее и архивное состояние приложений Tez в Tez-интерфейсе. Ссылка на Tez-интерфейс имеет следующий формат: *http://FQDN:9999*.

При попытке доступа к Tez-интерфейсу с удаленного хоста, могут возникнуть проблемы с DNS-разрешением и соединением с YARN Application Timeline Server. Для решения данной проблемы необходимо явно прописать внешний IP-адрес для полного доменного имени в файле *hosts*. Вне зависимости от операционной системы, файл *hosts* обычно расположен в директории *~/etc/*. Добавьте все FQDN-IP пары вашего кластера в этот файл. Файл *hosts* имеет следующий вид:

```
127.0.0.1       localhost
::1         localhost
172.31.1.234 adh1.us-west-2.compute.internal
172.31.3.37 adh2.us-west-2.compute.internal
172.31.2.65 adh3.us-west-2.compute.internal
```

После изменения файла *hosts*, вы можете открыть интерфейс Tez с внешнего хоста.



Last refreshed at **15 Dec 2021 10:06:35**

DAG Name: Search...
ID: Search...
Submitter: Search...
Status: All
Application ID: Search...
Queue: Search...
Caller ID: Search...

Dag Name	Id	Submitter	Status	Progress	Start Time	End Time
INSERT INTO TABL…	dag_163912401578…	aai	✓ SUCCEEDED	100%	10 Dec 2021 18:48:22	10 Dec 2021

Интерфейс Tez

Таблицы Iceberg

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
  - Отличия от таблиц Hive
  - Возможности
- Архитектура
  - Каталог Iceberg
  - Слой метаданных
  - Слой данных
  - Жизненный цикл запроса
- Примеры использования
  - Branching
  - Tagging
  - Time travel
  - Rollback
  - Schema evolution

Обзор

Apache Iceberg — это открытый формат таблиц для больших хранилищ данных, который обеспечивает ACID-транзакции, перемещение во времени (time travel), изменение схемы (schema evolution), изменение партиции (partition evolution) и предоставляет другие возможности работы с данными.

Он функционирует как слой абстракции между движком запросов и данными. Таблицы Iceberg работают аналогично SQL-таблицам и могут интегрироваться с вычислительными движками, такими как Spark, Hive, Impala и Trino.

Iceberg решает проблемы традиционных форматов таблиц, таких как формат таблиц Hive, которые не могут гарантировать согласованность данных, имеют проблемы с производительностью, не позволяют менять схему таблицы и не поддерживают согласованные параллельные записи.

Еще одной важной особенностью таблиц Iceberg является надежность. Iceberg позволяет отслеживать изменения в данных и откатывать хранилища к желаемому состоянию.

ПРИМЕЧАНИЕ

В парадигме ADCM Iceberg не является сервисом. Управление функциональностью Iceberg осуществляется за счет параметров конфигурации для сервисов Hive, Spark и Impala. Эти параметры позволяют измененить способ взаимодействия сервисов с данными. Iceberg-таблицы используются по умолчанию.

Для получения дополнительной информации о таблицах Iceberg обратитесь к спецификации.

Отличия от таблиц Hive

Традиционно в качестве формата таблиц для озер данных использовался формат таблиц Hive. Он отслеживает данные на уровне директорий, поэтому при работе с данными ему необходимо выполнять операции получения списка файлов.

Когда необходимо выполнить много таких операций, могут появиться проблемы с производительностью. Кроме того, существует вероятность потери данных, когда операции со списком файлов выполняются в объектном хранилище типа S3.

В отличие от таблиц Hive, Iceberg отслеживает полный список всех файлов в таблице, используя персистентные деревья. Изменения в таблице происходят на уровне атомарного коммита объекта/файла, который затем обновляет путь к новому файлу метаданных. Этот файл содержит местоположение всех отдельных файлов данных.

Возможности

Основные возможности таблиц Iceberg:

- **Выражения SQL**
  Iceberg поддерживает множество SQL-операций, которые позволяют выполнять такие задачи, как обновление строк, объединение данных, целевые удаления и т.д. Он также позволяет перезаписывать файлы и поддерживает запросы вроде `INSERT INTO`, `MERGE INTO` и `INSERT OVERWRITE`.
- **Изменение схемы таблицы**
  Изменение схемы (schema evolution) таблиц Iceberg влияет только на метаданные, а не на сами файлы данных. Это позволяет быстро добавлять, удалять, обновлять, реорганизовывать (reordering) и переименовывать столбцы в таблице.
- **Изменение партиции**
  В отличие от стандартного партиционирования, таблицы Iceberg позволяют изменять схему партиционирования данных (partition evolution) без необходимости перезаписи всех данных.
- **Снепшоты**
  Снепшоты позволяют сохранить состояние таблицы в конкретный момент. Iceberg ведет лог созданных снепшотов, что позволяет выполнять запросы time travel.
- **Перемещение во времени и откат изменений**
  Функция "перемещения во времени" (time travel) позволяет выполнять запросы к более ранним версиям таблиц. Функция отката изменений (rollback) позволяет пользователям восстанавливать состояние таблицы в определенный момент времени.
- **Транзакционная согласованность**
  Iceberg поддерживает ACID-транзакции, что обеспечивает безопасные параллельные записи в кластере и предотвращает влияние операций записи на операции чтения. Когда происходят изменения, Iceberg генерирует новую, неизменяемую версию файлов данных и метаданных таблицы.
- **Быстрое выполнение запросов**
  Iceberg повышает скорость выполнения запросов за счет возможности организации инкрементальной обработки данных, быстрого планирования сканирования и фильтрации неактуальных данных.

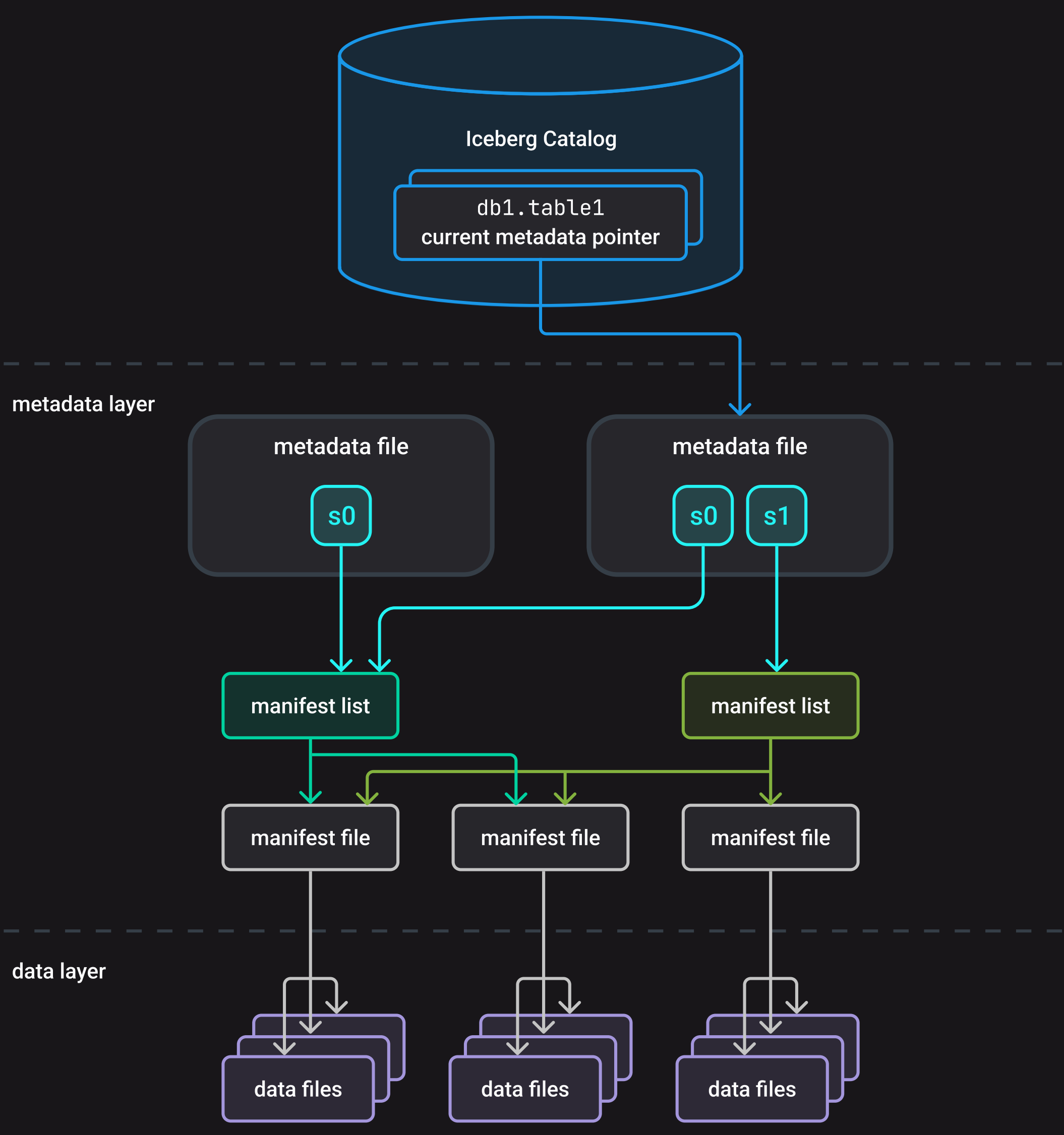
ПРИМЕЧАНИЕ

Доступные возможности зависят от используемого сервиса и его версии. Для уточнения информации о доступных возможностях Iceberg в конкретном сервисе обратитесь к документации этого сервиса.

Архитектура

В архитектуре таблиц Iceberg три основных слоя:

- Каталог Iceberg
- Слой метаданных
- Слой данных



Архитектура таблиц Iceberg

Каталог Iceberg

В Iceberg группа таблиц объединяется в пространство имен. Каталог Iceberg отслеживает эти пространства имен и сами таблицы, а также ссылки или указатели на соответствующие файлы метаданных.

Каталог Iceberg — это надстройка над существующим сервисом управления метаданными. Например, HDFS, Hive metastore или реляционная база данных (JDBC).

В ADH в качестве реализации каталога Iceberg используется Hive metastore.

Слой метаданных

Apache Iceberg использует три типа файлов метаданных для поддержания карты расположения файлов:

- **Файлы манифеста (manifest files)** — каждый файл манифеста отслеживает подмножество других файлов в таблице для одного снепшота. Они отслеживают отдельные файлы, их партиции и метрики столбцов.
- **Списки манифестов (manifest lists)** — описывают снепшот таблицы и содержат список всех файлов манифеста, которые составляют этот снепшот. Списки манифестов также содержат метаданные для каждого файла манифеста в данном снепшоте.
- **Файлы метаданных** — хранят описание таблицы, списки манифестов, снепшоты, схемы таблицы и партиций.

Слой данных

Iceberg отслеживает каждый файл в таблице и сохраняет информацию о его партиции и столбцах.

Есть две категории файлов в слое данных:

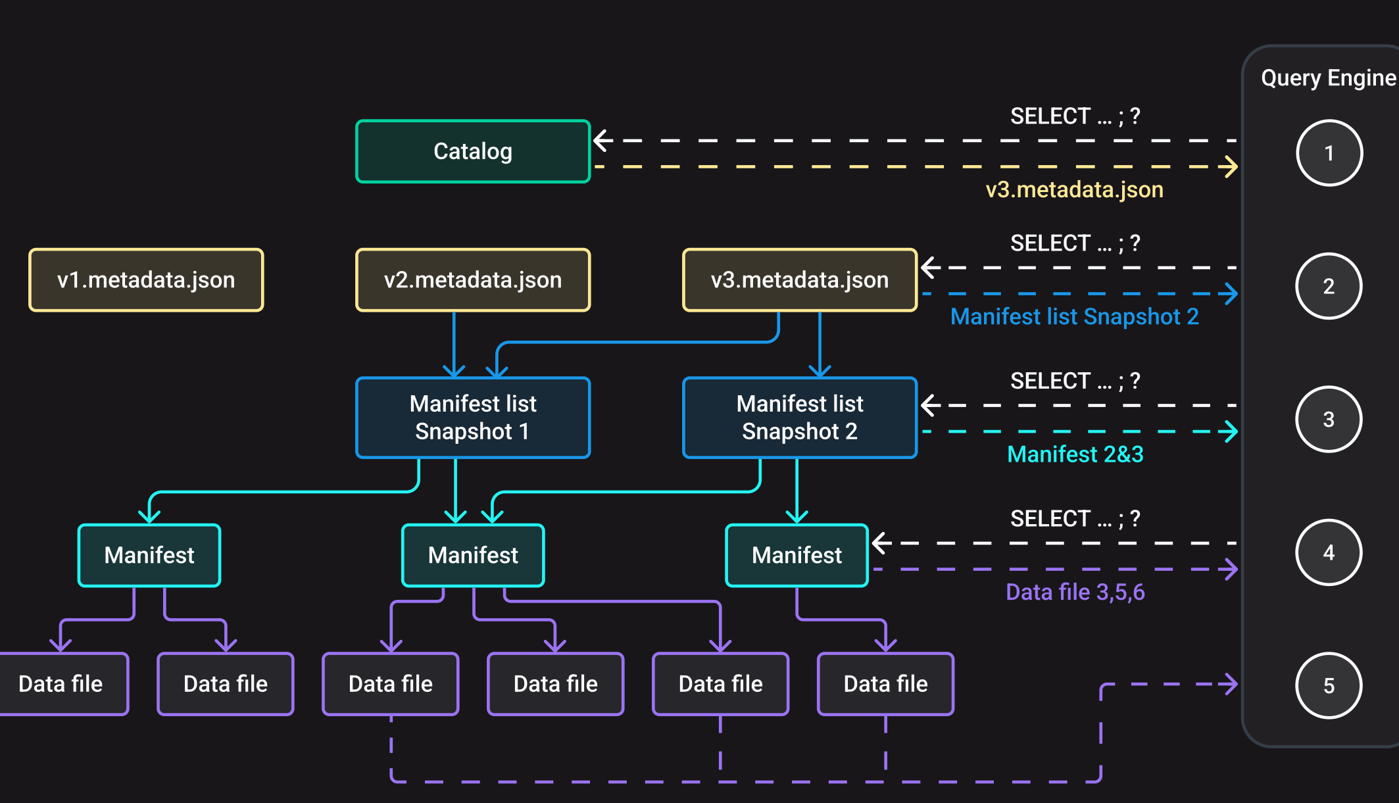
- **Data files** — файлы самих данных в форматах Parquet, Avro или ORC.
- **Delete files** — записи о файлах, которые были помечены как удаленные.

Жизненный цикл запроса

Чтобы проиллюстрировать, как простой запрос работает в кластере с использованием таблиц Iceberg, можно разбить жизненный цикл запроса на следующие этапы:

1. Когда пользователь отправляет запрос, движок запрашивает в каталоге Iceberg путь к текущему файлу метаданных таблицы.
2. Из файла метаданных движок получает следующую информацию:
   - Схему таблицы.
   - Информацию о партиции таблицы. Это поможет исключить файлы, которые не имеют отношения к запросу.
   - Список файлов манифеста текущего снепшота. Это поможет определить, какие файлы сканировать.
3. Затем движок сканирует полученный список файлов манифеста. Каждый манифест содержит список файлов, в которых находятся данные выбранного снепшота. В значении `partition-spec-id` указана схема партиционирования, использованная при создании снепшота, и информация о партициях (`PartitionFieldSummary`), которая определяет необходимость сканирования манифеста.
4. Далее движок параллельно проверяет каждый из выбранных файлов манифеста и считывает метаданные. В метаданных содержатся значения `schema-id` и `partition-id` для каждого файла, а также тип файла (data file или delete file). Эти данные позволяют определить, релевантен ли файл данных для запроса.
5. Движок сканирует релевантные файлы и возвращает результат.

Выполнение запроса в таблице Iceberg значительно быстрее по сравнению с другими форматами таблиц, так как движок может игнорировать нерелевантные файлы. Это позволяет сильно сократить объем данных, которые нужно сканировать.



Жизненный цикл запроса в таблицах Iceberg

Примеры использования

Ниже приведены примеры SQL-запросов, иллюстрирующие некоторые функции таблиц Iceberg.

Branching

Создание веток (branching) в Iceberg-таблицах устроено аналогично системам контроля версий, таким как Git. Вы можете создать локальную ветку, внести необходимые изменения и объединить ее с основной веткой.

Пример команды для создания ветки:

```
ALTER TABLE test.db.table CREATE BRANCH 'task_1';
```

Этот синтаксис создает ветку с тем же состоянием, что и текущая таблица. Чтобы создать ветку для другой версии таблицы, используйте оператор `FOR`.

Пример команды для добавления новых значений в созданную ветку:

```
INSERT INTO test.db.table.task_1 VALUES (1, 'a'), (2, 'b');
```

Чтобы прочитать данные из определенной ветки, используйте синтаксис:

```
SELECT column1, column2 FROM test.db.table.task_1;
```

Дополнительную информацию о создании веток можно получить в статье Branching and Tagging.

Tagging

Iceberg позволяет маркировать снепшоты, чтобы вернуться к ним позже. Теги можно создавать, запрашивать и удалять.

Пример синтаксиса для создания тега:

```
ALTER TABLE test.db.table CREATE TAG `test` RETAIN 10 DAYS;
```

Time travel

С помощью функции "перемещения во времени" (time travel) можно делать запросы к определенному состоянию таблицы во времени.

Пример синтаксиса запроса к таблице в определенный момент времени:

```
SELECT * FROM test.db.table TIMESTAMP AS OF '2007-09-03 04:21:00';
```

Пример синтаксиса запроса к снепшоту:

```
SELECT * FROM test.db.table VERSION AS OF 10726387402803;
```

Пример синтаксиса запроса к вершине (HEAD) определенной ветки:

```
SELECT * FROM test.db.table VERSION AS OF 'task_1';
```

Более подробно о функции time travel читайте в разделе Time travel.

Rollback

С помощью функции отката изменений (rollback) можно восстановить одно из предыдущих состояний таблицы. Состояние может быть передано в виде ID снепшота либо в виде временной метки.

Пример синтаксиса для восстановления таблицы из снепшота:

```
ROLLBACK TABLE test.db.table TO SNAPSHOT '2894821635228246189';
```

Schema evolution

С помощью таблиц Iceberg можно менять структуру таблицы: добавлять, удалять, переименовывать или переставлять столбцы.

Поскольку Iceberg обновляет только метаданные при изменении схемы таблицы, данные не меняются, когда обновляется схема.

Пример синтаксиса для изменения имени столбца:

```
ALTER TABLE test.db.table CHANGE COLUMN names employee_names STRING;
```

Архитектура Impala

Евгения Кузина

Содержание

Назад к содержанию

- Обзор
- Физическое представление
- Компоненты сервиса Impala в ADH
  - Impala Daemon
  - Impala Statestore
  - Impala Catalog Service
  - Impala Client
- Роль Impala в ADH
  - Impala и Hive
  - Метаданные Impala и Metastore
  - Impala и HDFS
  - Impala и HBase

Обзор

Impala обеспечивает быстрые интерактивные SQL-запросы к данным, хранящимся в HDFS, HBase или S3-хранилище. В дополнение к унифицированной платформе хранения Impala также использует те же метаданные, синтаксис SQL (Hive SQL) и драйвер JDBC, что и Apache Hive. Это делает Impala унифицированной платформой для запросов в режиме реального времени или пакетных запросов.

Impala предназначена для интеграции со стандартными средами бизнес-аналитики и поддерживает актуальные отраслевые стандарты: клиенты могут подключаться через JDBC, аутентификация выполняется с помощью Kerberos или LDAP.

Impala имеет следующие преимущества:

- Интерфейс SQL, который уже знаком специалистам по данным и аналитикам.
- Возможность выполнять запросы над большими объемами данных.
- Распределенные запросы в среде кластера, которые способствуют легкому масштабированию и использованию недорогого стандартного оборудования.
- Возможность совместного использования файлов данных между различными компонентами без операций копирования или экспорта/импорта. Например, можно записывать и преобразовывать данные с помощью Hive и делать выборку с помощью Impala.

Решение на основе Impala состоит из следующих частей:

- **Клиенты** — Hue, клиенты JDBC и Impala Client могут взаимодействовать с Impala. Эти интерфейсы используются для выполнения запросов или административных задач, таких как подключение к Impala.
- **Hive Metastore** — хранит информацию о данных, доступных для Impala. Например, Metastore сообщает Impala, какие базы данных доступны, и какую они имеют структуру. Если вы создаете, удаляете или изменяете объекты, а также загружаете данные в таблицы с помощью SQL-операторов Impala, соответствующие метаданные изменяются автоматически.
- **Impala** — процессы, работающие на нодах DataNode, которые координируют и выполняют запросы. Каждый процесс Impala может получать, планировать и координировать запросы от клиентов. Запросы распределяются между нодами с Impala, и процессы Impala на этих нодах затем действуют как рабочие процессы, выполняя параллельные фрагменты запросов.
- **HBase и HDFS** — хранилища данных, на которых могут выполняться запросы.

Пользовательские приложения отправляют SQL-запросы в Impala через JDBC. Они могут подключаться к любому демону Impala в кластере. Этот демон становится координатором запроса. Impala анализирует запрос и распределяет задачи, которые должны быть выполнены, между экземплярами Impala в кластере. Выполнение запланировано таким образом, чтобы получить оптимальную эффективность. Локальные экземпляры Impala обращаются к сервисам HDFS и HBase для получения данных. Каждый демон Impala возвращает данные координатору, который агрегирует результаты и отправляет их клиенту.

Физическое представление

По умолчанию Impala хранит файлы данных непартицированных таблиц непосредственно в корневом каталоге. Однако все файлы данных, расположенные в любом каталоге ниже корневого, являются частью табличного набора данных. Это распространенный подход к работе с непартицированными таблицами, и Apache Hive также использует этот метод.

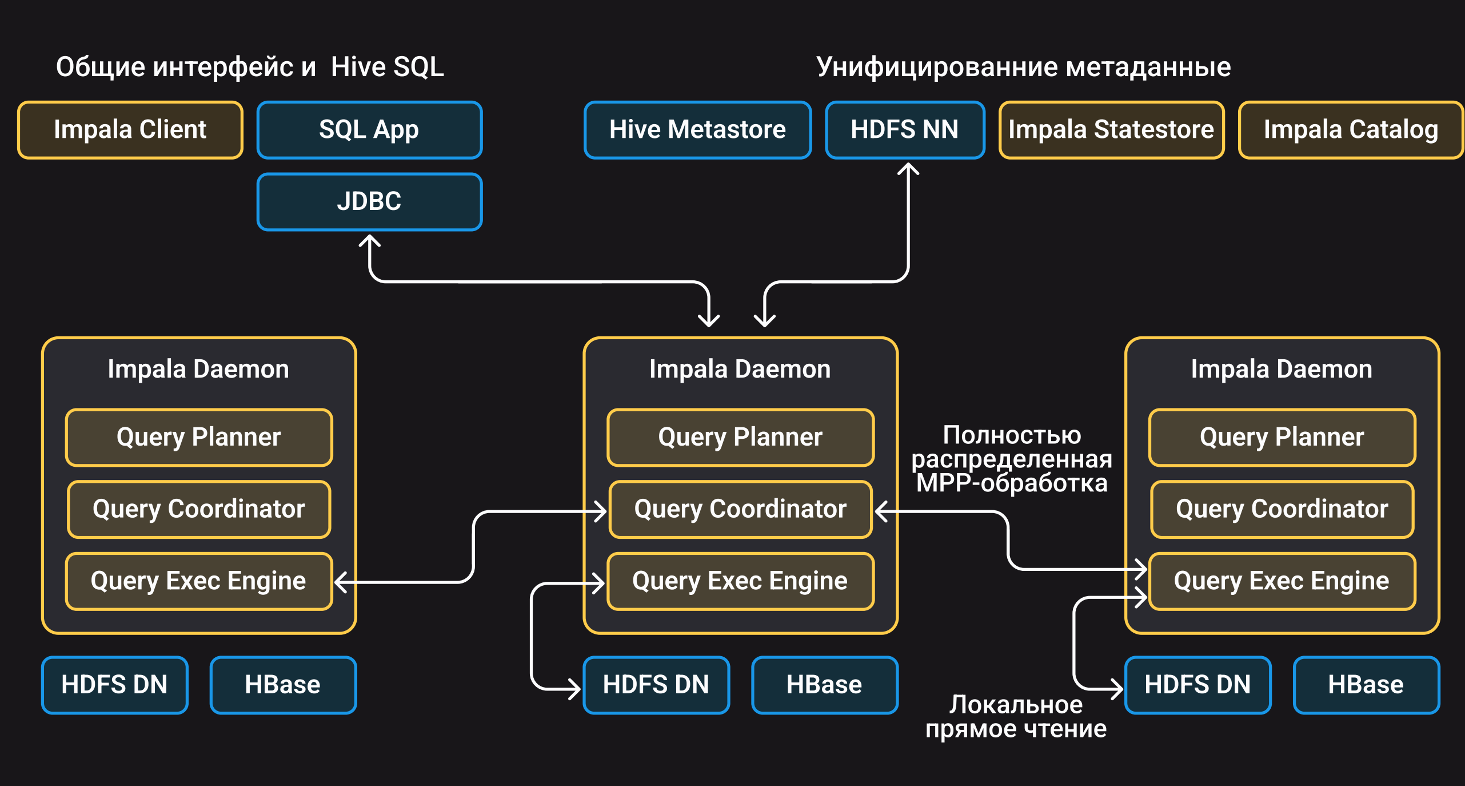
Файлы данных партицированной таблицы размещаются в подкаталогах. Путь отражает значения партицированного столбца. Например, для числа `17` месяца `2` таблицы `T` все файлы данных будут расположены в каталоге `<root>/day=17/month=2/`. Обратите внимание, что эта форма партицирования не подразумевает совместного размещения данных отдельной партиции. Блоки файлов данных, принадлежащих одной партиции, распределяются по узлам данных HDFS.

Impala также дает пользователю большую гибкость при выборе форматов файлов. Она поддерживает наиболее популярные форматы файлов: Avro, RC, Sequence, простой текстовый формат и Parquet. Эти форматы можно комбинировать с различными алгоритмами сжатия, такими как snappy, gzip и bz2. Вы можете указать формат в выражениях `CREATE TABLE` или `ALTER TABLE`. Также можно выбрать отдельный формат для каждой партиции таблицы.

В большинстве случаев рекомендуется использовать Apache Parquet, так как он является современным колумнарным форматом с открытым исходным кодом, обеспечивающим высокое сжатие и высокую эффективность сканирования. Большинство платформ на основе Hadoop, включая Hive, Pig, MapReduce и Cascading, поддерживают Parquet.

Компоненты сервиса Impala в ADH

Impala представляет собой движок распределенной базы данных с массовой параллельной обработкой (MPP). Он состоит из различных процессов, которые работают на хостах кластера.



Архитектура Impala

Impala Daemon

Основным компонентом Impala является ***Impala Daemon***. Он представлен процессом `impalad`. Демон Impala получает запрос от клиента, берёт на себя функции координатора, анализирует запрос, разделяет на подзадачи и распределяет подзадачи по другим нодам с процессами `impalad`. После того как каждый процесс рабочей ноды `impalad` получает подзапрос, он начинает выполнять локальный запрос (например, запрашивать HDFS DataNode или region-сервер HBase) и возвращает результаты запроса центральному координатору. Координатор собирает результаты запросов от других процессов `impalad`, объединяет их и отправляет результат клиенту.

Демон Impala состоит из трех модулей:

- **Query planner** — получает запросы от SQL APP, JDBC и других клиентов и преобразует во множество подзапросов (создает план выполнения).
- **Query coordinator** — распределяет подзапросы среди нод Impala.
- **Query executor** — отвечает за выполнение подзапросов и возврат их результатов координатору.

Демоны Impala могут быть развернуты одним из следующих способов:

- HDFS и Impala расположены совместно, и демон Impala работает на каждом хосте, являющемся DataNode.
- Impala развертывается отдельно в вычислительном кластере и получает данные удаленно из HDFS, S3 и других хранилищ.

Демоны Impala находятся в постоянном взаимодействии с Statestore, чтобы иметь информацию о том, какие демоны в рабочем состоянии и могут принимать новые запросы.

Они также получают широковещательные сообщения от сервиса Impala Catalog всякий раз, когда какой-либо демон Impala в кластере создает, изменяет или удаляет объекты любого типа, или когда Impala обрабатывает оператор `INSERT` или `LOAD DATA`.

Вы можете контролировать, какие ноды работают как координаторы запросов, а какие — как исполнители, чтобы улучшить масштабируемость высококонкурентных рабочих нагрузок в больших кластерах. Для получения дополнительной информации см. How to configure Impala with dedicated coordinators.

Impala Statestore

Компонент ***Impala Statestore*** собирает информацию о работоспособности всех демонов Impala и постоянно пересылает результаты всем нодам с процессом `impalad`. Statestore представлен демоном `statestored`. Кластеру нужен этот процесс только на одном хосте. Если демон Impala отключается из-за аппаратного сбоя, сетевой ошибки, проблемы с программным обеспечением или по другим причинам, Statestore информирует об этом все другие демоны Impala, и новые запросы не отправляются недоступному демону.

Поскольку Statestore играет важную роль только когда нормальная работа кластера Impala нарушена, а также транслирует метаданные координаторам, его функции не всегда являются критическими. Если Statestore становится недоступным, демоны Impala продолжают работать и распределять нагрузку между собой, как обычно при работе с данными, известными Impala. Кластер может стать менее надежным, если некоторые демоны Impala выйдут из строя. Метаданные также становятся менее согласованными, поскольку они изменяются, пока Statestore недоступен. Когда Statestore возвращается к работе, он восстанавливает связь с демонами Impala и возобновляет функции мониторинга и вещания.

Если вы выполните выражение DDL, когда Statestore недоступен, запросы, обращающиеся к новому объекту, созданному DDL, завершатся ошибкой.

Impala Catalog Service

Компонент ***Impala Catalog Service*** отправляет широковещательные сообщения всем демонам Impala при изменении метаданных, сделанных операторами SQL через Impala. Например, когда создаётся таблица, загружаются данные или выполняются другие операции по изменению таблиц или данных. Он представляет собой процесс `catalogd`.

Этот процесс нужен только на одном хосте в кластере. Поскольку изменения метаданных передаются через демон Statestore, имеет смысл устанавливать компоненты ***Impala Catalog Service*** и ***Impala Statestore*** на одном и том же хосте.

Вы можете использовать параметр ***load_catalog_in_background***, чтобы определить, когда следует загружать метаданные таблицы. Флажок ***load_catalog_in_background*** расположен на вкладке ***CLUSTERS → Кластер ADH → Services → Impala → Components → Impala Catalog Service → Configuration***.

Если установлено значение `false`, метаданные таблицы загружаются при первом обращении к ней. В этом случае первое выполнение запроса может быть медленнее последующих.

Если для параметра ***load_catalog_in_background*** установлено значение `true`, Catalog Service загружает метаданные для таблицы, даже если эти метаданные не требуются ни для одного запроса. Таким образом, метаданные могут быть уже загружены, когда будет выполнен первый запрос к этой таблице. Однако не рекомендуется устанавливать для этого параметра значение `true`. Фоновая загрузка может мешать загрузке метаданных для текущего запроса и приводить к случайным долго выполняемым запросам, которые трудно диагностировать. Impala также может загружать метаданные для таблиц, которые никогда не используются, увеличивая размер каталога и использование памяти как для Catalog Service, так и для демона Impala.

Impala Client

Компонент ***Impala Client*** — это impala-shell, который представляет собой интерфейс командной строки для запросов к демону Impala. Можно установить ***Impala Client*** на один или несколько хостов в сети. Он может удаленно подключаться к любому экземпляру демона Impala.

Роль Impala в ADH

Основная цель Impala — сделать операции SQL-on-Hadoop быстрыми и эффективными. Impala может обмениваться данными с другими компонентами Hadoop как в качестве потребителя, так и в качестве производителя, поэтому она может гибко вписываться в процессы ETL и ELT.

Impala и Hive

Там, где это целесообразно, Impala использует существующую инфраструктуру Apache Hive, которая уже есть у многих пользователей Hadoop, для выполнения длительных, пакетных SQL-запросов. Impala хранит определения таблиц в базе данных MySQL или PostgreSQL, известной как Metastore, в той же базе данных, где Hive хранит данные такого типа. Таким образом, Impala может получить доступ к таблицам, определенным или загруженным через Hive, если во всех столбцах используются поддерживаемые Impala типы данных, форматы файлов и кодеки сжатия.

Первоначальный фокус на функциях запросов и производительности означает, что Impala может считывать больше типов данных с помощью оператора `SELECT`, чем записывать с помощью оператора `INSERT`. Чтобы запросить данные с использованием форматов файлов Avro, RCFile или SequenceFile, вы должны загрузить данные с помощью Hive.

Оптимизатор запросов Impala также может использовать статистику по таблицам и столбцам. Первоначально в Hive эта информация собиралась с помощью оператора `ANALYZE TABLE`. Вместо этого в Impala можно использовать оператор `COMPUTE STATS`. `COMPUTE STATS` требует меньше настроек, более надежен и не требует переключения между impala-shell и оболочкой Hive.

Метаданные Impala и Metastore

Как описано выше, Impala хранит информацию об определениях таблиц в централизованной базе данных, называемой Metastore. Impala также отслеживает метаданные низкоуровневых характеристик файлов данных — физическое расположение блоков в HDFS.

Для таблиц с большим объемом данных и/или множеством партиций получение всех метаданных может быть длительным. Каждый узел Impala кеширует эти метаданные для повторного использования в будущих запросах к той же таблице.

Если определение таблицы или данные в таблице обновлены, все демоны Impala в кластере должны получить последние метаданные и заменить устаревшие кешированные метаданные, прежде чем выполнять запрос к этой таблице. Обновление метаданных автоматически координируется компонентом ***Impala Catalog Service*** для всех операторов DDL и DML, выполняемых через Impala. Для DDL и DML, запущенных через Hive, или изменений, внесенных вручную в файлы HDFS, необходимо использовать оператор `REFRESH` (когда новые файлы данных добавляются в существующие таблицы) или оператор `INVALIDATE METADATA` (для новых таблиц, после удаления таблицы, перебалансировки HDFS и удаления файлов данных). `INVALIDATE METADATA` извлекает метаданные для всех таблиц, отслеживаемых Metastore. Если вы знаете, какие таблицы были изменены за пределами Impala, можно запустить `REFRESH <table_name>` для каждой затронутой таблицы, чтобы получить последние метаданные для этих таблиц.

Impala и HDFS

Impala использует HDFS в качестве основного хранилища данных. Impala полагается на избыточность, обеспечиваемую HDFS, для защиты от сбоев оборудования или сети на отдельных узлах. Данные таблиц Impala представлены в виде файлов данных в HDFS с использованием стандартных форматов файлов HDFS и кодеков сжатия. Когда файлы данных присутствуют в каталоге новой таблицы, Impala читает их все, независимо от имён файлов. Новые данные добавляются в файлы с именами, контролируемыми Impala.

Impala и HBase

HBase — это альтернатива HDFS в качестве хранилища данных Impala. Это система баз данных, построенная поверх HDFS, без встроенной поддержки SQL. Чтобы запросить содержимое таблиц HBase через Impala, определите таблицы в Impala и сопоставьте их с эквивалентными таблицами в HBase. Таким образом, вы можете выполнять даже JOIN-запросы, включающие как таблицы Impala, так и HBase.

Архитектура Kyuubi

Елена Костюченко

Содержание

Назад к содержанию

Обзор
Архитектура
Высокая доступность

Обзор

Apache Kyuubi — это распределенный многопользовательский JDBC-интерфейс для обработки и анализа больших данных. Он построен на базе движка Apache Spark и может запускать любой JDBC-совместимый фреймворк (например, Hive, Flink и т.д.).

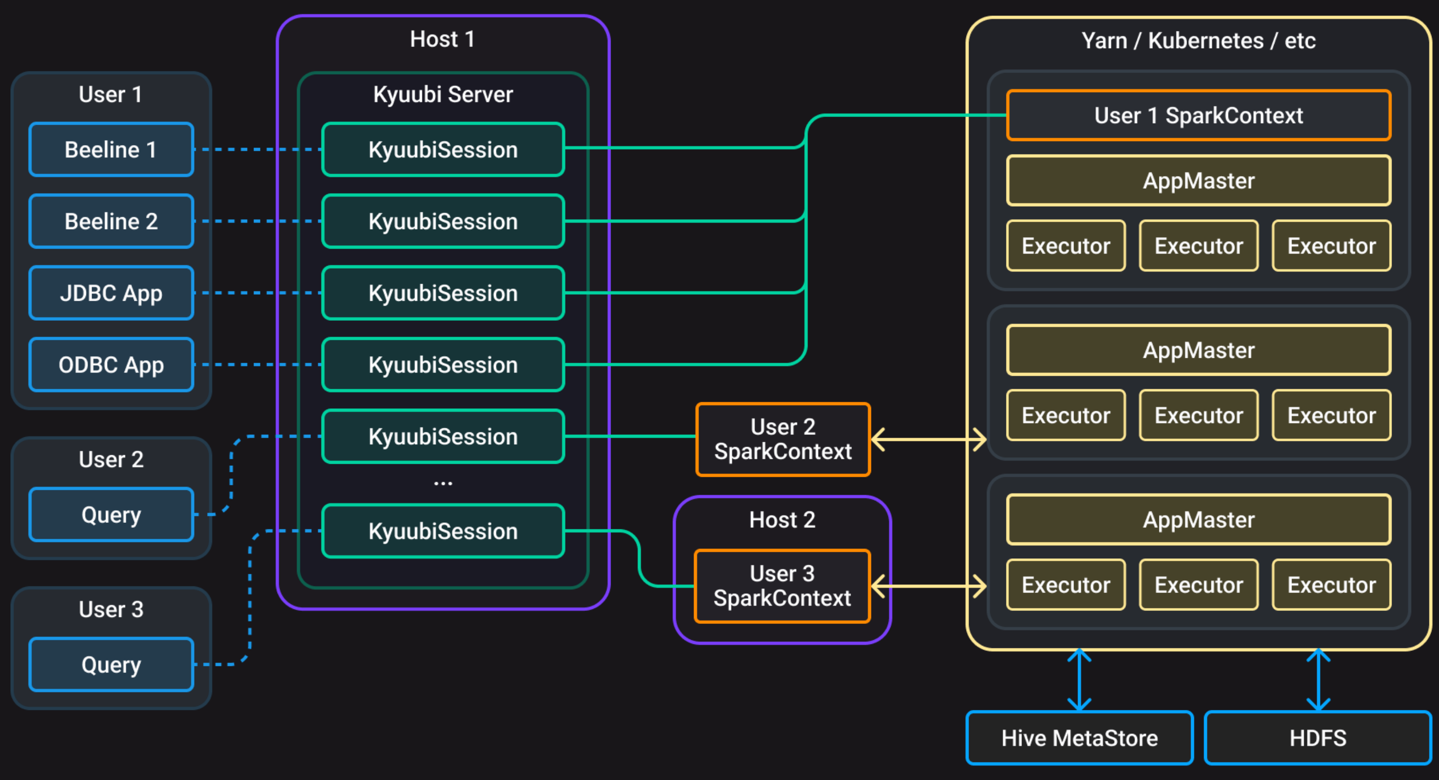
Kyuubi используется как SQL-шлюз для операций на хранилищах данных и озерах данных. Он реализует модуль Hive Service RPC — интерфейс доступа к данным, используемый в HiveServer2. Подробнее о том, что отличает Kyuuby от них, читайте в статье Сравнение Kyuubi, HiveServer, Spark Thrift Server.

Основные возможности Kyuubi:

- **Многопользовательский доступ** — сквозная поддержка доступа нескольких пользователей к данным через единую систему аутентификации и авторизации;
- **Высокая доступность** — балансировка нагрузки через ZooKeeper;
- **Несколько рабочих нагрузок** — поддержка нескольких различных рабочих нагрузок в рамках одной платформы, одной копии данных и одного SQL-интерфейса.

Архитектура

Основным компонентом Kyuubi является **Kyuubi Server** — процесс-демон, который обрабатывает одновременные запросы от клиентов на подключение и выполнение задач и управляет их выполнением.


User 1
Beeline 1
Beeline 2
JDBC App
ODBC App
User 2
Query
User 3
Query
Host 1
Kyuubi Server
KyuubiSession
KyuubiSession
KyuubiSession
KyuubiSession
KyuubiSession
...
KyuubiSession
User 2 SparkContext
Host 2
User 3 SparkContext
Yarn / Kubernetes / etc
User 1 SparkContext
AppMaster
Executor
Executor
Executor
AppMaster
Executor
Executor
Executor
AppMaster
Executor
Executor
Executor
Hive MetaStore
HDFS

Архитектура Kyuubi

Когда пользователь запрашивает доступ к данным через один или несколько клиентов, сервер создает для выполнения задач пользователя отдельную **сессию Kyuubi (Kyuubi session)**. Все запросы этого пользователя привязываются к соответствующей сессии.

Для каждой сессии создается **SparkContext** для обработки запросов пользователей: компиляция, оптимизация и выполнение SQL-инструкций.

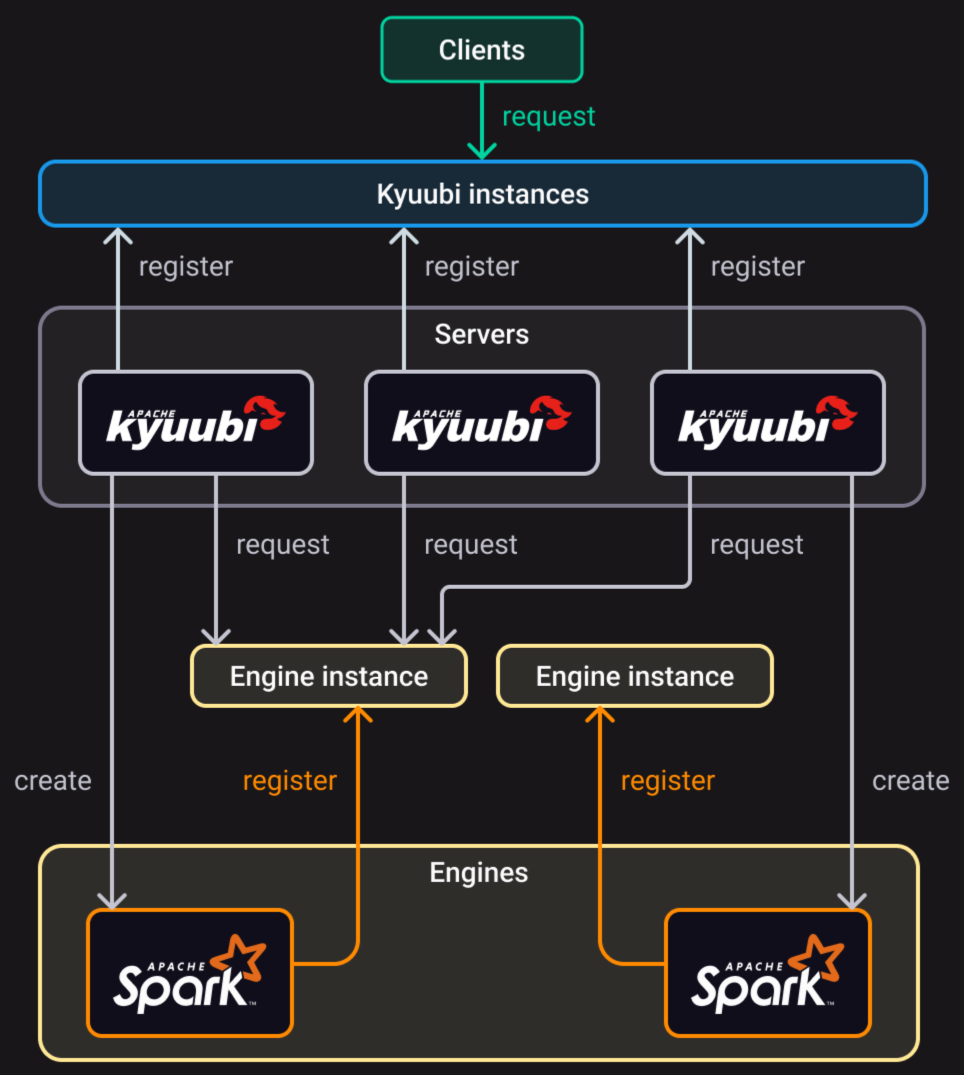
SparkContexts можно создавать локально, в режиме развертывания клиента (`--deploy-mode client`) в рамках одного сервиса, или в кластерах YARN или Kubernetes в режиме развертывания кластера (`--deploy-mode cluster`). SparkContexts сами управляют своим жизненным циклом, кешированием и утилизацией. На них не влияет аварийное переключение на сервере Kyuubi.

Если для SparkContext нет активных сессий, Kyuubi освобождает его ресурсы.

Высокая доступность

Kyuubi обеспечивает высокую доступность (High Availability, HA) и балансировку нагрузки с помощью ZooKeeper.

В режиме HA клиенты могут использовать SparkConexts, созданные другими сервисами Kyuubi. Зарегистрированные в одном пространстве имен сервисы балансируют нагрузку между собой.


Clients
request
Kyuubi instances
register
register
register
Servers
kyuubi
kyuubi
kyuubi
request
request
request
Engine instance
Engine instance
create
register
register
create
Engines
Spark
Spark

Kyuubi в режиме высокой доступности

Когда клиент делает запрос, он выбирает между несколькими серверами. Выбранный инстанс Kyuubi выбирает доступный SparkContext. Если доступных нет, сервер создает новый.

Если тот же пользователь запрашивает новое соединение, сессия может быть создана на другом инстансе Kyuubi, но SparkContext будет использован тот же.

Если другой пользователь делает запрос, весь процесс повторяется. Когда клиент выбирает инстанс Kyuubi, пространства имен, используемые для хранения адресов инстансов движка, изолируются по имени пользователя (по умолчанию) и разные пользователи не могут получить доступ к инстансам других пользователей.

Сравнение Kyuubi, HiveServer, Spark Thrift Server

Елена Костюченко

Назад к содержанию

Kyuubi, Spark Thrift Server и HiveServer используют одни и те же интерфейсы и протоколы. За счет этого они похожи в использовании, но значительно отличаются по некоторым ключевым характеристикам. Главное преимущество Kyuubi и Spark Thrift Server перед HiveServer — это улучшенная производительность.

В таблице ниже перечислены основные отличия Kyuubi от HiveServer и Spark Thrift Server в производительности и доступных возможностях.

Характеристика	HiveServer	Spark ThriftServer	Kyuubi
Интерфейс	HiveJDBC	HiveJDBC	HiveJDBC
Синтаксис SQL	Hive QL	Spark SQL	Spark SQL
Оптимизатор SQL	Hive Optimizer	Spark SQL Catalyst	Spark SQL Catalyst
Парсинг и планирование выполнения SQL-запросов	На сервере	На сервере	В движке
Загрузка UDF	На сервере	На сервере	В движке
Добавление задач	Запрос разбивается на несколько приложений Spark (`RemoteDriver`)	На сервере	Через движки Kyuubi с выбором политики изоляции
Совместимость со Spark	Поддержка одной версии	Встроенный	Поддержка нескольких версий
Управление каталогом	HMS	HMS	HMS
Высокая доступность	Есть	Нет	Есть
Мультиарендность	Есть	Нет	Есть
Управление разрешениями	Стандарт SQL, детализированный (fine-grained)	Нет	Стандарт SQL, детализированный (fine-grained)
Производительность	Неплохая	Хорошая	Хорошая
Параллелизм клиентов	Высокий	Низкий	Высокий
Создание очередей	Для запросов	Нет	Для движков
Настройка ресурсов	Для запросов	Для пулов	Для движков
Управление ресурсами	YARN	Пулы	YARN, Kubernetes и т.д.
Ограничение времени занятости ресурсов	На время выполнения запроса	Без ограничения	В зависимости от уровня общего доступа, поддерживается несколько режимов. Для всех режимов доступно динамическое распределение ресурсов

Политики изоляции в Kyuubi

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
- Share levels
  - CONNECTION
  - USER
  - GROUP
  - SERVER
  - Subdomain
- Конфигурация

Обзор

Kyuubi поддерживает настройку общего доступа (share levels) к движкам и позволяет указывать условия их совместного использования для отдельных сессий, пользователей или групп.

Эта функция обеспечивает большую гибкость в управлении рабочей нагрузкой кластера: более изолированные движки обеспечивают лучшую стабильность работы, а совместное использование движка несколькими пользователями позволяет экономить ресурсы кластера.

Текущие поддерживаемые уровни изоляции:

- CONNECTION
- USER
- GROUP
- SERVER

Уровень общего доступа по умолчанию — USER. Помимо основных четырех уровней общего доступа, Kyuubi позволяет использовать несколько движков для одного пользователя или группы.

Все поддерживаемые уровни общего доступа могут использоваться вместе на одном сервере или кластере Kyuubi.

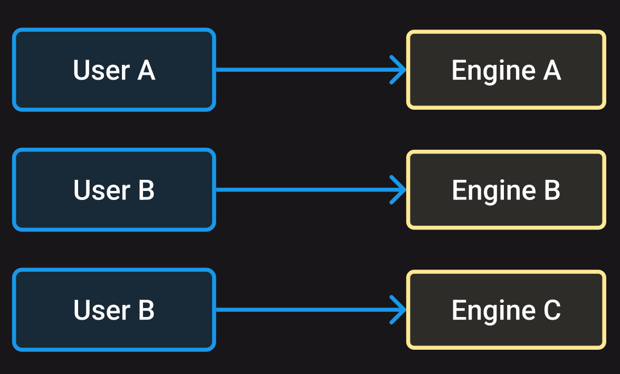
Уровни общего доступа работают одинаково как для конфигураций с одним узлом, так и для кластеров с высокой доступностью (High Availability, HA).

Сравнение уровней общего доступа

Share level	Количество движков	Возможности	Уровень изоляции	Доступность
CONNECTION	Один движок на сессию	Large-scale ad hoc ETL	Высокий	Низкая
USER	Один движок на пользователя	Small-scale ad hoc ETL	Средний	Средняя
GROUP	Один движок на группу	Small-scale ad hoc ETL	Низкий	Высокая
SERVER	Один движок на инстанс	Admin	Если сервер защищен — высокий, если не защищен — низкий	Только для администратора, если сервер защищен

Share levels

CONNECTION

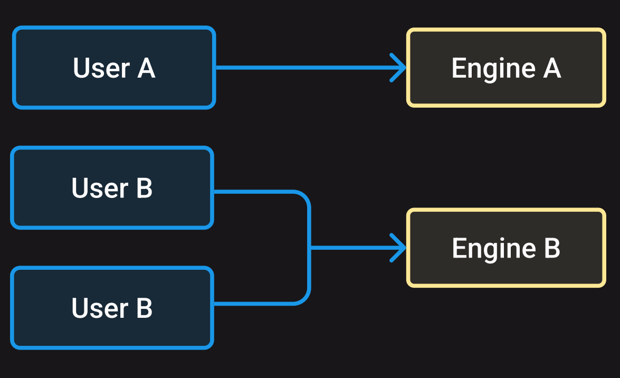


Запросы от клиентов к движкам на уровне общего доступа CONNECTION

Уровень общего доступа CONNECTION создает изолированный движок, доступный только для одной сессии. В рамках этой сессии пользователь может выполнять разные операции, включая вызовы функций или запросы метаданных.

Как только сессия, создавшая движок, закрывается, движок также отключается.

USER



Запросы от клиентов к движкам на уровне общего доступа USER

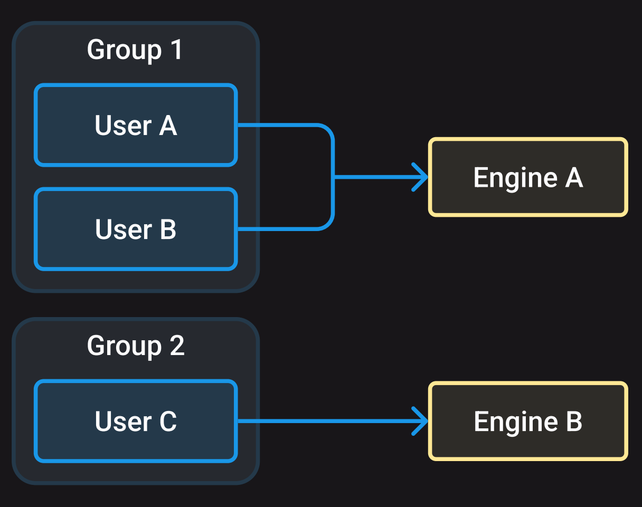
При уровне общего доступа USER движок изолируется по пользователю. У каждого движка есть конкретный пользователь, являющийся владельцем этого движка. Любые запросы, отправленные от этого пользователя с одного или нескольких клиентов, перенаправляются на движок этого пользователя.

Если у пользователя открыто несколько сессий, они будут принадлежать одному и тому же SparkContext, но каждая сессия будет иметь свой собственный экземпляр SparkSession, который содержит отдельное состояние сессии, включая временные представления, конфигурацию SQL, UDF и т.д.

Чтобы создать Singleton-экземпляр SparkSession, который будет использоваться несколькими сессиями, установите для параметра `kyuubi.engine.single.spark.session` значение `true` вручную или с помощью ADCM.

Движок с общим уровнем доступа USER выключается после закрытия всех подключенных к нему сессий и после истечения срока его жизни (TTL).

GROUP



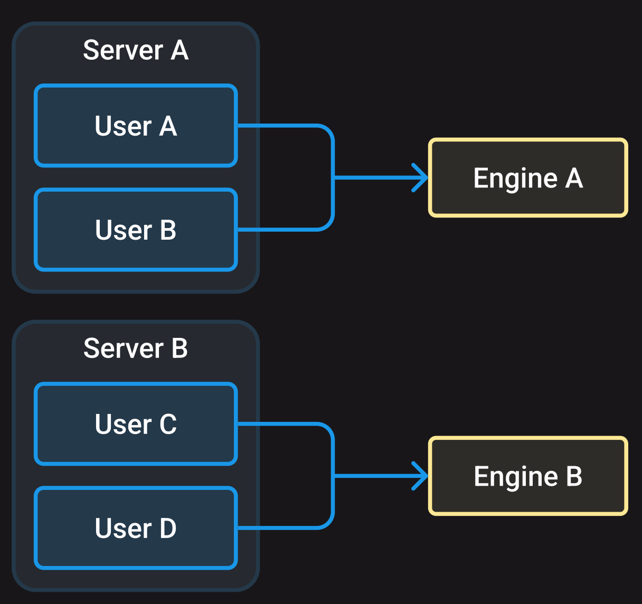
Запросы от клиентов к движкам на уровне общего доступа GROUP

При уровне общего доступа GROUP движок используется пользователями из одной группы совместно. Все сессии пользователей в одной группе направляются к движку этой группы.

В качестве владельца движка указывается название группы. Группа пользователей формируется в соответствии с Hadoop GroupsMapping. Если группа не найдена, уровень общего доступа меняется на USER.

Движок с уровнем общего доступа GROUP отключается после закрытия всех сессий пользователей группы и после истечения его времени жизни (TTL).

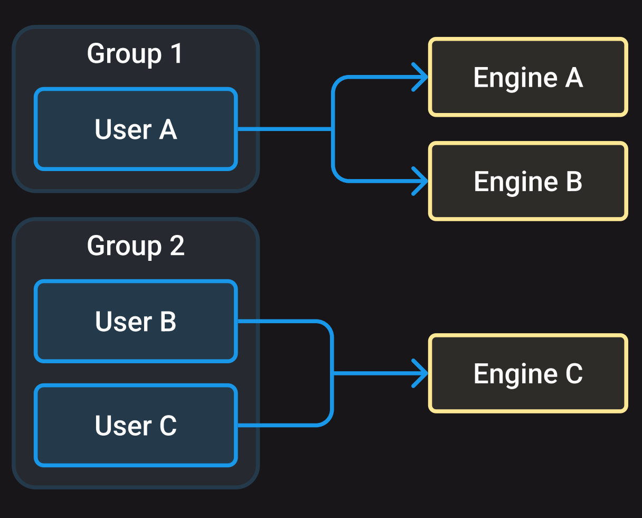
SERVER



Запросы от клиентов к движкам на уровне общего доступа SERVER

При использовании уровня общего доступа SERVER для каждого инстанса Kyuubi создается отдельный движок.

Subdomain



Пример subdomain-конфигурации для уровня общего доступа GROUP

С помощью параметра `kyuubi.engine.share.level.subdomain` можно настроить дополнительные политики изоляции для уровней общего доступа USER, GROUP или SERVER. Конфигурация этого параметра позволяет создавать несколько движков для одного пользователя, группы или сервера.

Если доступно несколько движков, пользователь может указать, какой использовать в URL соединения JDBC.

Конфигурация

Настройки share levels

Параметр	Значение	Значение по умолчанию	Описание
kyuubi.engine.share.level (kyuubi.session.engine.share.level)	USER, CONNECTION, GROUP, SERVER	USER	Тип уровня общего доступа. Можно задать как в файле конфигурации сервера, так и в URL-адресе подключения. У URL более высокий приоритет
kyuubi.session.engine.idle.timeout	Число с индикатором единиц времени	PT30 (30 минут)	Время, которое движок может простаивать. Можно задать как в файле конфигурации сервера, так и в URL-адресе подключения. У URL более высокий приоритет
kyuubi.engine.share.level.subdomain (kyuubi.engine.share.level.sub.domain)	Валидное имя ноды ZooKeeper	—	Дополнительная изоляция движка по znode. Можно задать как в файле конфигурации сервера, так и в URL-адресе подключения. У URL более высокий приоритет

Архитектура MapReduce

Альберт Багдасарян

Содержание

Назад к содержанию

Операции

Компоненты

Алгоритм MapReduce

Вычислительными процессами в кластере Hyperwave на нижнем уровне управляет MapReduce — фреймворк, тесно связанный с YARN. MapReduce основан на Java и предоставляет классы, интерфейсы и рабочее окружение для сборки и исполнения приложений. Через эти компоненты приложение взаимодействует с менеджером ресурсов YARN (Resource Manager) для выполнения своих задач на тех нодах, где установлен Node Manager. HDFS DataNodes и YARN Node Managers установлены на одних и тех же нодах. С помощью них реализуются обе функции Hadoop: хранение данных и их обработка. Поэтому названия Node Manager и DataNode часто используются для обозначения одной и той же ноды, на которой они установлены.

Операции

MapReduce использует следующие операции для обработки данных по запросу приложения:

1. **Split** (разделение). Приложение определяет каким образом MapReduce должен разделять входные данные на отдельные записи, называемые **split**. Например, если приложение объявляет тип входных данных как `TextInputFormat`, то MapReduce разобьет входные данные на строки.

2. **Map** (преобразование). Эту операцию выполняет метод `map`, определенный в классе `Mapper` приложения. Этот класс определяет задачу, реплицируемую на все ноды, где расположены входные блоки HDFS приложения. MapReduce вызывает метод `map` для каждой записи split на ноде, где работает задача. Задача создает множество пар ключ/значение и сохраняет их в локальной файловой системе. MapReduce разделяет выходной файл по ключу, что необходимо для следующей операции.

3. **Shuffle** (перераспределение). Система передает данные, полученные на предыдущей операции, в те ноды, где будет выполняться следующая операция. На вход каждой задачи reduce должны поступить все пары с одинаковым ключом.

4. **Reduce** (агрегирование). Эту операцию выполняет метод `reduce`, определенный в классе `Reducer` приложения. Этот класс определяет задачу, реплицируемую на все ноды, которые выполняют агрегацию данных. MapReduce вызывает метод `reduce` для каждого уникального ключа со всеми его данными, полученными на данной ноде.

В приведенном выше списке приложение MapReduce отвечает за определение классов Java для двух основных стадий обработки:

- Стадия **Map** преобразует входные данные в пары ключ/значение в соответствии с кодом приложения.

- Стадия **Reduce** агрегирует результат, полученный на стадии Map, в другие пары и сохраняет свой результат в HDFS. При каждом вызове задача reduce создает уникальную пару ключ/значение, выполняя таким образом агрегацию данных.

Если оригинальные входные данные хранятся в локальной файловой системе, необходимо скопировать их в HDFS для распределения их по разным нодам кластера. Можно также использовать цепочку заданий MapReduce, когда одно задание принимает на вход данные, созданные другим заданием. Выходные данные задания сохраняются в HDFS для обеспечения надежного хранения. MapReduce и YARN обеспечивают планирование и мониторинг приложений. Если обнаружен сбой при выполнении задачи, то YARN запускает эту задачу заново.

Компоненты

Для выполнения упомянутых ранее операций MapReduce использует следующие компоненты (большинство из них принадлежат YARN):

- Клиент Hadoop необходим для запуска приложения MapReduce.

- Resource Manager управляет выделением ресурсов приложению.

- Node Manager управляет ресурсами в пределах одной ноды.

- Application Master (`MRAppMaster`) управляет ходом выполнения приложения.

- YARN container предоставляет необходимые вычислительные ресурсы для задачи map или reduce, работающей на конкретной ноде кластера, где создан этот контейнер.

Алгоритм MapReduce

Можно использовать клиент Hadoop для запуска приложения как задания MapReduce с помощью следующей команды:

```
$ hadoop jar <jar_file_name> <class_name> <input> <output>
```

Данная команда принимает следующие аргументы (в примере показаны не все возможные аргументы):

- `<jar_file_name>` — путь к JAR-файлу, содержащему пакет с Java-приложением.

- `<class_name>` — название основного класса данного приложения.

- `<input>` — путь к файлу или каталогу с входными данными, а также это может быть шаблон пути, определяющий набор файлов (например, */path/to/folder/test**) в HDFS.

- `<output>` — путь к каталогу HDFS для выходных данных.

MapReduce и YARN выполняют задание MapReduce в следующей последовательности (для краткости некоторые операции приведены в упрощенном варианте):

1. MapReduce (в качестве клиента YARN) поставляет приложение в Resource Manager.

2. Resource Manager создает специальный для данного типа приложений Application Master (для приложений MapReduce — `MRAppMaster`) в контейнере YARN в одной из нод, управляемых Node Manager.

3. Application Master запрашивает у Resource Manager вычислительные ресурсы на тех нодах, где находятся входные блоки данных. Затем он запрашивает Node Manager, расположенный на каждой из этих нод, создать контейнер с выделенными ресурсами и в нем подготовить задачу map. Количество задач map будет равно количеству входных блоков (исключая реплицированные блоки).

4. MapReduce подготавливает записи split и запускает обработку каждой записи путем вызова метода `map` в соответствующем контейнере. Процесс выполнения задачи посылает периодически отчет в Application Master. Если происходит сбой в процессе выполнения задачи, Application Master запускает задачу заново.

5. Задачи map сохраняют полученные на выходе данные (output) в локальной файловой системе. HDFS не используется, чтобы не создавать излишнюю нагрузку на кластер для хранения промежуточных данных.

6. После завершения выполнения задач map Application Master организует контейнеры для задач reduce.

7. Application Master запрашивает сервис shuffle на тех нодах, где сохранены полученные на выходе данные (output) задач map, выполнить перераспределение этих данных по нодам, где созданы задачи reduce.

8. Задачи reduce агрегируют данные, используя код приложения, и сохраняют полученные на выходе данные (output) в HDFS. Каждая задача reduce сохраняет данные в своем файле HDFS.

9. Application Master завершает выполнение задания.

В разделе Начало работы с MapReduce приведена дополнительная информация.

Обзор работы MapReduce

Альберт Багдасарян

Содержание

Назад к содержанию

Map и Reduce

Сервис MapReduce оперирует данными, состоящими из пар ключ/значение (key/value). Независимо от того, являются входные данные структурированными или неструктурированными, сервис преобразует их в подобные пары:

- Ключ ссылается на значение.
- Значение представляет собой данные для обработки.

Map и Reduce

Функция Map вырабатывает список пар ключ/значение со следующими характеристиками:

- Выходные данные функции Map являются промежуточными.
- Их тип может отличаться от типа входных данных.
- Выходные данные функции сохраняются на локальном диске с последующим их перераспределением (shuffle) по узлам, выполняющим операцию reduce, то есть агрегацию данных.

Функция Reduce принимает промежуточные пары ключ/значение от функции Map и обрабатывает их. Обычно узлы (reducer), где работает эта функция, выполняют агрегацию или обобщение данных:

- Входными данными для агрегатора являются данные, сгенерированные функцией Map.
- Входные пары ключ/значение отсортированы по ключу.

Функция Reduce вырабатывает окончательный список пар ключ/значение. Он сохраняется в HDFS, при этом типы ключей и значений могут отличаться от их типов во входных данных.

Входные данные, подаваемые на вход преобразователя mapper, обрабатывает функция, заданная разработчиком. Основная нагрузка по реализации требуемой бизнес-логики ложится в основном на уровень Map, поэтому параллельно работающие узлы mapper наиболее нагружены. По этой причине количество обработчиков mapper значительно превышает количество агрегаторов reducer. Обработчики mapper генерируют промежуточные данные, подаваемые на вход агрегаторов reducer.

Промежуточный результат обрабатывает пользовательская функция, работающая как агрегатор данных. Она генерирует окончательные данные, сохраняемые в HDFS. Репликация данных выполняется как обычно.

Начало работы с MapReduce демонстрирует процесс преобразования и агрегирования.

Архитектура мониторинга в ADH

Константин Алпашкин

Содержание

Назад к содержанию

- Обзор
- Архитектура
  - Prometheus
  - Grafana
  - Node Exporter
  - Сервисные экспортеры
  - Принцип работы
- Рекомендации по использованию

Обзор

Monitoring — это сервис ADH, предназначенный для сбора, хранения и визуализации метрик кластера и отдельных ADH-сервисов. Данный сервис предоставляет единую панель управления для мониторинга работоспособности, производительности и потребления ресурсов узлами и компонентами кластера.

Сервис мониторинга построен на популярном стеке с открытым исходным кодом:

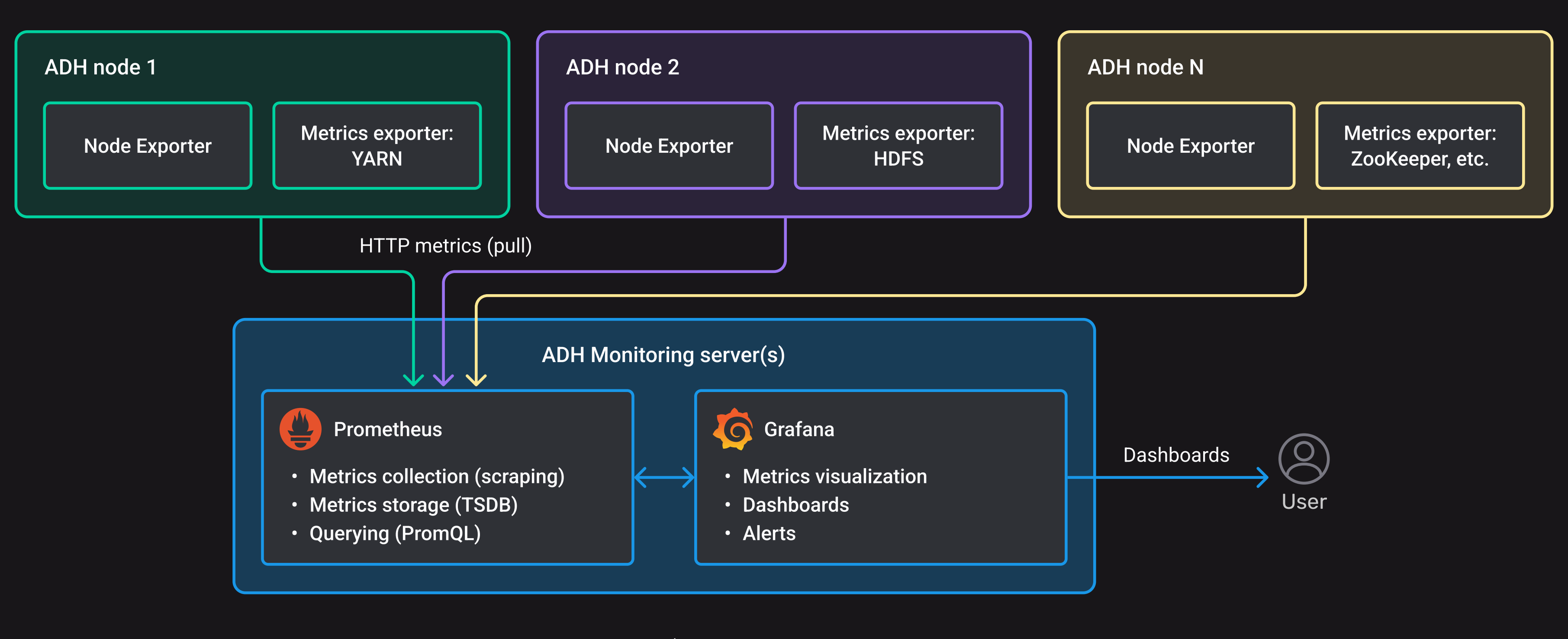
- Prometheus. Система мониторинга и оповещения, которая собирает и хранит метрики из приложений и инфраструктуры.
- Grafana. Платформа визуализации и мониторинга, которая позволяет визуализировать метрики, полученные из различных источников данных.
- Экспортеры метрик (metric exporter). Легковесные агенты для сбора и передачи в Prometheus метрик операционной системы и сервисных метрик.

Основные принципы мониторинга в ADH:

- **Централизованное хранилище**. Prometheus выступает в качестве центрального узла для сбора и хранения метрик.
- **Децентрализованные источники данных**. Экспортеры метрик функционируют как часть ADH-сервисов (например, существуют отдельные экспортеры для HDFS, YARN, Zookeeper и других). Сервер мониторинга получает метрики от экспортеров.
- **Готовые дашборды**. Сервис включает преднастроенные дашборды Grafana для ключевых ADH-сервисов.
- **Масштабируемость**. Гибкая архитектура позволяет добавлять новые метрики и самостоятельно создавать дашборды без необходимости ожидания нового релиза.

Архитектура

Архитектура сервиса мониторинга представляет собой классическую реализацию стека Grafana/Prometheus, который включает в себя следующие компоненты, развернутые на одном или нескольких хостах ADH-кластера.



Архитектура мониторинга

ПРИМЕЧАНИЕ

Установка компонентов мониторинга на хостах ADH вместе с другими ADH-сервисами считается допустимой, однако в производственной среде рекомендуется выделять отдельный хост для компонентов мониторинга.

Основные компоненты сервиса мониторинга описаны ниже.

Prometheus

Prometheus — это система мониторинга и уведомлений с открытым исходным кодом, является ключевым компонентом сервиса. Отвечает за сбор, хранение и обработку метрик.

Основные характеристики:

- **Мультидисциплинарная модель данных**. Метрики идентифицируются по имени и набору пар ключ/значение (label).
- **Модель HTTP pull**. Prometheus собирает метрики из целевых эндпойнтов по HTTP.
- **Мощный язык запросов PromQL**. Позволяет выполнять сложные агрегации и анализ метрик.
- **Конфигурация в ADH**. Дистрибутив Prometheus, входящий в состав сервиса мониторинга, преднастроен на сбор метрик от основных ADH-сервисов, а также метрик операционной системы от компонентов Node Exporter.

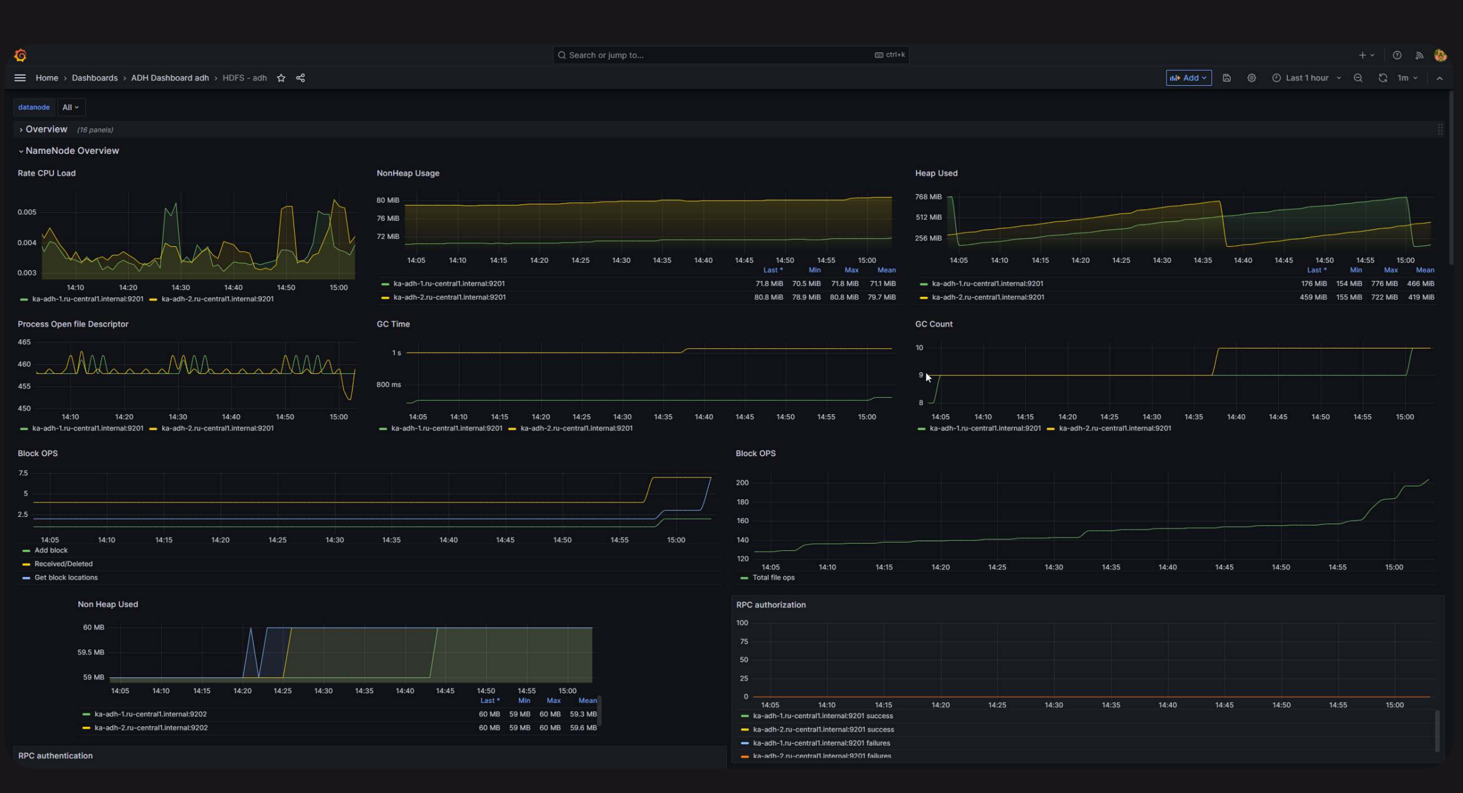
Grafana

Grafana — это аналитическая и мониторинговая платформа с открытым исходным кодом, предназначенная для визуализации данных. Grafana визуализирует метрики, хранящиеся в Prometheus, с помощью информативных дашбордов.

Основные характеристики:

- **Поддержка нескольких источников данных**. Grafana может работать не только с Prometheus, но и с другими источниками метрик.
- **Богатые возможности визуализации**. Предоставляет готовые графики, диаграммы, таблицы, тепловые карты и так далее.
- **Интерактивные дашборды**. Позволяет создавать динамические дашборды с переменными и переключателями (switch).
- **Гибкие настройки**. Готовые дашборды можно адаптировать под конкретные цели, а также создавать новые, соответствующие требованиям бизнеса.
- **Конфигурация в ADH**. Дистрибутив Grafana, поставляемый с ADH, содержит набор преднастроенных дашбордов для мониторинга ключевых ADH-сервисов (HDFS, YARN, Zookeeper и прочих), а также дашборды для отслеживания состояния хостов кластера.

Ниже представлен веб-интерфейс Grafana. Актуальная ссылка на web UI доступна в ADCM (***Services*** → ***Monitoring*** → ***Info***).



Веб-интерфейс Grafana

Node Exporter

Node Exporter — официальный экспортер метрик Prometheus для сбора низкоуровневых аппаратных метрик и метрик операционной системы. Это небольшие агенты, которые запускаются на каждом хосте ADH-кластера и собирают метрики для Prometheus.

Основные характеристики:

- **Сбор метрик ОС**. Node Exporter предоставляет метрики по использованию CPU, RAM, дисковому I/O, использованию дискового пространства, сетевому трафику и многим другим показателям.
- **Доступ по API**. Предоставляет метрики внешним системам через HTTP-эндпойнт.
- **Расширяемость**. Используя подключаемые коллекторы (collector), можно настроить Node Exporter для сбора специальных метрик.
- **Установка в ADH**. Компоненты Node Exporter работают как системные демоны на каждом узле ADH-кластера, обеспечивая единообразный сбор метрик с хостов.

Сервисные экспортеры

Сервисные экспортеры (service exporter) — это агенты, функционирующие как часть ADH-сервиса (HDFS, YARN, Hive и другие) и отвечающие за предоставление метрик Prometheus, специфичных для конкретного сервиса. Некоторые сервисы ADH имеют JMX-экспортеры, которые передают метрики напрямую из JVM сервиса, используя JMX. Другие сервисы предоставляют метрики через фиксированный HTTP-эндпойнт.

В отличие от компонентов Node Exporter, сервисные экспортеры не являются частью Monitoring-сервиса, однако являются критически важными элементами для его работы. Сервисные экспортеры автоматически устанавливаются и настраиваются при добавлении сервисов ADH в кластер. Например, при установке компонентов HDFS NameNode/Datanode также устанавливается и настраивается экспортер HDFS.

ПРИМЕЧАНИЕ

При добавлении нового сервиса ADH необходимо перезапустить сервис мониторинга, чтобы получать метрики от нового сервиса.

Ниже приведен список преднастроенных сервисных экспортеров ADH.

Сервис ADH	Тип экспортера	Расположение настроек (ADCM)
HDFS	jmx-exporter	Раздел настроек ***Monitoring*** в сервисных компонентах DataNode, JournalNode и NameNode
Ozone	Предоставляет метрики в формате Prometheus через HTTP-эндпойнт	***Services*** → ***Ozone*** → ***Primary Configuration*** → ***ozone-site.xml*** → ***hdds.prometheus.endpoint.enabled***
Impala	Предоставляет метрики в формате Prometheus через HTTP-эндпойнт (*/metrics_prometheus*)	—
Trino	jmx-exporter	Раздел настроек ***Monitoring*** для каждого компонента сервиса
Hive	jmx-exporter	Раздел настроек ***Monitoring*** для сервисных компонентов Hive HiveServer2 и Hive Metastore
YARN	jmx-exporter	Раздел настроек ***Monitoring*** для сервисных компонентов Node Manager и Resource Manager
HUE	Предоставляет метрики в формате Prometheus через HTTP-эндпойнт	***Services*** → ***HUE*** → ***Components*** → ***HUE Server*** → ***Primary Configuration*** → ***hue.ini*** → ***desktop.enable_prometheus***
Zeppelin	Предоставляет метрики в формате Prometheus через HTTP-эндпойнт	***Services*** → ***Zeppelin*** → ***Primary Configuration*** → ***zeppelin-site.xml*** → ***zeppelin.metric.enable.prometheus***
HBase	jmx-exporter	Раздел настроек ***Monitoring*** для каждого компонента сервиса
Solr	solr-exporter	Конфигурации экспортеров находятся в настройках компонента ***Solr prometheus exporter***
Kyuubi	Предоставляет метрики в формате Prometheus через HTTP-эндпойнт	В разделе настроек ***kyuubi-defaults.conf*** (***Services*** → ***Kyuubi*** → ***Components*** → ***Kyuubi Server***) доступны следующие параметры: ▪ ***kyuubi.metrics.enabled*** ▪ ***kyuubi.metrics.reporters*** ▪ ***kyuubi.metrics.prometheus.port*** ▪ ***kyuubi.metrics.prometheus.path***
Zookeeper	Предоставляет метрики в формате Prometheus через HTTP-эндпойнт	В разделе настроек ***zoo.cfg*** (***Services*** → ***Zookeeper*** → ***Primary configuration***) доступны следующие параметры: ▪ ***metricsProvider.className*** ▪ ***metricsProvider.httpPort***

Принцип работы

1. Компоненты Node Exporter работают на каждом физическом или виртуальном хосте ADH-кластера и собирают метрики ОС (CPU, RAM, дисковые показатели, сетевой ввод-вывод).
2. На каждом хосте кластера, где развернуты ADH-сервисы, запускаются соответствующие сервисные экспортеры (например, JMX-экспортер для компонента HDFS DataNode).
3. Prometheus, который работает на выделенном хосте, периодически опрашивает (scrape) все экспортеры метрик, отправляя HTTP-запросы на их эндпойнты (например, *http://<ADH_host_name>>:9100/metrics* для Node Exporter).
4. Prometheus получает метрики в текстовом формате, шифрует их и сохраняет в базе данных на локальном диске.
5. Grafana подключается к API Prometheus для получения и визуализации данных метрик.
6. Метрики отображаются в веб-интерфейсе Grafana в виде дашбордов.

Рекомендации по использованию

Для лучшей стабильности и более высокой производительности сервиса мониторинга учитывайте следующие рекомендации:

- **Разделение ролей**. Для повышения отказоустойчивости и улучшения производительности в больших кластерах рекомендуется устанавливать сервис мониторинга на выделенных узлах ADH, на которых не размещены основные вычислительные сервисы.
- **Настройка оповещений**. Используйте встроенный Prometheus Alertmanager для настройки оповещений. Вы можете интегрировать оповещения с популярными каналами уведомлений, такими как электронная почта, Slack и другими, чтобы быстро реагировать на инциденты. Данные компоненты не входят в дистрибутив ADH и требуют отдельной настройки.
- **Хранение данных мониторинга**. Отслеживайте объем дискового пространства, выделенного под данные Prometheus. Настройте политики хранения в соответствии с вашими требованиями к размеру хранимых метрик.
- **Кастомизация дашбордов**. Используя в качестве шаблонов преднастроенные дашборды Grafana, входящие в состав ADH, вы можете создавать свои дашборды для отслеживания специфичных показателей, имеющих критическое значение для ваших рабочих нагрузок и бизнес-процессов.
- **Следите за состоянием сервиса мониторинга**. Отслеживайте работоспособность и производительность сервиса мониторинга. Для этого можно использовать метрики, которые Prometheus собирает о себе.
- **Настройка сервисов**. При изменении конфигурации сервисных экспортеров (добавление новых сервисов в ADH-кластер, изменение топологии сервисов) необходимо перезапустить сервис мониторинга, чтобы применить новую конфигурацию.

Архитектура Ozone

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
- Концепции
- Компоненты
- Операции с данными
  - Операция записи
  - Операция чтения

Обзор

Apache Ozone (O3) — это распределенное хранилище объектов, использующее формат ключ/значение, оптимизированное для работы как с сервисами Hadoop, так и с хранилищами S3. Ozone обладает высокой масштабируемостью, может использовать те же политики безопасности кластера, что и HDFS, и способен эффективно обрабатывать файлы любого размера.

Основные преимущества Ozone:

- В отличие от файловой системы HDFS, Ozone разработан для эффективной работы с большим количеством небольших файлов.
- Хорошо интегрируется с сервисами Hadoop, такими как Hive, Spark, MapReduce, Kerberos и фреймворком Apache Ranger.
- Поддерживает работу с удаленными объектными хранилищами через API Amazon S3.
- Может работать с контейнерными средами, такими как Kubernetes и YARN.

Концепции

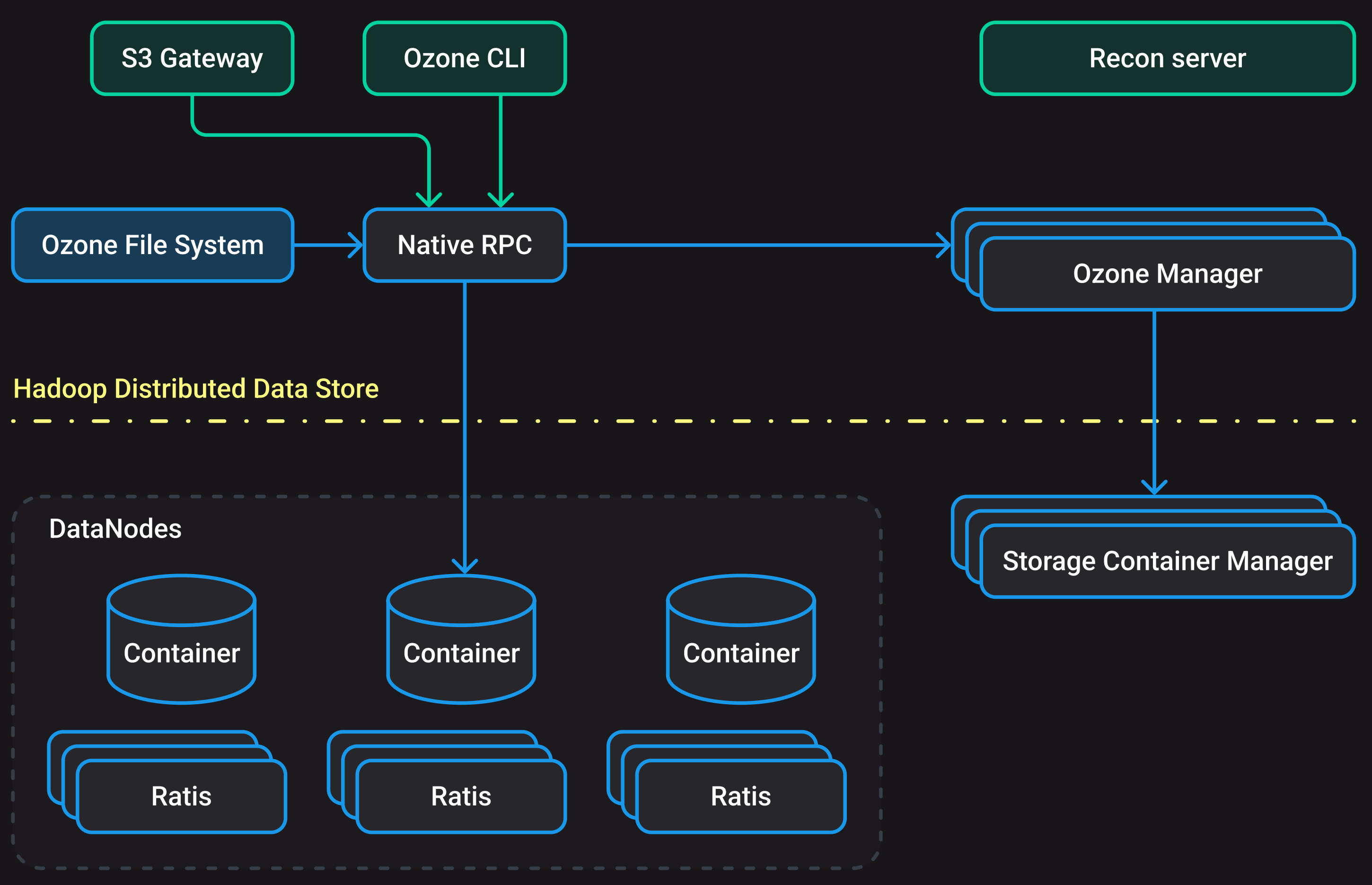
В Ozone существуют следующие основные элементы:

- **Блоки**
  Как и в HDFS, блоки являются базовой единицей хранения. Каждый файл в системе состоит из блоков. Стандартный размер блока в Ozone составляет 256 МБ.
- **Контейнеры**
  Контейнер — это группа блоков, которая реплицируется как единое целое.
- **Ключи**
  Ключ — это маркер, который указывает, какие блоки принадлежат определенному файлу. Ключи используются как идентификаторы файлов для всех операций с данными, выполняемых клиентами.
- **Бакеты (buckets)**
  Бакеты — это директории в терминах Ozone. Они могут содержать любое количество ключей, но не могут содержать другие бакеты.
- **Том (volume)**
  Том — это домашний каталог пользователя в Ozone, который может быть создан только администратором. Пространство имен Ozone может содержать любое количество томов. После создания тома пользователь может создавать контейнеры, принадлежащие этому тому.

Компоненты

Архитектура Ozone основана на следующих компонентах:

- **Ozone Manager**
  Системная служба, которая управляет пространством имен таким же образом, как HDFS NameNode. Она регистрирует блоки, расположенные на узлах DataNodes, и предоставляет идентификаторы блоков по запросу.
- **Ozone Storage Container Manager**
  Системная служба, которая управляет контейнерами. Она выделяет блоки и назначает их узлам DataNodes. Клиенты считывают и записывают эти блоки напрямую. Container Manager отслеживает все реплики блоков. Если происходит потеря данных или диска, он обнаруживает это и дает команду сделать копии недостающих блоков для обеспечения высокой доступности.
- **Ozone DataNode**
  Служба, которая объединяет блоки в контейнеры.
- **Ozone Recon**
  Сервер Ozone Recon взаимодействует со всеми другими компонентами Ozone и предоставляет единый API управления и пользовательский интерфейс.



Архитектура Ozone

Операции с данными

В Ozone есть три основных функциональных уровня:

- Уровень метаданных, управляемый Ozone Manager и Storage Container Manager.
- Уровень хранения данных, управляемый Ozone DataNodes.
- Уровень репликации, управляемый Apache Ratis и используемый для репликации метаданных.

Ниже приведено описание того, как эти уровни взаимодействуют во время операций чтения и записи.

Операция записи

В Ozone операция записи проходит следующие этапы:

1. Чтобы записать файл (ключ) в Ozone, клиент делает запрос в Ozone Manager. Запрос содержит информацию о бакете и томе, где будет создан ключ.
2. Чтобы выделить блоки для нового ключа, Ozone Manager отправляет запрос в Storage Container Manager.
3. Storage Container Manager выделяет блоки на узлах DataNode и возвращает их идентификаторы Ozone Manager.
4. Ozone Manager записывает метаданные и возвращает токен блока (разрешение на запись данных в блок) клиенту.
5. Клиент использует токен блока и записывает данные в DataNode.

Операция чтения

В Ozone операция чтения проходит следующие этапы:

1. Клиент запрашивает список блоков, принадлежащих ключу, у Ozone Manager.
2. Ozone Manager возвращает список блоков и токены блоков.
3. Клиент подключается к DataNode и считывает данные.

Сравнение HDFS и Ozone

Сергей Тихомиров

Содержание

Назад к содержанию

Сравнение функций

Плюсы и минусы

- HDFS
- Ozone

Области применения

Как HDFS, так и Ozone (O3) являются подходящими распределенными хранилищами данных с открытым исходным кодом для кластера Hyperwave, но они имеют ряд различий, которые рассматриваются в данной статье.

Сравнение функций

В таблице ниже представлено сравнение ключевых функций сервисов.

Функция	HDFS	Ozone
Модель данных	Файловое хранилище с иерархической структурой	Хранилище объектов, работающее с неструктурированными данными и оптимизированное для облака
Репликация данных	Репликация данных между узлами DataNode для предоставления отказоустойчивости	Программно определяемое хранилище, которое позволяет настраивать политики репликации данных и обеспечивать избыточность данных
Масштабируемость	Хорошая масштабируемость для больших задач по обработке данных	Разработан с целью предоставить еще более хорошую масштабируемость, чем в HDFS
Управление пространством имен	Единое пространство имен для всего кластера	Несколько пространств имен для различных задач
Хранилище объектов	Нет	Да
Поддержка S3 и других протоколов хранилищ объектов	Нет	Да
Управление доступом	Права в стиле POSIX	Права в стиле S3 и управление доступом на уровне бакета
Аутентификация и авторизация	Kerberos	Kerberos, Ozone Token
Согласованность данных	Согласованность рано или поздно достигается	Сильная согласованность за счет таких протоколов, как RAFT

Плюсы и минусы

HDFS

HDFS является файловой системой по умолчанию в Hadoop, и она имеет следующие преимущества:

- поддержка хранения большого количества данных;
- быстрое определение и реагирование на аппаратные сбои;
- поддержка потоковой передачи данных;
- упрощенная модель согласованности;
- высокая отказоустойчивость и легкость восстановления;
- предназначена для коммерческого оборудования.

Однако у нее есть и некоторые недостатки:

- не подходит для большого числа маленьких файлов;
- не поддерживает изменение файлов (HDFS 2.x поддерживает добавление данных в файлы);
- проблемы при хранении более 400 миллионов файлов;
- не поддерживает параллельную запись.

Ozone

Недостатки HDFS, приводящие к дискомфорту при удовлетворении современных потребностей в хранении больших данных, потребовали реализации нового решения с рядом ключевых преимуществ:

- высокая согласованность данных;
- разработано для хранения более 100 миллиардов объектов в одном кластере;
- отличная масштабируемость благодаря многослойной архитектуре;
- такая же отказоустойчивость и легкость восстановления, как у HDFS;
- может работать рядом с HDFS на одних и тех же хостах.

Так как проект зародился не так давно, есть и некоторые недостатки:

- мало примеров внедрения;
- разработан для интеграции с экосистемой Hadoop, однако пока не имеет широкого уровня поддержки, и для работы с Ozone некоторые сервисы могут потребовать дополнительной настройки;
- отсутствие локального сокета, общая производительность ниже.

Области применения

Apache Ozone имеет преимущество перед HDFS в средах, где требуется масштабируемость для небольших файлов, совместимость с S3 или работа с облаком. Однако HDFS остается подходящим для рабочих нагрузок Hadoop с меньшими требованиями к хранению небольших объектов без возможности их объединения или облачной интеграции.

Обзор Phoenix

Владимир Адаменков

Содержание

Назад к содержанию

- Введение
- Возможности
- Архитектура
- Phoenix в ADH
- Взаимодействие с сервисами
  - HBase
  - ZooKeeper
  - MapReduce
  - Spark
  - Hive
- Детали реализации и ключевые операции
  - Управление индексами
  - Формат хранения данных
  - Salting
  - Операция UPSERT
- Примеры использования
- Преимущества
- Ограничения и особенности

Введение

Apache Phoenix — это реляционный движок базы данных, позволяющий выполнять SQL-запросы к данным, хранящимся в HBase.

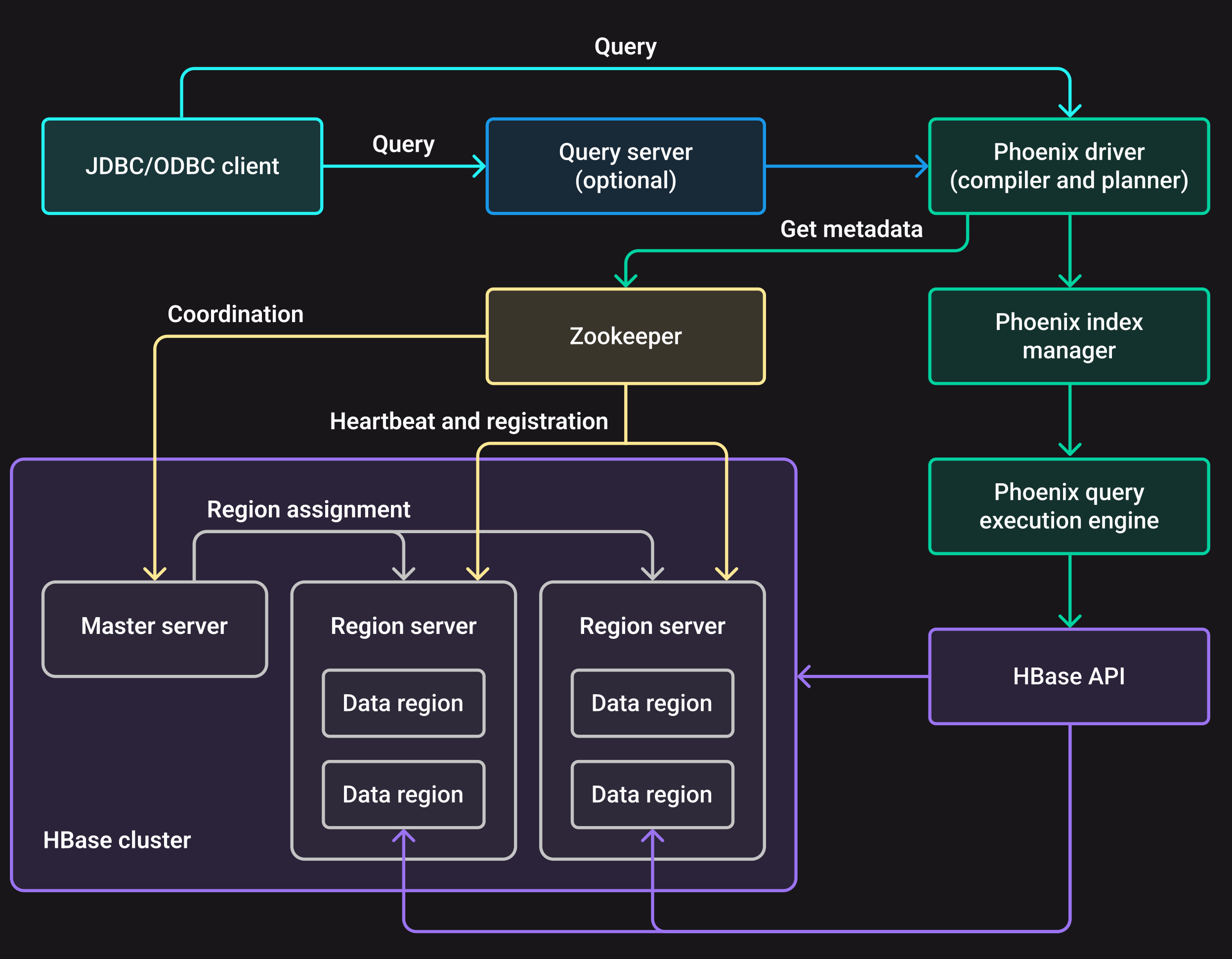
Возможности

Основные возможности Phoenix:

- Низкая задержка (low latency) и высокая производительность, достигаемые за счет параллельного исполнения, оптимизированного выполнения запросов и вторичной индексации.
- Массовая загрузка данных (например, с помощью MapReduce).
- Управление схемами: схемы данных и метаданные хранятся в централизованном каталоге, а столбцы таблиц можно добавлять и удалять.
- Мультиарендность (multi-tenancy): несколько пользователей или приложений могут совместно использовать один кластер HBase, сохраняя при этом изоляцию и безопасность данных.
- Использование salted-таблиц.
- Поддержка атомарных операций и транзакционного поведения для обеспечения согласованности данных.
- Бесшовная интеграция с Hadoop и Spark.

Архитектура

Ниже показана общая архитектура Phoenix.



Архитектурная схема Phoenix

В этой схеме данные передаются следующим образом:

1. SQL-запрос отправляется через JDBC/ODBC-клиент.
2. Сервер запросов (например, Apache Calcite Avatica), если он используется, выступает в качестве посредника между клиентским приложением и ядром Phoenix. Он выполняет следующие функции:
   - Прослушивает сетевой интерфейс в ожидании входящих SQL-запросов от клиентских приложений по сети через RESTful API.
   - При необходимости управляет доступом к Phoenix с помощью механизмов аутентификации и авторизации.
   - Управляет подключениями к кластеру Phoenix/HBase путем их объединения (connection pooling). Это помогает сократить нагрузку на сеть, вызываемую установлением новых подключений для каждого запроса.
   - Пересылает SQL-запросы в драйвер Phoenix.
3. Запрос отправляется в драйвер Phoenix, который передает метаданные в ZooKeeper. Компилятор и планировщик в драйвере Phoenix анализируют запрос, проверяют его, затем оптимизируют и создают план выполнения.
4. Если возможно, менеджер индексов Phoenix определяет, можно ли использовать какие-либо индексы для дальнейшей оптимизации запроса.
5. Механизм выполнения запросов выполняет план, взаимодействуя с HBase через API.
6. HBase извлекает данные из соответствующих регионов.
7. Phoenix обрабатывает данные (фильтрует, агрегирует и т.д.) и возвращает результаты клиенту тем же путем.

Phoenix в ADH

Как уже было сказано, Phoenix построен на основе HBase. Это SQL-обертка, которая использует сильные стороны HBase: масштабируемость, отказоустойчивость, хранение данных NoSQL. В то время как Phoenix фокусируется на обработке запросов, исходные данные находятся в HBase, который хранит их в HDFS. Phoenix использует HBase для доступа к этим данным.

Phoenix можно использовать совместно с другими компонентами экосистемы Hadoop, такими как Spark и Hive. Такая интеграция позволяет использовать эти инструменты для более сложной обработки и аналитики данных.

Взаимодействие с сервисами

Phoenix взаимодействует с различными сервисами в ADH.

HBase

Наиболее важным является взаимодействие с HBase. Phoenix широко использует API клиента HBase для всех операций чтения и записи. Он транслирует SQL-запросы в операции HBase, такие как `get`, `scan`, `put`, `delete` и другие.

ZooKeeper

Phoenix использует ZooKeeper для следующих целей:

- **Обнаружение кластера HBase**. Клиенты Phoenix и сервер запросов (при наличии) используют ZooKeeper для обнаружения местоположения master-сервера и серверов регионов HBase.
- **Выборы лидера**. В некоторых сценариях ZooKeeper используется для выбора лидера среди экземпляров Phoenix.
- **Координация метаданных**. Управление распределенными метаданными о таблицах, индексах и схемах Phoenix.

MapReduce

Phoenix может взаимодействовать с MapReduce для следующих целей:

- **Массовая загрузка**. Phoenix предоставляет механизм для массовой загрузки данных в HBase с помощью MapReduce. Это эффективнее, чем вставка строк по одной с помощью операторов SQL `INSERT` или `UPSERT`.
- **Пользовательская обработка**. Можно создавать пользовательские задания MapReduce, которые считывают данные из таблиц Phoenix, выполняют преобразования и записывают результаты обратно в Phoenix или другое хранилище данных.

Spark

Phoenix может взаимодействовать со Spark для следующих целей:

- **Spark-коннектор для Phoenix**. Этот коннектор позволяет приложениям Spark считывать и записывать данные в таблицы Phoenix. Spark можно использовать для преобразований данных и анализа, которые сложно или неэффективно выполнять напрямую в Phoenix.
- **Spark SQL**. Можно использовать Spark SQL для запросов к данным в таблицах Phoenix, используя механизм распределенных запросов Spark.

Hive

Phoenix может взаимодействовать с Hive косвенным образом. Данные, загруженные в HBase с помощью Phoenix, могут быть доступны Hive с помощью обработчиков хранилища, но это практикуется реже, чем использование Spark.

Детали реализации и ключевые операции

Управление индексами

При создании индекса Phoenix создает отдельную таблицу HBase для хранения данных индекса. Запись данных в исходную таблицу автоматически запускает обновление соответствующих индексов (для изменяемых индексов). Во время выполнения запроса механизм запросов анализирует его и определяет, можно ли использовать индекс для ускорения выполнения запроса.

Формат хранения данных

Phoenix хранит данные в HBase в формате, основанном на строках. Ключ строки обычно состоит из одного или нескольких столбцов из таблицы Phoenix. Семейства столбцов используются для группировки связанных столбцов.

Salting

Для предотвращения избыточной нагрузки на один сервер регионов при простое остальных используется механизм под названием salting.

Операция UPSERT

Оператор `UPSERT` в Phoenix используется для вставки новой строки в таблицу или обновления существующей строки. Если строка с указанным первичным ключом уже существует, оператор `UPSERT` обновит существующую строку. Если строка не существует, оператор `UPSERT` вставит новую строку.

Примеры использования

Ниже приведены примеры реальных сценариев, в которых использование Phoenix особенно выгодно:

- **Операционная аналитика**. Phoenix отлично справляется с анализом рабочих данных, хранящихся в HBase, в реальном времени или почти в реальном времени. Это отличает его от традиционных пакетно-ориентированных хранилищ данных. Примеры:
  - Мониторинг в реальном времени — мониторинг производительности систем, активности приложений или сетевого трафика.
  - Обнаружение мошенничества — выявление мошеннических транзакций в реальном времени.
  - Персонализация — предоставление персонализированных рекомендаций на основе поведения пользователя.
- **Анализ данных временных рядов**. HBase часто используется для хранения данных временных рядов (например, показаний датчиков, цен акций, событий журналов). Phoenix можно использовать для эффективных запросов к этим данным и последующего анализа.
- **Интернет вещей (Internet of Things, IoT)**. Phoenix хорошо подходит для обработки данных с устройств IoT. Обычно эти данные имеют большой объем и требуют низкую задержку передачи и обработки.
- **Анализ кликстрима**. Phoenix можно использовать для анализа данных кликстрима на веб-сайтах для понимания поведения пользователя и улучшения дизайна веб-сайтов.
- **Телекоммуникации**. Phoenix можно использовать для мониторинга сети, анализа записей о вызовах (Call Detail Record, CDR).
- **Хранилище данных**. Хотя Phoenix не является заменой полноценного хранилища данных, его можно использовать для сценариев хранения данных меньшего масштаба, где важна производительность запросов в реальном времени.
- **Обслуживающий уровень для машинного обучения**. Phoenix может выступать в качестве обслуживающего уровня, обеспечивая доступ в реальном времени к функциям для моделей машинного обучения.

Преимущества

Использование Phoenix может дать следующие преимущества:

- **Известность SQL**. Самым большим преимуществом Phoenix является использование SQL. Это делает HBase доступным для гораздо большей аудитории, поскольку SQL — широко известный и понятный язык.
- **Производительность**. Phoenix обеспечивает оптимизированное выполнение запросов к HBase, часто достигая гораздо более высокой OLTP-производительности, чем другие решения SQL-on-Hadoop.
- **Низкая задержка**. Phoenix разработан для запросов с низкой задержкой, что делает его подходящим для использования в приложениях в реальном времени или почти в реальном времени.
- **Масштабируемость и отказоустойчивость**. Phoenix эффективно использует масштабируемость и отказоустойчивость, которыми обладает HBase.
- **Вторичные индексы**. Phoenix обеспечивает более быстрые запросы с помощью вторичных индексов.
- **Поддержка транзакций**. Phoenix обеспечивает согласованность и надежность данных с помощью транзакций ACID.
- **Интеграция с экосистемой Hadoop**. Phoenix легко интегрируется с другими компонентами Hadoop, такими как Spark, MapReduce и Hive.
- **Сокращение усилий на разработку**. Phoenix упрощает разработку, предоставляя интерфейс SQL для HBase, что снижает необходимость в сложном программировании API HBase.

Ограничения и особенности

Phoenix имеет следующие ограничения и особенности:

- Phoenix не поддерживает стандарт ANSI SQL в полном объеме. Подробности см. в документации Apache Phoenix.
- Управление вторичными индексами может быть сложным, особенно для больших таблиц. Необходимо тщательно продумать, какие столбцы индексировать и как поддерживать индексы.
- NoSQL-устройство HBase влияет на то, как данные моделируются в Phoenix. Методы моделирования реляционных баз данных не всегда могут быть применимы напрямую.
- Если ключ строки не сформирован должным образом, записи могут быть сосредоточены на одном сервере регионов, что приводит к перегрузкам и снижению производительности. Использование salting может устранить этот эффект.

Обзор работы Solr

Константин Алпашкин

Содержание

Назад к содержанию

Индексация

Запрос

Ранжирование результатов поиска

Solr — это поисковая платформа для работы с большими объемами данных. Так как Solr может хранить данные, Solr является и нереляционным NoSQL-хранилищем, и платформой для обработки данных.

При отправке поискового запроса к серверу, Solr дробит запросы на более мелкие, а затем находит совпадения по запросу, используя инвертированный индекс документа, созданный ранее. В качестве ответа на запрос, поисковый сервер Solr возвращает набор документов, основываясь на классовой схожести и других критериях, указанных в файлах конфигурации *schema.xml* и *solr.config*.

Три основных процесса в работе Solr — это индексация, запрос и ранжирование.

Индексация

Solr использует различные способы для индексации документов и rich-text данных. Solr позволяет загружать документы в формате PDF, CSV, XML — система автоматически распознает и индексирует данные форматы. Также документы можно загружать из электронных писем и вложений к ним.

Для хранения данных Solr использует инвертированный индекс с keyword-ориентированной структурой (не page-ориентированной). Простейшая аналогия данного подхода — алфавитный указатель в конце любой книги, где указаны термин, страницы, где данный термин встречается, и его толкование. Такой подход уменьшает время отклика и позволяет мгновенно возвращать релевантные результаты поиска.

Запрос

Поисковым запросом называется любой поиск по тексту, изображениям или данным геолокации. После отправки запроса к Solr запрос обрабатывается обработчиком запросов, который возвращает документ из Solr-индекса.

Ранжирование результатов поиска

При обработке запросов Solr находит совпадения с данными из индексированных файлов на основании ключевых слов (keywords) и осуществляет ранжирование результатов по релевантности. Результаты поиска имеют иерархичную структуру, основанную на релевантности.

Spark Connect

Константин Алпашкин

Содержание

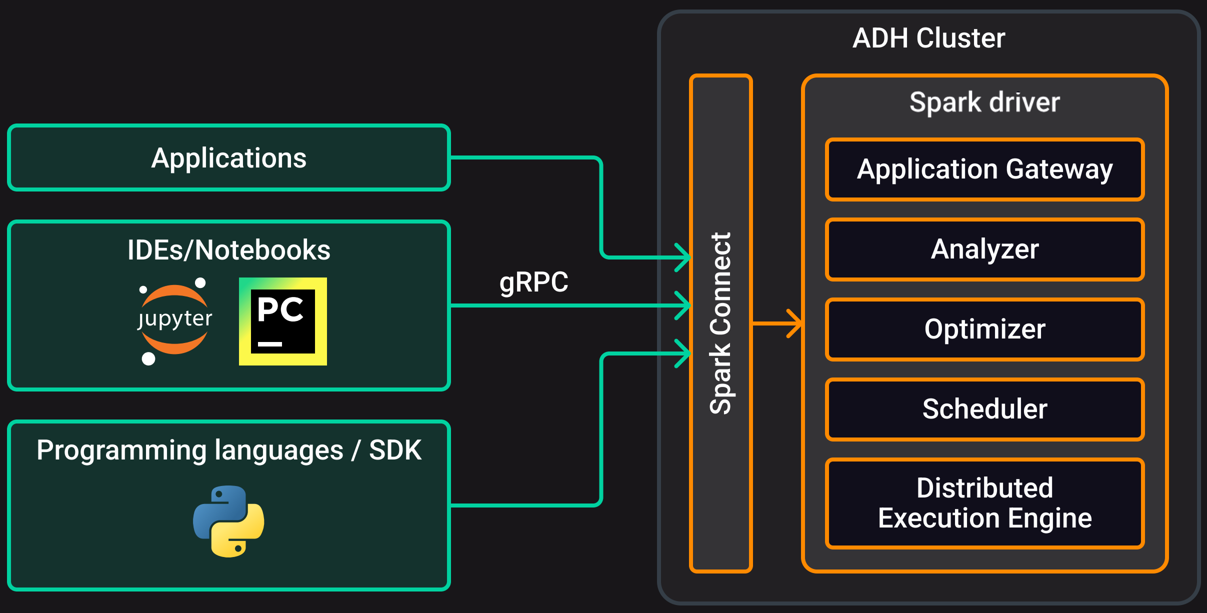
Назад к содержанию

Обзор
Как работает Spark Connect
Плюсы Spark Connect

Обзор

Spark Connect — это компонент сервиса Spark3, который, выполняя функции тонкого клиента, обеспечивает удаленное подключение к кластерам Spark. С помощью Spark Connect вы можете удаленно управлять кластером Spark, например, используя привычную IDE на обычном пользовательском ноутбуке.

На следующей диаграмме представлена высокоуровневая архитектура Spark Connect.



Архитектура Spark Connect

Как работает Spark Connect

Spark Connect реализует раздельную клиент-серверную архитектуру. Серверной частью является компонент ADH Spark Connect, который слушает команды от клиентов. Клиентом может выступать приложение Spark/Spark shell, которое задействует клиентскую библиотеку Spark Connect для общения с сервером.

Основной особенностью приложений Spark, где используется Spark Connect, является создание **удаленной** Spark-сессии в коде приложения. Экземпляр такой сессии устанавливает соединение с сервером Spark Connect и автоматически поддерживает коммуникацию с сервером, отправляя на сервер DataFrame-операции и получая результаты их выполнения. После создания удаленной Spark-сессии она используется так же, как и обычная сессия Spark.

Spark Connect использует DataFrame API и неразрешенные логические планы (unresolved logical plans) в качестве протокола для общения с кластером Spark. Клиентская библиотека Spark Connect транслирует операции над DataFrame в неразрешенные логические планы запросов, затем кодирует их и отправляет на сервер Spark Connect, используя gRPC. Сервер Spark Connect получает и преобразует неразрешенные логические планы в операторы логических планов Spark. Далее запускается стандартный процесс работы Spark, включая все оптимизации и возможности. Результаты каждого вызова передаются обратно клиенту через gRPC в виде пакетов строк, закодированных Apache Arrow.

Подробности использования Spark Connect доступны в статье Использование Spark Connect.

Плюсы Spark Connect

Среди многих плюсов использования Spark Connect можно выделить следующие:

- **Легковесная клиентская часть**. Вы можете работать с массивным кластером Spark, используя IDE/Jupyter notebook на обычном пользовательском ноутбуке. Основное требование — использование удаленной сессии Spark, которая устанавливает соединение с сервером Spark Connect с помощью клиентской библиотеки.
- **Улучшенная изоляция ресурсов**. При работе с кластером Spark через Spark Connect создаются изолированные песочницы (sandbox) на хостах ADH с заранее выделенными ресурсами. Spark executors, принадлежащие разным приложениям, работают в этих изолированных песочницах и не отбирают друг у друга ресурсы. Это позволяет избежать случаев массового потребления памяти, когда, например, интенсивные вычисления приложения Spark или неправильно указанные параметры запуска могут привести к нехватке памяти на узле.
- **Расширенные возможности обновления**. Драйвер Spark можно обновлять отдельно, независимо от клиентской части. Например, вы можете легко обновить драйвер Spark, чтобы подтянуть последние фиксы безопасности, не внося изменений в ваши клиентские приложения. Основное требование — gRPC-интерфейсы серверной и клиентской части Spark Connect должны оставаться совместимыми после обновления.

Архитектура SSM

Елена Костюченко

Содержание

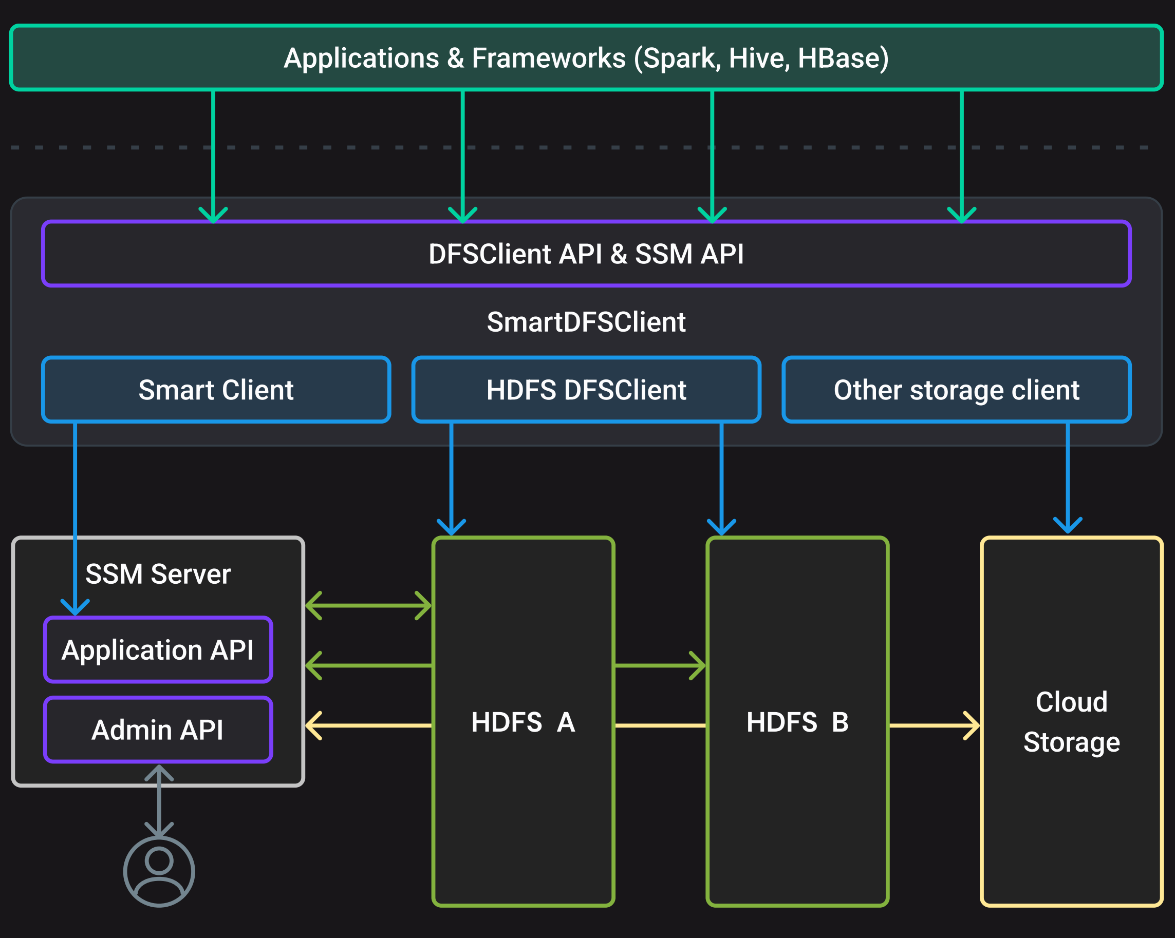
Назад к содержанию

- Обзор
- SSM Server
  - Правила
  - Действия
- SSM Agent
- SSM Client
- SSM Metastore

Обзор

SSM (Smart Storage Manager) — это инструмент для оптимизации хранения данных в HDFS, который анализирует метаданные и позволяет выбрать более эффективный способ хранения и управления данными в соответствии с созданными пользователем правилами и политиками.

SSM считает количество обращений к данным в кластере (access count) и позволяет классифицировать их в зависимости от их "температуры": популярные данные маркируются как горячие (HOT), редко используемые данные как холодные (COLD), а умеренно используемые данные как теплые (WARM). Точные критерии классификации данных определяются правилами, установленными пользователями.



Applications & Frameworks (Spark, Hive, HBase)
DFSClient API & SSM API
SmartDFSClient
Smart Client
HDFS DFSClient
Other storage client
SSM Server
Application API
Admin API
HDFS A
HDFS B
Cloud Storage

Архитектура SSM

SSM добавляет в HDFS следующие возможности:

- **Перемещение данных** — перемещение данных между различными типами хранилищ в зависимости от существующих правил.
- **Асинхронная репликация** — репликация данных между разными кластерами с HDFS или между HDFS и облачным хранилищем. SSM отслеживает операции изменения данных, такие как создание, удаление, добавление и переименование, чтобы обеспечить синхронизацию данных в реальном времени и избежать вычислительных затрат MapReduce.
- **Оптимизация хранения маленьких файлов** — сжатие небольших файлов в один файл-контейнер. Он хранится в HDFS и данные в нем доступны для приложений верхнего уровня.
- **Erasure coding (EC)** — гибкая настройка включения Erasure coding или репликации и управление файлами с разными политиками EC с помощью правил.
- **Сжатие** — сжатие данных в HDFS без ограничения доступа к ним для внешних приложений.

SSM предоставляет веб-интерфейс, с помощью которого можно создавать правила, запускать действия, проверять статус их выполнения и следить за статистикой кластера.

Вы можете управлять SSM и его компонентами с помощью ADCM или обновлять файлы конфигурации вручную. Дополнительно о параметрах конфигурации SSM читайте на странице Конфигурационные параметры.

SSM Server

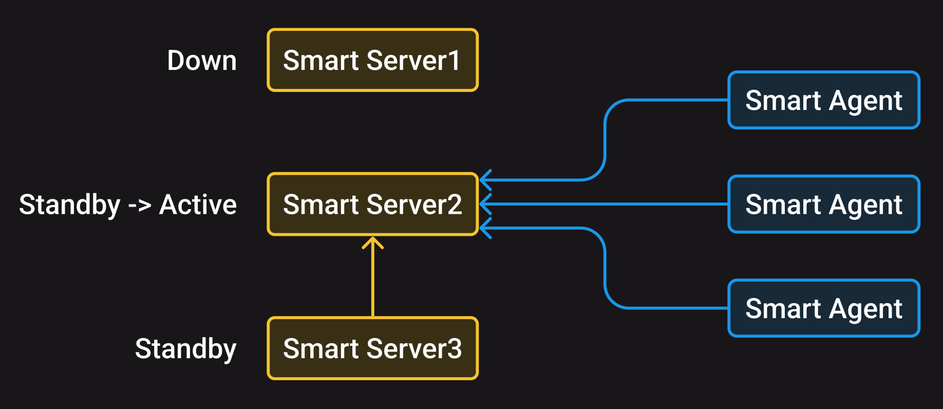
Сервер SSM (SSM Server, Smart Server) — основной компонент SSM. Он отвечает за сбор метаданных из NameNode, срабатывание правил и выполнение действий.

Сервер просматривает файл edits на NameNode, анализирует его и сохраняет в своей базе данных — **SSM Metastore**. Если информация, полученная из кластера, подходит под определенное правило, сервер запускает соответствующее действие.

Пользователь может взаимодействовать с сервером через API:

- API администратора
- API приложений

Для достижения высокой доступности (High Availability, HA) можно развернуть дополнительные SSM-серверы. В этом случае лишние серверы будут действовать как резервные (Standby).



Down
Smart Server1
Standby -> Active
Smart Server2
Standby
Smart Server3
Smart Agent
Smart Agent
Smart Agent

Сценарий аварийного переключения сервера SSM

Если активный сервер не работает, один из резервных серверов становится активным и все запросы переадресуются не него.

Правила

Правила (SSM rules) — это сценарии, которые пользователи могут создавать для автоматизации задач управления данными в HDFS. Правила написаны на собственном языке SSM (domain-specific language) и состоят из четырех частей: обект, триггер, условие и команда (действие).

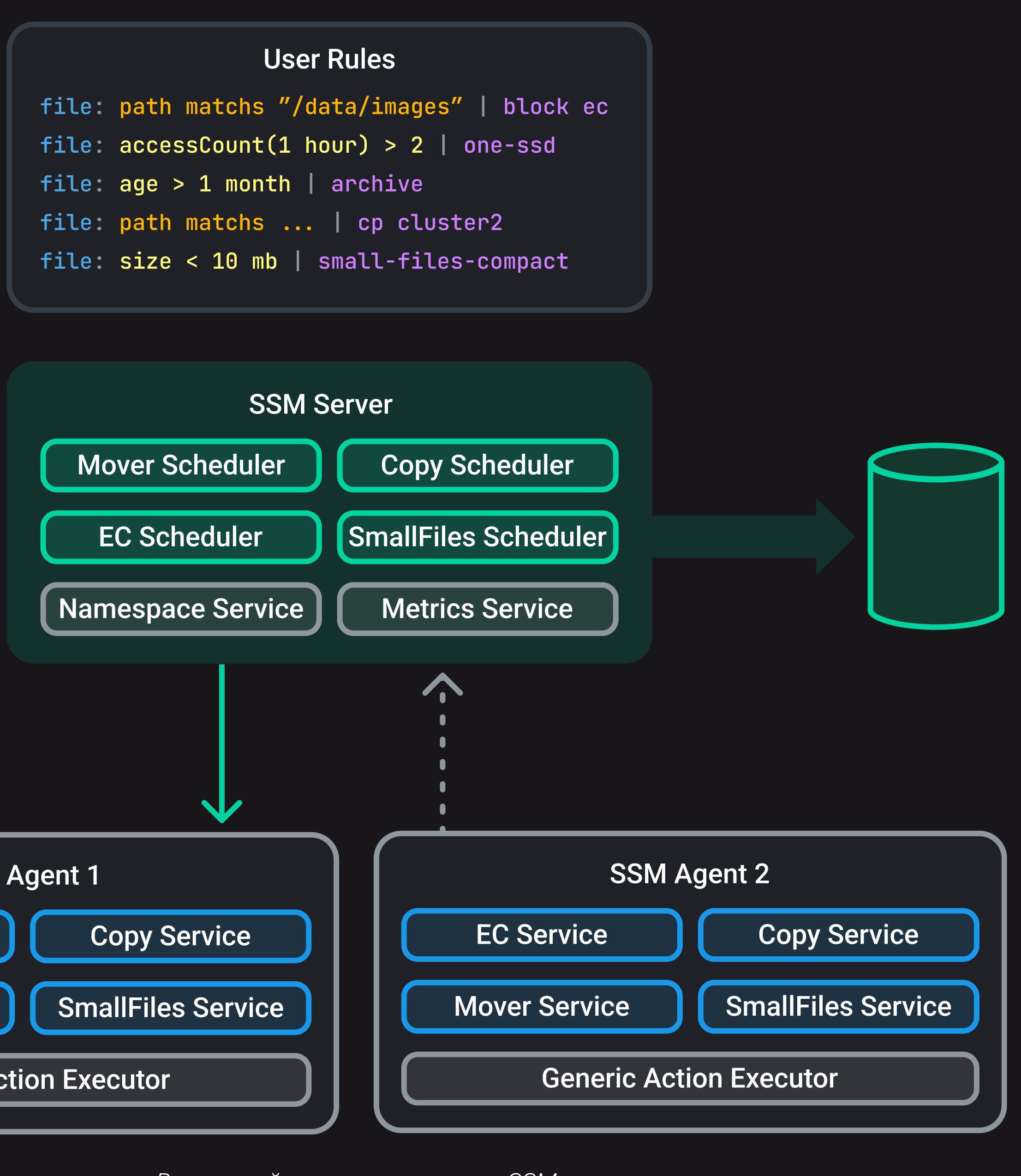
Например, если к определенным данным обращались более 3 раз за последние 10 минут, пользователь может классифицировать такие данные как "горячие" и создать правило, согласно которому SSM автоматически переместит их на более быстрый носитель.

Более подробно о том, как создавать правила для SSM читайте в статьях Определение правил в SSM и Примеры использования правил в SSM.

Действия

Действие (SSM action) — это одноразовая операция, выполняемая SSM при срабатывании правила или при ручном запуске пользователем. Каждое действие выполняется выделенным сервисом-планировщиком:

- Mover Scheduler — отвечает за операции, связанные с перемещением данных;
- Copy Scheduler — копирует данные;
- EC Scheduler — управляет файлами, для которых применен Erasure Coding;
- SmallFiles Scheduler — управляет контейнерами с маленькими файлами;
- NameSpace service — просматривает информацию из edits log на NameNode;
- Metrics service — собирает метрики (access count).



User Rules
SSM Server
Mover Scheduler
Copy Scheduler
EC Scheduler
SmallFiles Scheduler
Namespace Service
Metrics Service
SSM Agent 1
EC Service
Copy Service
Mover Service
SmallFiles Service
Generic Action Executor
SSM Agent 2
EC Service
Copy Service
Mover Service
SmallFiles Service
Generic Action Executor

```
file: path matchs "/data/images" | block ec
file: accessCount(1 hour) > 2 | one-ssd
file: age > 1 month | archive
file: path matchs ... | cp cluster2
file: size < 10 mb | small-files-compact
```

Взаимодействие компонентов SSM

Более подробно о действиях в SSM читайте в статье Определение действий в SSM.

SSM Agent

SSM Agent (Smart Agent) — это специальный процесс для выполнения задач, отправляемых сервером SSM. Чтобы расширить возможности обработки задач SSM, в кластере может быть запущено несколько агентов.

SSM Client

SSM Client (SmartDFSClient, Smart Client) — это обертка над клиентом HDFS, которая собирает данные для сервера SSM и предоставляет приложениям возможность читать сжатые файлы.

Когда приложение использует клиент для доступа к файлам в HDFS, SmartDFSClient сообщает об этом SSM Server. Событие регистрируется на сервере и он обновляет счетчик доступа к соответствующим данным в SSM Metastore.

SSM Metastore

SSM Metastore — это SQL-база данных, в которой хранится информация о количестве обращений к файлам, пользовательские правила и метрики.

Архитектура Trino

Елена Костюченко

Содержание

Назад к содержанию

- Обзор
- Концепции
- Компоненты
- Коннекторы

Обзор

Trino — это движок SQL-запросов с открытым исходным кодом, используемый для параллельной обработки данных, распределенных по хранилищам разного типа, таких как объектные хранилища (S3), базы данных и файловые системы.

Он был разработан для OLAP (Online Analytical Processing) и неэффективен для OLTP (Online Transaction Processing).

Все компоненты сервиса можно разместить на одном хосте. Несколько нод Trino, настроенные для совместной работы друг с другом, составляют кластер Trino.

Концепции

В Trino существуют следующие основные концепции:

- **Коннекторы**
  Коннектор — это плагин Trino, который создает соединение с источником данных с помощью интерфейса поставщика услуг (Service Provider Interface, SPI). Один коннектор может использоваться для доступа к одному или нескольким источникам данных одного типа.

- **Каталог**
  Каталог — это абстракция Trino для доступа к определенному источнику данных (например, базе данных MySQL). Он содержит настройки подключений, учетные данные, URL и т.д. Каждый каталог использует определенный коннектор, указанный в свойстве `connector.name`. Каталог содержит одну или несколько схем, которые также содержат объекты, такие как таблицы или представления. Больше информации о каталогах можно получить в статье Управление каталогами в Trino.

- **Менеджер каталогов**

  Менеджер каталогов — это механизм, отвечающий за загрузку, валидацию и управление каталогами Trino. Сервис Trino предлагает несколько режимов для управления каталогами, а именно:

  - Статический. Считывает конфигурационные файлы и создает объекты каталогов в момент запуска сервера Trino. Добавление нового каталога требует перезапуска сервиса Trino.
  - Динамический. Позволяет управлять каталогами динамически, используя команды `CREATE CATALOG` и `DROP CATALOG`.
  - Реализация DBCatalogManager. Позволяет управлять каталогами динамически и хранит метаданные каталога в базе данных ADPG/PostgreSQL. Позволяет обновлять конфигурации каталогов без перезапуска сервиса Trino.

- **Схема**
  Схема определяет данные, которые можно запросить. Для реляционной базы данных схема Trino повторяет схему в целевой базе данных, но для других источников данных схема может быть представлена как другой тип организации таблиц.

- **Выражение (statement)**
  Выражение в Trino — это текстовое представление оператора, написанного на SQL. При выполнении оператора Trino создает запрос и план запроса, которые реализуются как ряд взаимосвязанных этапов, выполняемых на рабочих узлах Trino.

- **Этап (stage)**
  В Trino каждое выполнение запроса делится на несколько этапов, имеющих древовидную иерархию. Когда Trino собирает данные из хранилища, он создает корневой этап для агрегирования выходных данных других этапов, а те уже выполняют задачи распределенного плана самого запроса.

- **Задача (task)**
  Задача Trino — это часть запроса, которая выполняется на сервере Trino Worker. Каждый этап выполнения может быть представлен несколькими задачами.

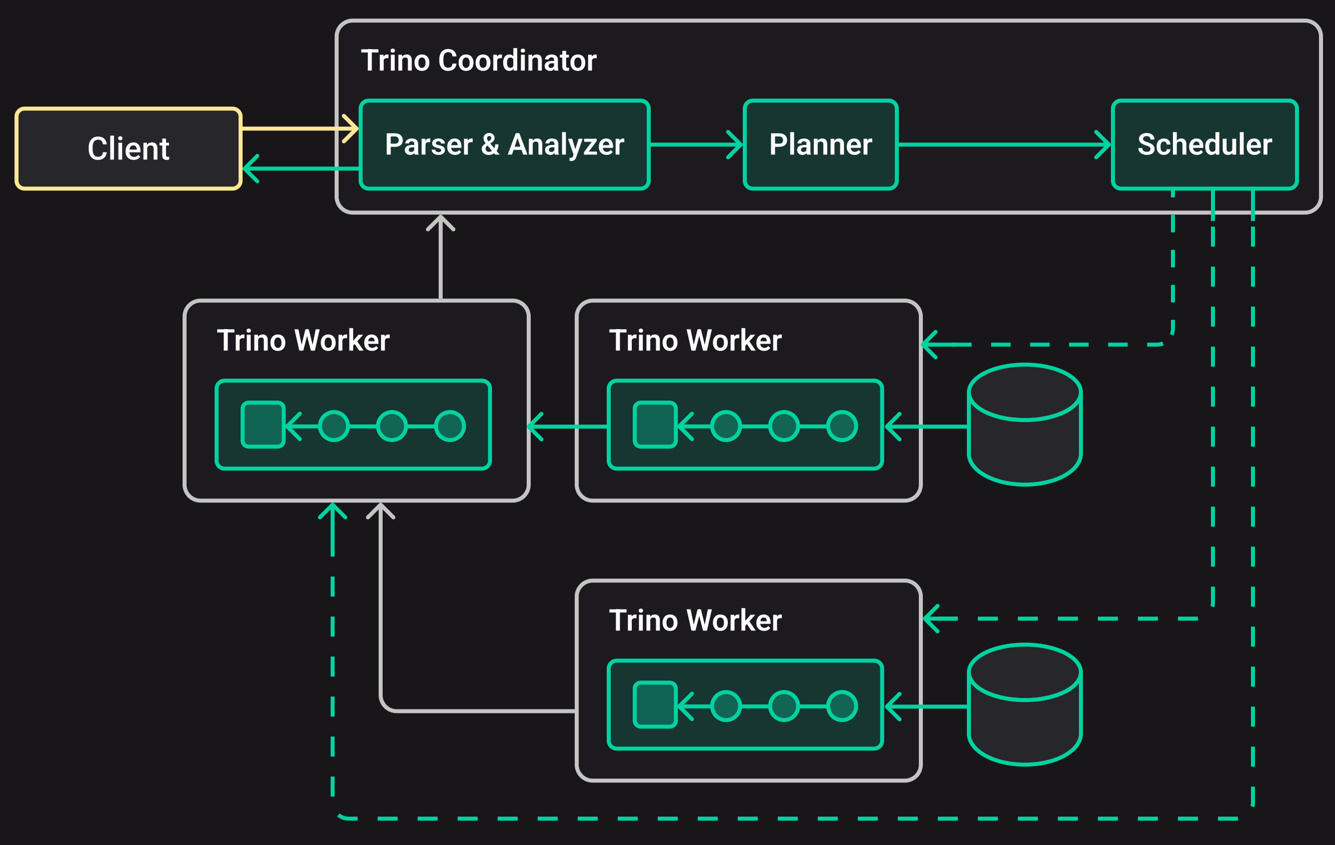
- **Сплит (split)**
  Сплит — это небольшая часть целевого набора данных, обрабатываемая на раннем этапе выполнения запроса. При составлении плана выполнения запроса Trino Coordinator запрашивает у коннектора список всех сплитов, доступных для таблицы.

Компоненты

Архитектура Trino основана на следующих компонентах:

- **Trino Coordinator**
  Координатор — это сервер Trino, обрабатывающий входящие запросы клиентов. Он управляет серверами Trino Worker, которые выполняют эти запросы. Координатор отслеживает какие узлы выполняют задачи и какие сплиты обрабатываются задачами.

- **Trino Worker**
  Trino Worker — это сервер Trino, отвечающий за выполнение задач и обработку данных.



Архитектура Trino

Кластер Trino состоит из одного сервера Trino Coordinator и нескольких Trino Worker. Вся коммуникация и передача данных между компонентами Trino происходит через HTTP/HTTPS.

Коннекторы

Trino содержит множество встроенных коннекторов для различных источников данных:

- Озера и хранилища данных: HDFS, Hive, Iceberg и другие.
- Реляционные базы данных: MariaDB, MySQL, PostgreSQL, Oracle, ADPG и т.д.
- Другие системы и утилиты: Cassandra, ClickHouse, OpenSearch, Prometheus, Snowflake, JMX и т.д.

Дополнительную информацию о коннекторах можно получить в статье Обзор коннекторов в Trino.

Архитектура YARN

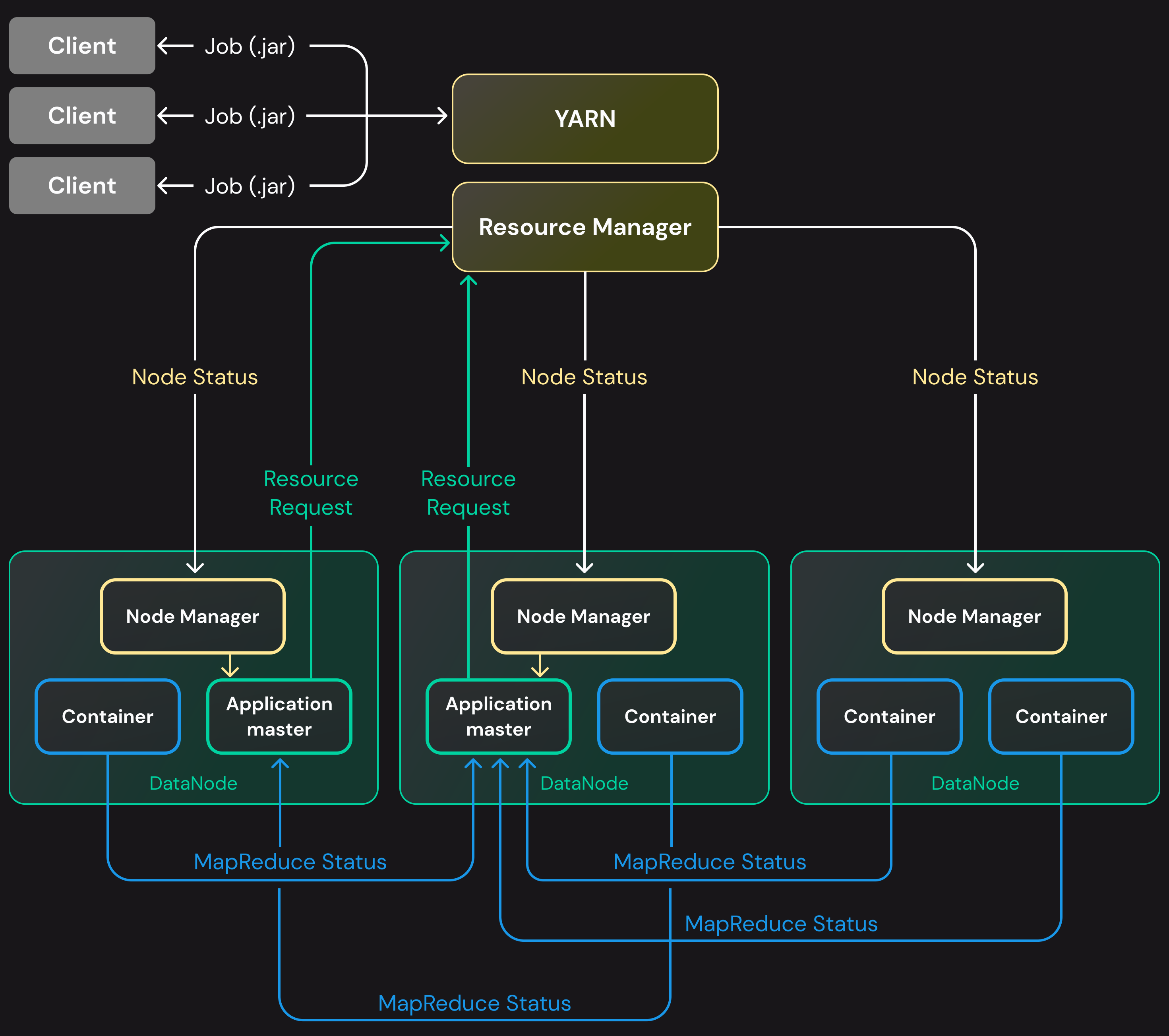
Альберт Багдасарян

Содержание

Назад к содержанию

- Resource Manager
  - Scheduler
  - Application Manager
- Node Manager
- Application Master
- Container

YARN планирует запуск приложений и организует распределение ресурсов кластера между различными процессами обработки данных, такими как пакетная (batch processing), интерактивная (interactive processing) и потоковая (stream processing).



Компоненты YARN

YARN включает следующие компоненты:

- **Resource Manager** — главный сервис по управлению распределением ресурсов в кластере.
- **Node Manager** — сервис, устанавливаемый на каждую ноду HDFS DataNode и отвечающий за предоставление ресурсов приложениям, работающим на этой ноде. Он подчиняется Resource Manager.
- **Application Master** — временный процесс, координирующий полный цикл работы одного приложения. Он взаимодействует с Resource Manager и Node Managers для запуска и мониторинга работы приложений.
- **Container** — это логический набор ресурсов, включая часть оперативной памяти, ядер процессора, сетевых ресурсов, диска и других исчисляемых ресурсов одной ноды. Эти ресурсы выделяются для одного приложения.

Resource Manager

Resource Manager является главным "распорядителем" всех вычислительных ресурсов кластера. Он получает данные о наличии ресурсов и принимает решение о выделении доступных ресурсов конкурирующим между собой приложениям. Когда он получает запрос на выделение ресурсов, он распределяет его в виде запросов к Node Managers тех нод, где будет производиться обработка данных. Resource Manager состоит из двух главных компонентов: планировщика (Scheduler) и менеджера приложений (Application Manager).

Scheduler

Планировщик является частью Resource Manager и выполняет следующие функции:

- Отвечает за предоставление ресурсов приложениям с учетом ограничений на ресурсы, очереди и так далее.
- Выполняет исключительно функции планирования, не затрагивая функции мониторинга и отслеживания состояния приложений. Если приложение перестает работать, планировщик не обеспечивает его восстановление.
- Планирует запуск заданий (job), исходя из запросов ресурсов, приходящих от приложений.
- Реализует свой алгоритм с помощью встраиваемого сервиса планирования. Им может быть Capacity Scheduler или Fair Scheduler, встраиваемые в Resource Manager. По умолчанию используется Capacity Scheduler.

Application Manager

Это еще одна часть Resource Manager, реализующая следующие возможности:

- Принимает запросы на создание заданий (job).
- Запрашивает Resource Manager создать первый контейнер для запуска в нем процесса Application Master, специфичного для приложений определенного типа.
- Управляет работающими процессами Application Master и обеспечивает повторный старт такого процесса в случае сбоев.

Node Manager

Node Manager обеспечивает выделение ресурсов и управление приложениями в пределах одной ноды. Его главной функцией является управление контейнерами для приложений по запросам от Resource Manager.

Node Manager выполняет следующие операции:

- Посылает в Resource Manager отчеты о состоянии ноды и наличии ресурсов.
- Создает и контролирует контейнеры, в которых запускает процессы по запросам от Application Master.
- Контролирует использование ресурсов в контейнерах.
- Предоставляет сервис ведения журналов для контейнерных процессов.
- Удаляет контейнеры по запросам от Application Master или Resource Manager.

Application Master

Любое приложение начинает свою работу в YARN в качестве отдельного задания (job). У каждого работающего приложения есть свой Application Master. Каждый тип приложений имеет свой Application Master, реализованный в виде JAR-файла и являющийся одним из необходимых приложению ресурсов. Например, задачи MapReduce используют `MRAppMaster`.

Application Master координирует процессы приложения в кластере, включая слежение за сбоями и восстановление после них. Он запрашивает выделение ресурсов через Resource Manager и взаимодействует с Node Managers для запуска приложений в контейнерах и мониторинга процесса их выполнения. После своего старта он посылает периодические сигналы (heartbeat), содержащие данные о собственном состоянии и обновленный список необходимых ресурсов.

Container

Контейнер представляет собой коллекцию физических ресурсов, таких как оперативная память, ядра процессора и дисковое пространство, выделенные для определенного приложения в ноде кластера. Каждый контейнер YARN имеет собственный контекст запуска (launch context), используемый в течение всего жизненного цикла контейнера. Это запись, содержащая значения переменных окружения, токены безопасности, данные для передачи в Node Manager, команду для запуска процесса в контейнере и другие параметры. Контекст запуска гарантирует заданию (job) привилегии на использование определенного количества ресурсов (памяти, ядер процессора и другие) в данной ноде.

Zeppelin

Константин Алпашкин, Дарья Барышева

Содержание

Назад к содержанию

Обзор

Архитектура

Интерпретаторы (Interpreter)

Заметки (Note) и параграфы (Paragraph)

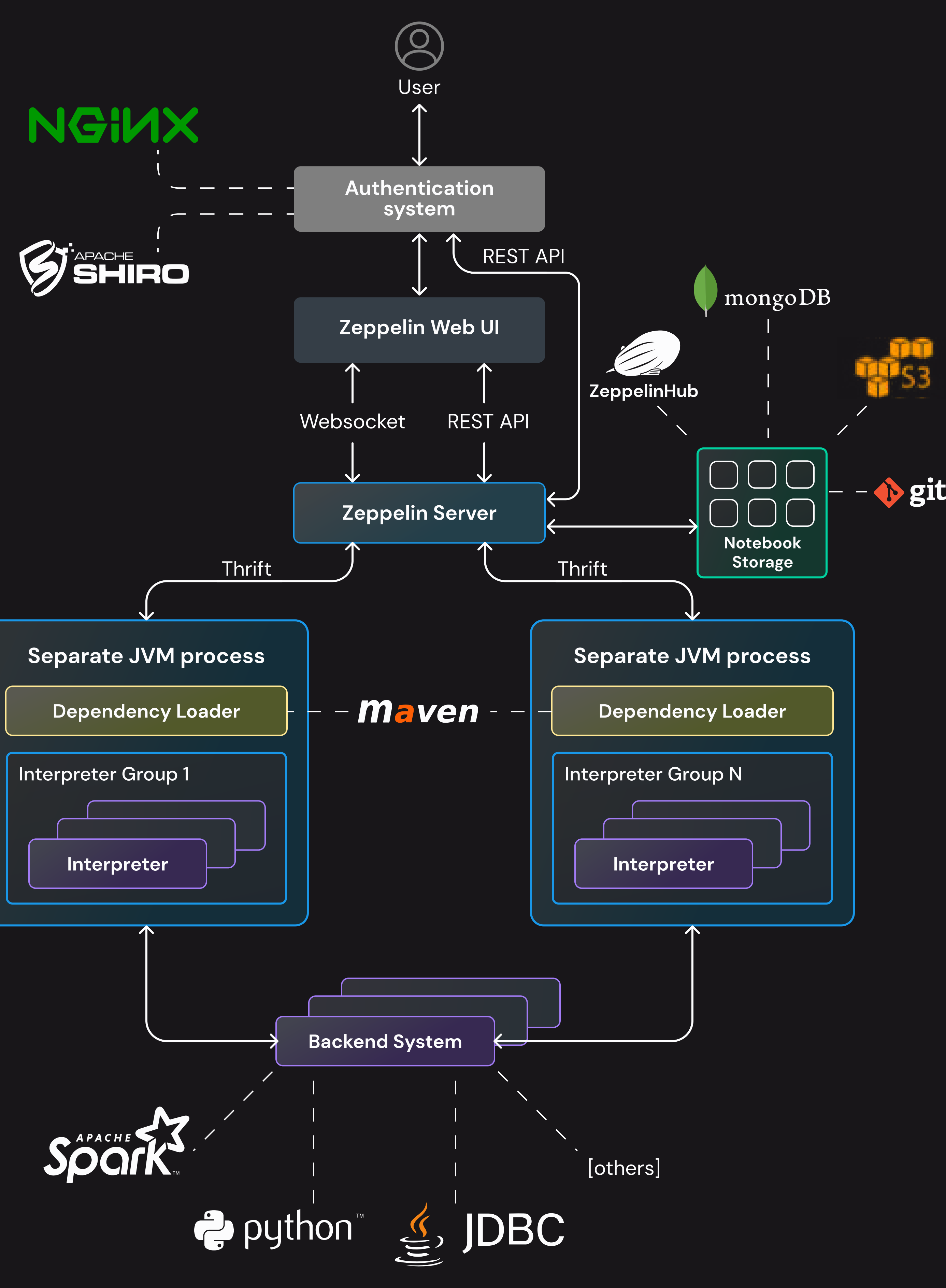
Обзор

Zeppelin — это интерактивный web-блокнот (notebook), предназначенный для решения следующих задач при работе с Big Data:

- **Получение данных из различных источников**. Благодаря механизму интерпретаторов (Interpreter), Zeppelin совместим фактически с любым языком программирования или платформой обработки данных.
- **Визуализация и аналитика данных**. Web-интерфейс Zeppelin поддерживает множество способов визуализации результатов выполнения кода: таблицы, гистограммы (bar chart), круговые диаграммы (pie chart), линейные диаграммы (line chart), диаграммы рассеяния (scatter chart). Для добавления полей ввода можно использовать динамические формы. Кроме этого, Zeppelin позволяет добавлять html непосредственно в поля вывода. Эти и иные функции Zeppelin помогают специалистам в области данных (data scientists) получать исчерпывающую информацию для аналитики.
- **Совместная работа с данными**. Zeppelin-заметки (Note) имеют собственный URL, который можно предоставить пользователям. Любые изменения в заметках отслеживаются в режиме реального времени.

Архитектура

Верхнеуровневая архитектурная схема Zeppelin представлена ниже.



Архитектура Zeppelin

Остановимся на основных моментах схемы:

- Пользователи подключаются к Zeppelin, используя одну из систем аутентификации. Например, Nginx, Apache Shiro и так далее.
- Авторизованные пользователи могут работать с Zeppelin через web-интерфейс либо напрямую с Zeppelin server, используя доступные REST API.
- Заметки (Note), создаваемые пользователями, могут храниться в различных системах. Среди них: Git, S3, MongoDB, ZeppelinHub.
- Ключевую роль в Zeppelin играют интерпретаторы (Interpreter). Они отвечают за использование различных языков программирования и backend-систем: Spark, Python, JDBC, SQL и так далее. Интерпретаторы взаимодействуют с Zeppelin Server через Apache Thrift.
- По умолчанию Zeppelin использует центральный Maven-репозиторий для разрешения зависимостей и загрузки библиотек, необходимых для интерпретаторов. Также существует возможность загрузки библиотек из локальной файловой системы и внешних источников. Дополнительная информация о разрешении зависимостей доступна в документации Zeppelin.

Интерпретаторы (Interpreter)

Интерпретатор (Interpreter) в Zeppelin — это плагин, который позволяет использовать определенный язык программирования или backend-платформу для извлечения и анализа данных. Например, интерпретатор `%sh` предназначен для запуска shell-команд, интерпретатор `%jdbc` — для jdbc-запросов и так далее.

Каждый интерпретатор входит в определенную группу. Интерпретаторы из одной группы запускаются в рамках одного JVM-процесса. Таким образом, группа — это средство запуска и остановки интерпретаторов. Каждая группа может включать несколько интерпретаторов, но один интерпретатор всегда принадлежит только одной группе. Например, группа интерпретаторов Spark состоит из таких интерпретаторов, как `%spark`, `%sql`, `%pyspark`, `%ipyspark`, `%r`, а для разрешения зависимостей использует `%dep`. Интерпретаторы, входящие в одну группу, могут обращаться друг к другу.

Список доступных community-интерпретаторов можно найти в документации Zeppelin. Фактически они представляют собой группы интерпретаторов. Некоторые из групп включают один интерпретатор, некоторые — несколько. Новый интерпретатор можно добавить в одну из существующих групп. Кроме этого, существует несколько сторонних (3rd party) интерпреторов — они также могут быть загружены в Zeppelin. Наконец, есть возможность создать собственный интерпретор, не входящий ни в одну из существующих групп.

В таблице ниже описаны основные действия, применимые к интерпретаторам Zeppelin.

Возможные действия с интерпрераторами

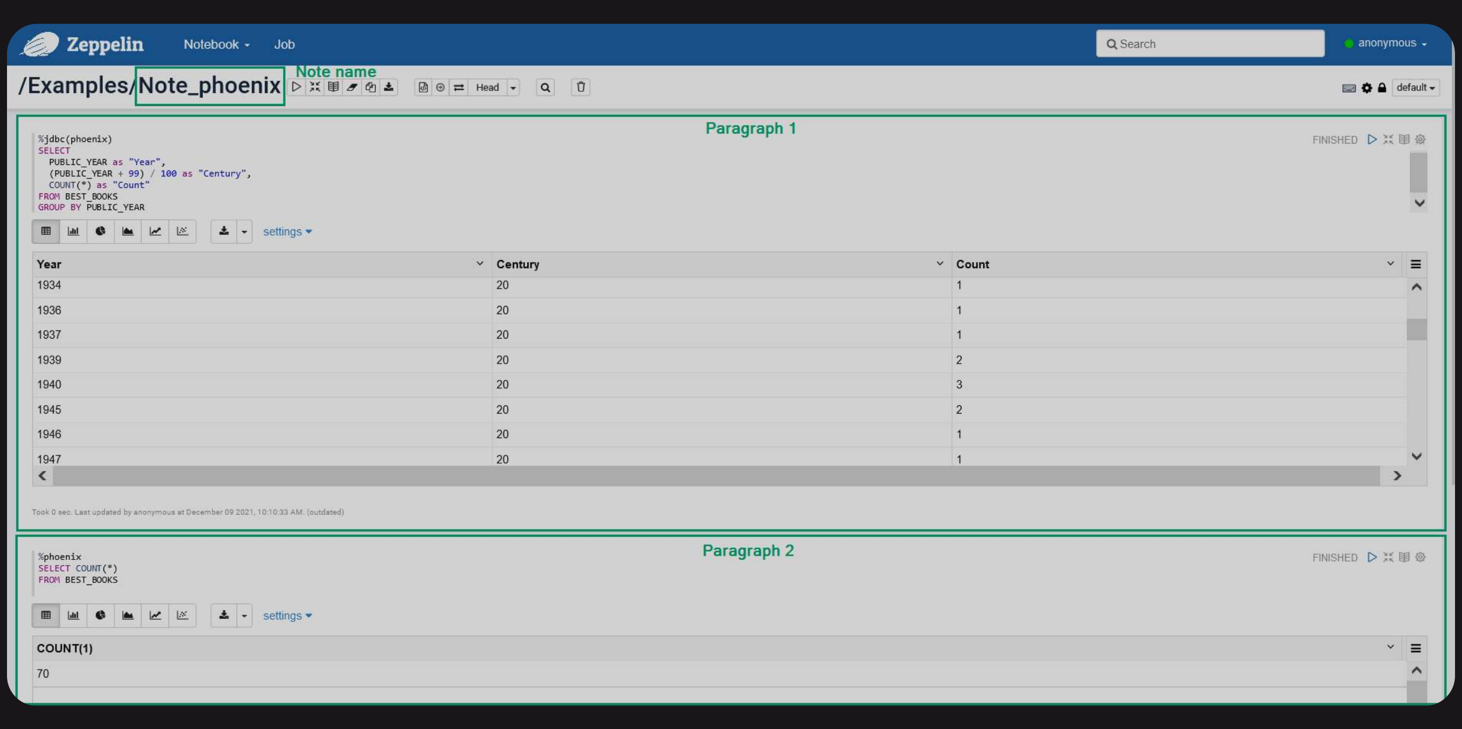
Действие	Пример/Документация
Добавление нового интерпретатора в существующую группу	Пример
Создание нового интерпретатора	Документация
Установка стороннего (3rd party) интерпретатора	Пример

Заметки (Note) и параграфы (Paragraph)

Заметка (Note) в Zeppelin — это набор фрагментов кода (code snippets), написанных на языках программирования, которые поддерживаются интерпретаторами, связанными с данной заметкой. У каждой заметки есть URL-адрес, которым можно поделиться с другими пользователями.

Одна заметка может состоять из нескольких **Параграфов (Paragraph)**. Эти параграфы могут быть как запросами к различным backend-системам, так и пояснительными текстами, построенными с помощью html. Каждый параграф включает два раздела: программный код и результат его выполнения. Любой из этих разделов может быть скрыт.

Веб-интерфейс Zeppelin поддерживает несколько режимов отображения заметок, включая полное сокрытие программного кода.



```
%jdbc(phoenix)
SELECT
  PUBLIC_YEAR as "Year",
  (PUBLIC_YEAR + 99) / 100 as "Century",
  COUNT(*) as "Count"
FROM BEST_BOOKS
GROUP BY PUBLIC_YEAR
```

Year	Century	Count
1934	20	1
1936	20	1
1937	20	1
1939	20	2
1940	20	3
1945	20	2
1946	20	1
1947	20	1

Took 0 sec. Last updated by anonymous at December 09 2021, 10:10:33 AM. (outdated)

```
%phoenix
SELECT COUNT(*)
FROM BEST_BOOKS
```

COUNT(1)
70

Структура заметок

ПРИМЕЧАНИЕ

Для получения дополнительной информации о заметках и параграфах можно обратиться к документации Zeppelin.

ZooKeeper

Альберт Багдасарян

Содержание

Назад к содержанию

- Архитектура
- Модель данных
  - Znodes
  - Watches
- Принципы работы
  - Сессии
  - Чтение и запись

ZooKeeper — это централизованный координационный сервис, выполняющий следующие функции в кластере Hyperwave:

- Обслуживание конфигурационной информации.
- Регистрация наименований.
- Обеспечение распределенной синхронизации.
- Обеспечение защиты от ошибок вида "состояние гонки" (race condition) и "взаимная блокировка" (deadlock).
- Предоставление групповых сервисов.

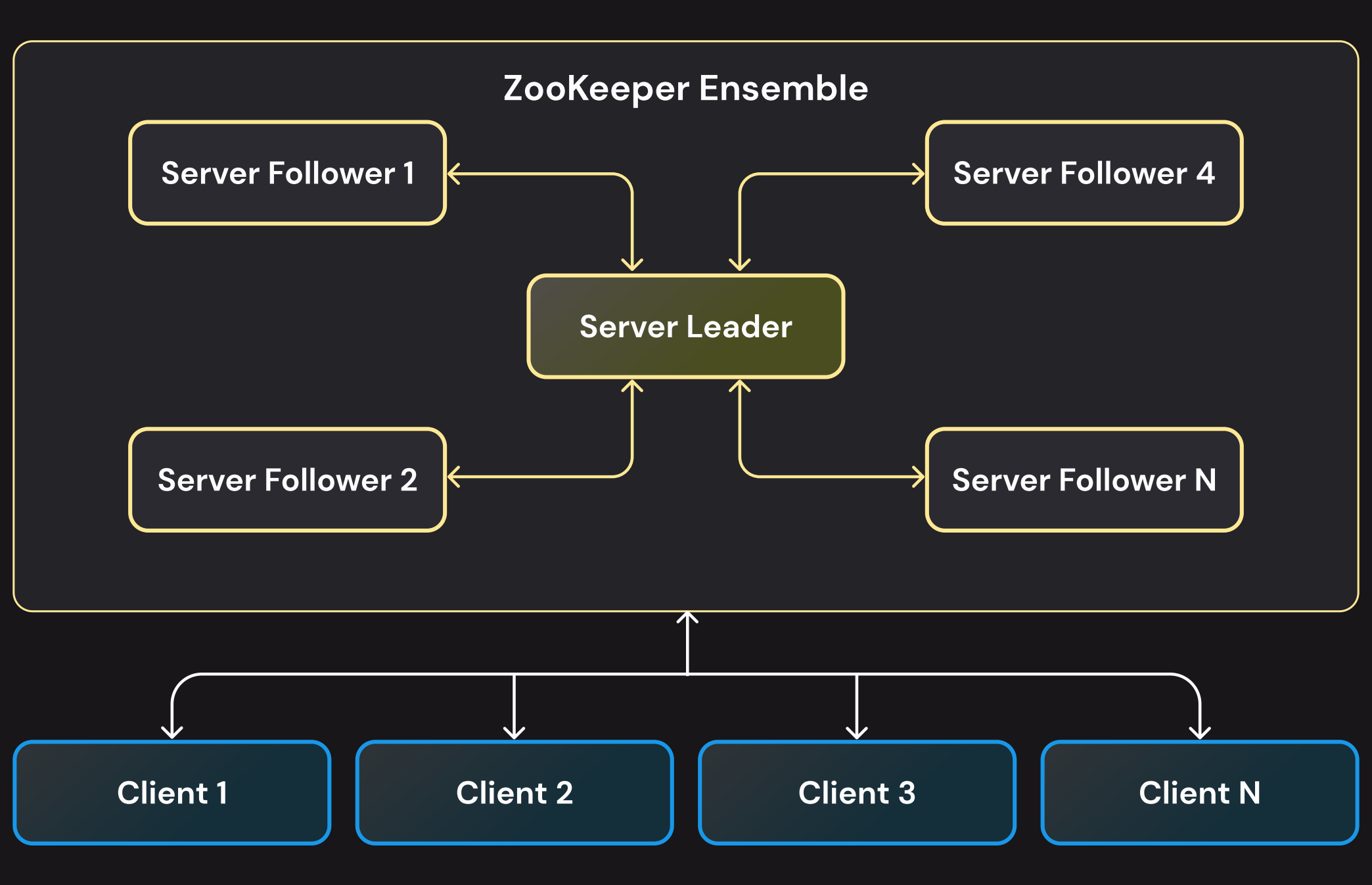
Этот сервис основан на Java с привязкой к Java и C. Его простая архитектура и персонифицированный API помогают избежать возможных ошибок при управлении большими распределенными системами.

Архитектура

Zookeeper использует архитектуру клиент-сервер и содержит следующие компоненты:

- **Ensemble** — это группа (ансамбль) серверов ZooKeeper в количестве не менее трех для обеспечения надежности. Когда один сервер становится недоступным, оставшиеся два образуют кворум (**quorum**), который продолжает работу кластера. Рекомендуется использовать нечетное количество серверов в кластере, то есть: 3, 5, 7 и так далее.
- **Server** — один из участников ансамбля Zookeeper. Его назначение состоит в предоставлении всех видов сервиса, необходимых клиентам. Существуют два типа серверов:
  - **Server Leader** — лидер, обеспечивает автоматическое восстановление работоспособности кластера в случае сбоев на серверах другого типа. Серверы выбирают лидера при начальном запуске ZooKeeper.
  - **Server Follower** — ведомый сервер, которым является каждый сервер ансамбля, кроме лидера. Он следует указаниям лидера.
- **Client** — клиент, являющийся потребителем сервисов ZooKeeper.

Компоненты ZooKeeper представлены на следующей схеме.



Компоненты ZooKeeper

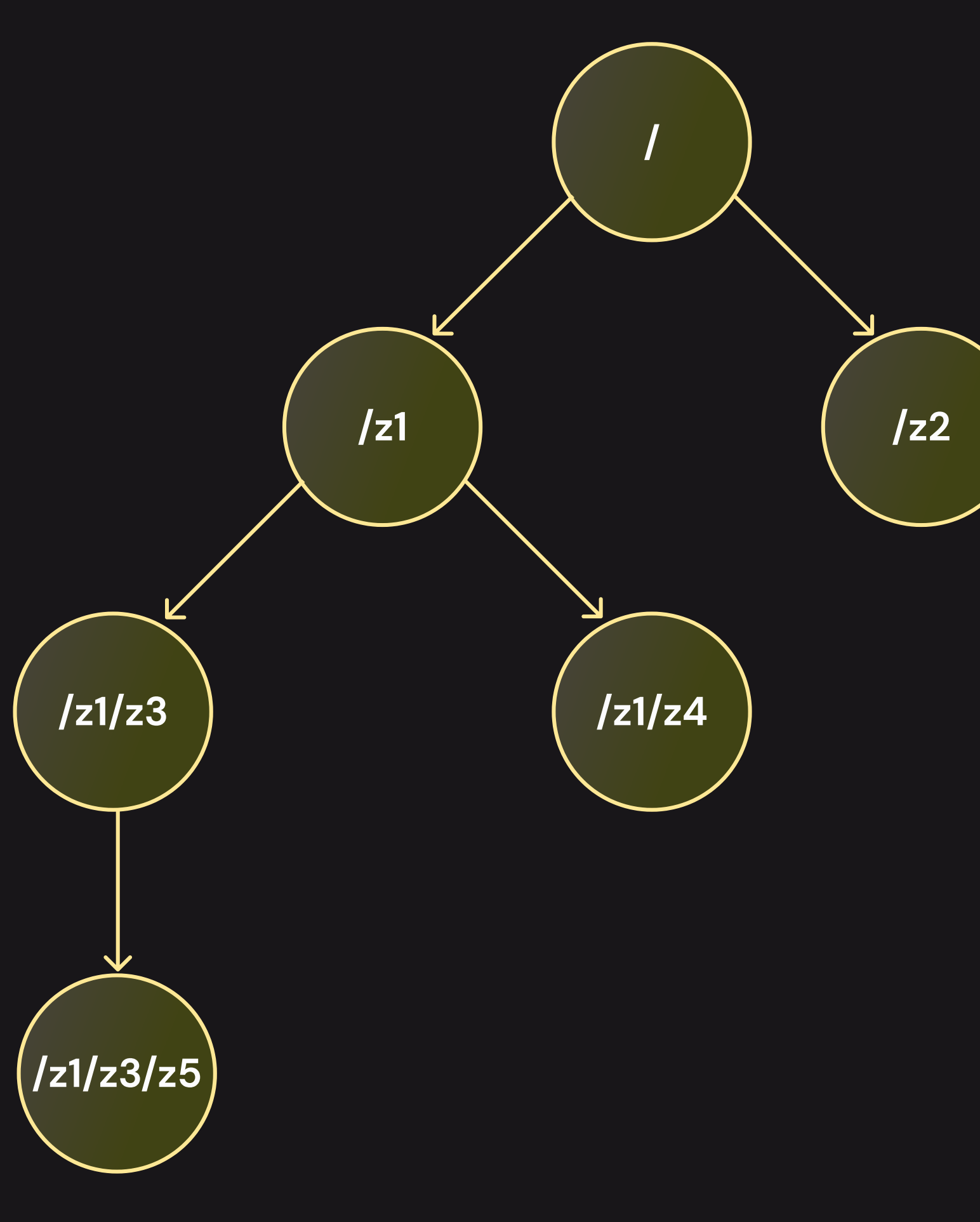
Модель данных

Znodes

ZooKeeper позволяет распределенным процессам взаимодействовать между собой посредством разделяемого иерархического пространства имен (namespace), организованного подобно обычной файловой системе. Каждая нода в этом пространстве называется **znode** и идентифицируется с помощью ее пути. В отличие от типичной файловой системы, znode выполняет роль файла и каталога одновременно. Все данные хранятся в оперативной памяти, что позволяет сервису ZooKeeper обеспечивать высокую пропускную способность и малую задержку.

> **ВНИМАНИЕ**
>
> Поскольку ZooKeeper разработан для хранения координационных данных (конфигурация, состояния, информация о локации и так далее), предполагается, что объем этих данных не будет слишком большим. Поэтому объем данных в znode не может превышать 1 МБ.

На следующей диаграмме дано схематичное изображение модели данных в ZooKeeper. Пример содержит корневую znode, идентифицированную как */*, и две дочерние znodes: */z1* и */z2*. Эти znodes могут иметь свои дочерние znodes, последние — свои и так далее.



Модель данных ZooKeeper

Операции чтения и записи данных в znode атомарные:

- При чтении клиент получает все данные, хранящиеся в указанной znode.
- При записи происходит полная замена всего содержимого znode.

Каждая znode содержит метаданные в составе структуры stat. Она включает номера версий, временные метки (timestamps), идентификаторы соответствующих транзакций и другие метаданные. Znodes защищены посредством списков доступа (Access Control Lists, ACLs), содержащих перечень пользователей с их привилегиями.

Возможные типы znodes приведены в следующей таблице.

Типы znodes

Тип	Описание
Persistent	Тип znodes, используемый по умолчанию. Znodes такого типа продолжают существовать, даже если создавший их клиент не подключен к серверам кластера
Ephemeral	Тип znodes, время существования которых определяется продолжительностью клиентской сессии (в рамках которых они созданы). Ephemeral znodes не могут иметь дочерних znodes
Sequential	Когда клиент запрашивает создание znode такого типа, ZooKeeper добавляет к указанному полному пути znode 10-значный последовательный номер, например */top/myspace/mynode0000000001*. Sequential znodes играют важную роль в блокировании и синхронизации процессов. Sequential znodes являются одновременно persistent или ephemeral. Это означает, что существует не один тип, а два: **persistent-sequential** и **ephemeral-sequential**
Container	Тип znodes, созданный для упрощения выполняемой ZooKeeper очистки от неиспользуемых более znodes (znodes, оставшихся без дочерних нод). Container-нода аналогична Persistent-ноде, но с дополнительной функцией постановки в кандидаты на удаление ZooKeeper-сервером, как только удаляется ее последняя дочерняя znode

Watches

Watches — это специальный механизм, предоставляющий клиентам возможность получать уведомления об изменениях в указанных ими znodes. Объект watch удаляется сразу после его срабатывания, произошедшего в результате изменения наблюдаемой znode.

Watches используется обычно теми операциями, которые следят за состоянием ZooKeeper: проверка наличия znode, чтение данных из znode и получение списка дочерних znodes.

> **ВНИМАНИЕ**
>
> Watch основан на использовании одноразового триггера. После того, как он сработал, последующих срабатываний не будет, пока клиент не создаст новый watch.

Принципы работы

Сессии

ZooKeeper использует механизм сессий. **Сессия (session)** — это промежуток времени, в течение которого клиент взаимодействует с сервером для получения требуемых услуг. В каждой сессии сервер исполняет запросы клиента поочередно в порядке поступления (FIFO). При создании сессии клиент получает идентификатор сессии.

Клиент подключается только к одному серверу для обслуживания всех своих запросов. Это может быть лидер или ведомый сервер. Если клиент не получает подтверждения успешного соединения, он пытается подключиться к другому серверу ZooKeeper.

После установления соединения клиент посылает контрольные сигналы (heartbeat) на сервер для поддержания сессии. Если ZooKeeper не получает контрольный сигнал от клиента по истечении заданного периода (timeout), он полагает, что клиент недоступен.

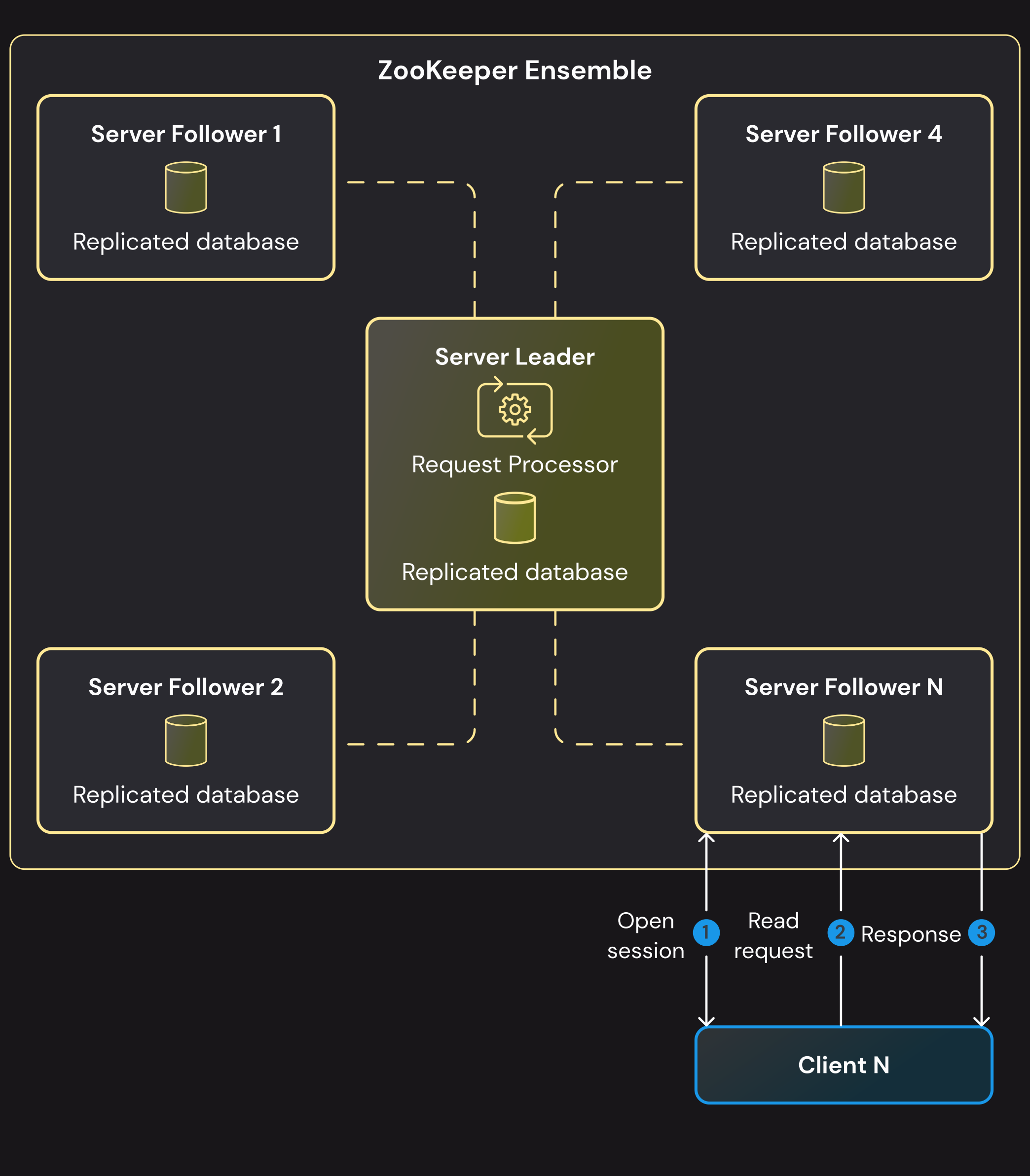
Чтение и запись

Каждый сервер в ансамбле ZooKeeper содержит реплику базы данных. Она хранится в оперативной памяти (in-memory database) и содержит полное дерево znodes. Обновления в дереве серверы записывают в журнал на диске, что позволяет восстановить базу данных при необходимости.

Наличие реплицированной базы данных позволяет серверам выполнять операции чтения без обращения к лидеру. Запросы на запись обрабатываются лидером с использованием специального протокола соглашения (agreement protocol).

Типичная операция чтения состоит из следующих шагов:

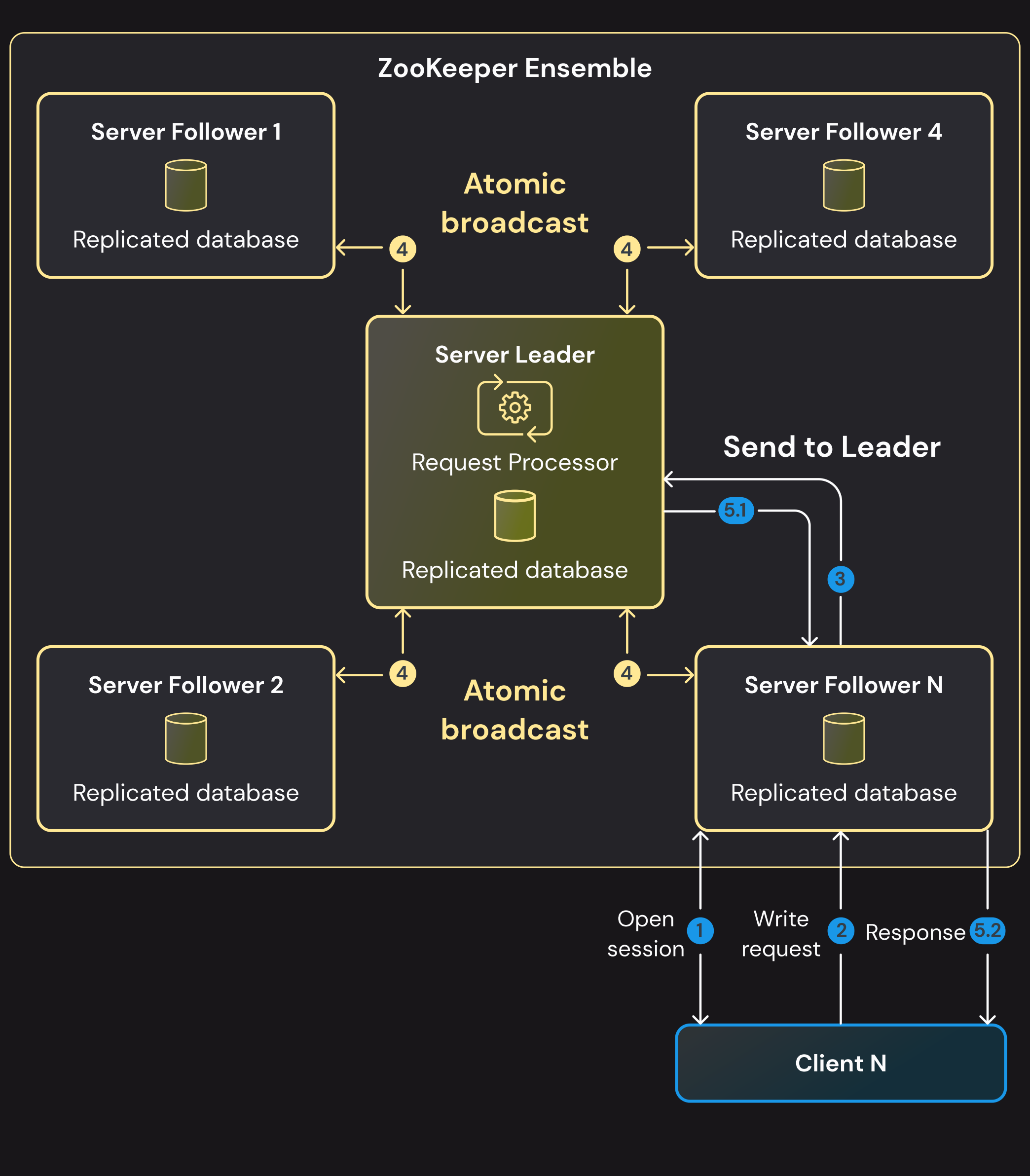
1. Клиент устанавливает сессию с одним из серверов.
2. Клиент посылает серверу запрос на чтение. Запрос включает путь к znode, содержащей требуемые данные.
3. Сервер предоставляет клиенту требуемую информацию, хранящуюся в реплике базы данных на этом сервере. Для этой операции связь с лидером не требуется.



Типичная последовательность чтения

Типичная последовательность записи выглядит следующим образом:

1. Клиент запускает сессию на одном из серверов.
2. Клиент посылает серверу запрос на запись. Запрос включает путь к znode, данные которой необходимо перезаписать.
3. Сервер посылает запрос на запись лидеру.
4. Лидер пересылает запрос всем ведомым серверам посредством атомарной широковещательной рассылки (atomic broadcast). Запрос на запись считается обработанным, если большинство ведомых серверов (кворум) подтвердит успешное завершение запроса. В таком случае изменения в базе данных реплицируются на все серверы.
5. Клиент получает ответ о завершении операции.



Типичная последовательность записи

> **ПРИМЕЧАНИЕ**
>
> - Взаимодействие HBase с ZooKeeper описано в статье Архитектура HBase.
> - Примеры основных команд, которые можно выполнить в ZooKeeper CLI, описаны в статье Работа с Znodes.
> - Более подробная информация приведена в документации ZooKeeper:
>   - Overview
>   - Programmer’s Guide
>   - ZooKeeper Internals

Kerberos

Константин Алпашкин

Содержание

Назад к содержанию

Обзор работы Kerberos

Процесс аутентификации

└ Пример

Типы центров распределения ключей (KDC)

Обзор работы Kerberos

Строгая аутентификация и проверка подлинности каждого пользователя является обязательным требованием для безопасного доступа к кластеру Hyperwave. Все пользователи должны быть безошибочно идентифицированы внутри кластера. Пользователи, успешно прошедшие идентификацию, получают доступ к ресурсам (например, файлы и директории) и могут взаимодействовать с кластером (например, выполнять MapReduce-задачи).

Если пользователю выдан доступ к кластеру под определенным именем, все компоненты кластера также доверяют этому имени. Процесс аутентификации может быть упрощен с использованием базы данных Kerberos, которая содержит учетные данные доступа для всех пользователей. Ресурсы Hadoop-кластера (такие как хосты и сервисы) также должны проходить взаимную аутентификацию. Это необходимо для предотвращения доступа к кластеру различных вредоносных систем, действующих под видом компонентов кластера.

Основные понятия Kerberos

Термин	Описание
Key Distribution Center, KDC	Доверенный сервис, предоставляющий тикеты и временные сеансовые ключи
Kerberos KDC Server	Компьютер или сервер, на котором расположен KDC
Kerberos Client	Хост кластера, выполняющий аутентификацию через KDC
Kerberos Principal	Уникальное имя пользователя или сервиса, который проходит аутентификацию через KDC. Имя состоит из базового имени пользователя и полного доменного имени хоста, на котором развернут сервер
Key	Уникальный ключ, присваиваемый каждому принципалу. Данный ключ отправляется в систему аутентификации
Keytab	Файл, который содержит записи одного или нескольких принципалов и их ключи. Данный файл хранится в базе данных Kerberos и запрашивается принципалом сервера. Затем файл сохраняется в защищенной директории в узле сервера
Realm	Сеть Kerberos, которая включает KDC и одного или нескольких клиентов
KDC Admin Account	Аккаунт администратора, который используется в ADCM для создания новых принципалов и генерации KDC-ключей
Ticket Granting Server (TGS)	Сервер, выдающий тикеты аутентификации (ticket) и права доступа
Authentication Server (AS)	Сервер, выполняющий первоначальную проверку аутентификации и выпускающий Ticket Granting Ticket (TGT)
Ticket Granting Ticket	Тикет на выдачу тикета (TGT). Включает в себя следующее: вторая копия ключа сессии, имя пользователя, окончание действия тикета. Тикет зашифрован мастер-ключом KDC, который известен только KDC. Это означает, что только KDC может расшифровать TGT

Процесс аутентификации

Hadoop использует Kerberos для строгой аутентификации и для проверки подлинности пользователей и сервисов. Сервер Kerberos называется **Key Distribution Center** (KDC). Он состоит из трех ключевых компонентов:

- **База данных пользователей и сервисов MIT KDC** — хранит принципалы и соответствующие пароли Kerberos.
- **Сервер Аутентификации (Authentication Server, AS)** — выполняет первоначальную проверку подлинности и выдает тикет на выдачу тикета (Ticket Granting Ticket, TGT).
- **Сервис выдачи тикета (Ticket-Granting Service, TGS)** — выдает Kerberos-токен на основании первичного тикета на выдачу тикета (TGT).

Упрощенный процесс аутентификации Kerberos представлен ниже:

1. Принципал инициализирует процесс аутентификации на сервере аутентификации (AS).
2. В качестве ответа AS отправляет принципалу TGT, зашифрованный паролем принципала. Этот пароль известен только принципалу и AS.
3. Принципал расшифровывает полученный TGT, используя пароль Kerberos. С этого момента каждый принципал может использовать TGT для получения тикетов аутентификации от TGS. Эти тикеты позволяют принципалу получить доступ к различным сервисам.

Так как ресурсы кластера (сервисы и хосты) не могут каждый раз предоставлять пароль для расшифровки TGT, используется файл *keytab*. Этот файл содержит учетные данные для аутентификации ресурса. Совокупность хостов, пользователей и сервисов, управляемых Kerberos-сервером, называется **Realm**.

Пример

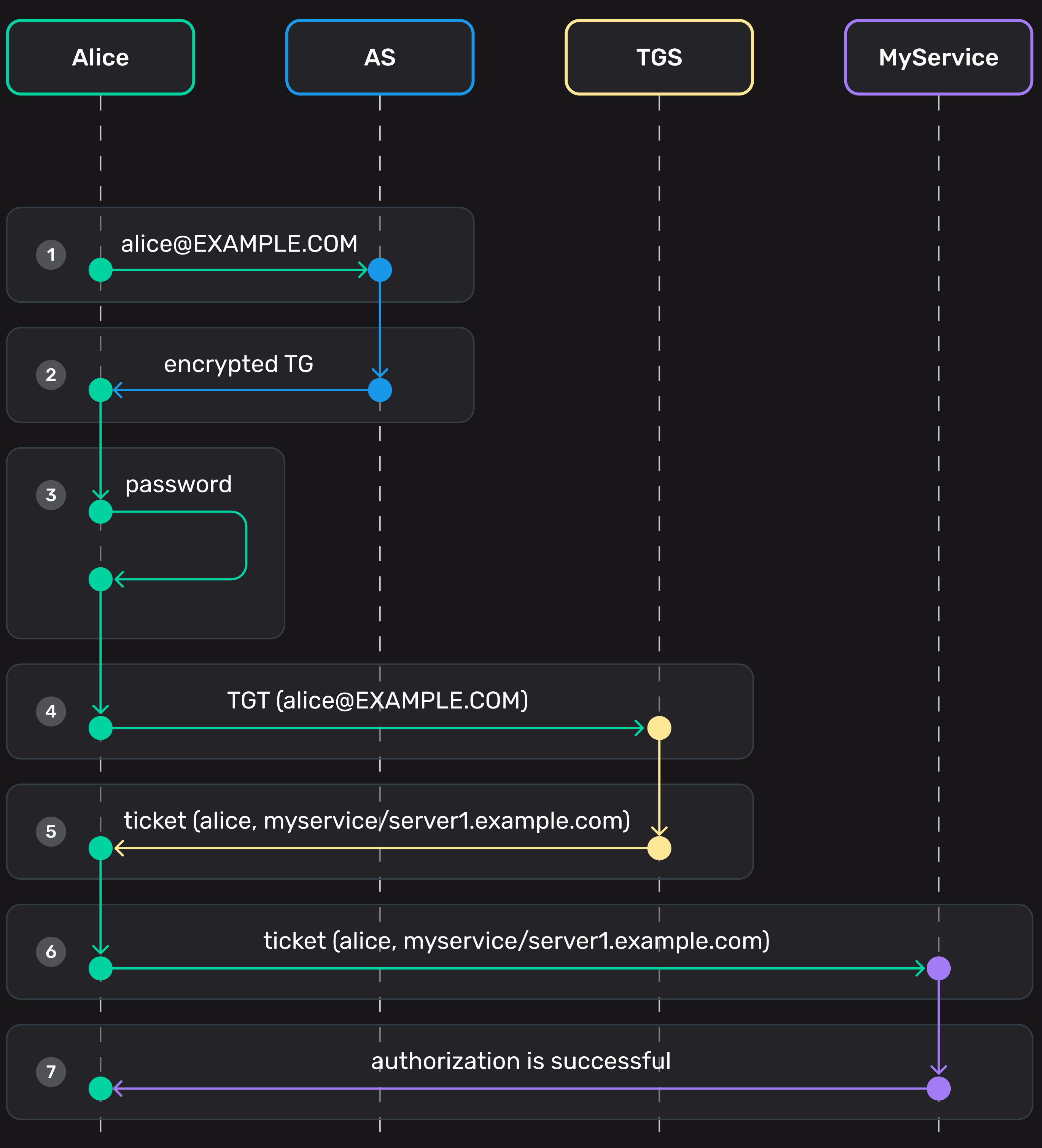
Ниже приведен простой пример процесса аутентификации с использованием Kerberos. В примере задействованы следующие компоненты:

- *EXAMPLE.COM* — Kerberos realm.
- *Алиса* — пользователь с именем принципала (UPN) *alice@EXAMPLE.COM*.
- *myservice* — сервис, расположенный в узле *server1.example.com* и имеющий имя принципала сервиса (SPN) *myservice/server1.example.com@EXAMPLE.COM*.
- *kdc.example.com* — центр распределения ключей (KDC) для области *EXAMPLE.COM*.

Чтобы получить доступ к сервису *myservice*, Алиса должна предоставить действующий тикет аутентификации для *myservice*. Этот процесс описан ниже (некоторые детали пропущены для краткости):

1. Алиса должна получить тикет на выдачу тикета (TGT). Для этого Алиса инициирует запрос к Серверу Аутентификации (AS), который расположен на хосте *kdc.example.com*. Алиса идентифицирует себя как *alice@EXAMPLE.COM*.
2. В качестве ответа, AS отправляет Алисе тикет на выдачу тикета (TGT), зашифрованный паролем принципала *alice@EXAMPLE.COM*.
3. После получения зашифрованного сообщения Алиса использует пароль принципала *alice@EXAMPLE.COM* для декодирования сообщения.
4. После успешного декодирования сообщения с TGT Алиса запрашивает сервисный тикет у сервиса выдачи тикетов (TGS, расположен на *kdc.example.com*), чтобы получить доступ к *myservice/server1.example.com@EXAMPLE.COM*. В запросе от Алисы содержится TGT, а в ответе — тикет аутентификации сервиса.
5. Сервис выдачи тикетов (TGS) проверяет полученный TGT и выдает Алисе сервисный тикет аутентификации, зашифрованный ключом принципала *myservice/server1.example.com@EXAMPLE.COM*.
6. Алиса отправляет сервисный тикет на сервис *myservice*. Сервис *myservice* расшифровывает сервисный тикет, используя ключ принципала *myservice/server1.example.com@EXAMPLE.COM*. Также сервис *myservice* выполняет проверку расшифрованного сервисного тикета.
7. После успешной аутентификации Алиса получает права на использование сервиса *myservice*.

Процесс, описанный выше, показан на следующей схеме:



Alice
AS
TGS
MyService
1
alice@EXAMPLE.COM
2
encrypted TG
3
password
4
TGT (alice@EXAMPLE.COM)
5
ticket (alice, myservice/server1.example.com)
6
ticket (alice, myservice/server1.example.com)
7
authorization is successful

Процесс аутентификации Kerberos

Типы центров распределения ключей (KDC)

Для керберизации кластера доступны следующие типы центров распределения ключей (KDC):

- MIT Kerberos — состоит из базы данных принципалов и хранилища ключей Kerberos.
- MS Active Directory — содержит базу данных принципалов и хранилище ключей Windows Server.
- FreeIPA — бесплатная система управления идентификацией пользователей для Linux/UNIX с открытым исходным кодом.
- Samba — ПО, которое можно использовать как контроллер домена Active Directory.

При использовании любого из этих типов процесс идентификации и аутентификации пользователей практически одинаков. Единственное различие заключается в используемом типе центра распределения ключей (KDC).

Релизы ADH

Сергей Тихомиров, Константин Алпашкин

Содержание

Назад к содержанию

- 4.1.0
- 4.0.0
- 3.3.6.2
- 3.3.6.1
- 3.2.4.3
- 3.2.4.2
- 3.2.4.1
- 3.1.2.1
- 2.1.10
- 2.1.8
- 2.1.7
- 2.1.6
- 2.1.4
- 2.1.3
- 2.1.2
- 2.1.1
- 2.1.0

4.1.0

4.1.0.b1

Дата: 30.10.2025

Новые сервисы

Добавлен сервис Flink2 2.0.0_arenadata1 со следующими патчами:

- Fixed OrcShim to support hive 4.
- Добавлен flink-connector-hive 3.0.0_arenadata1 с поддержкой Hive4.

Добавлен компонент HAproxy для сервиса Trino

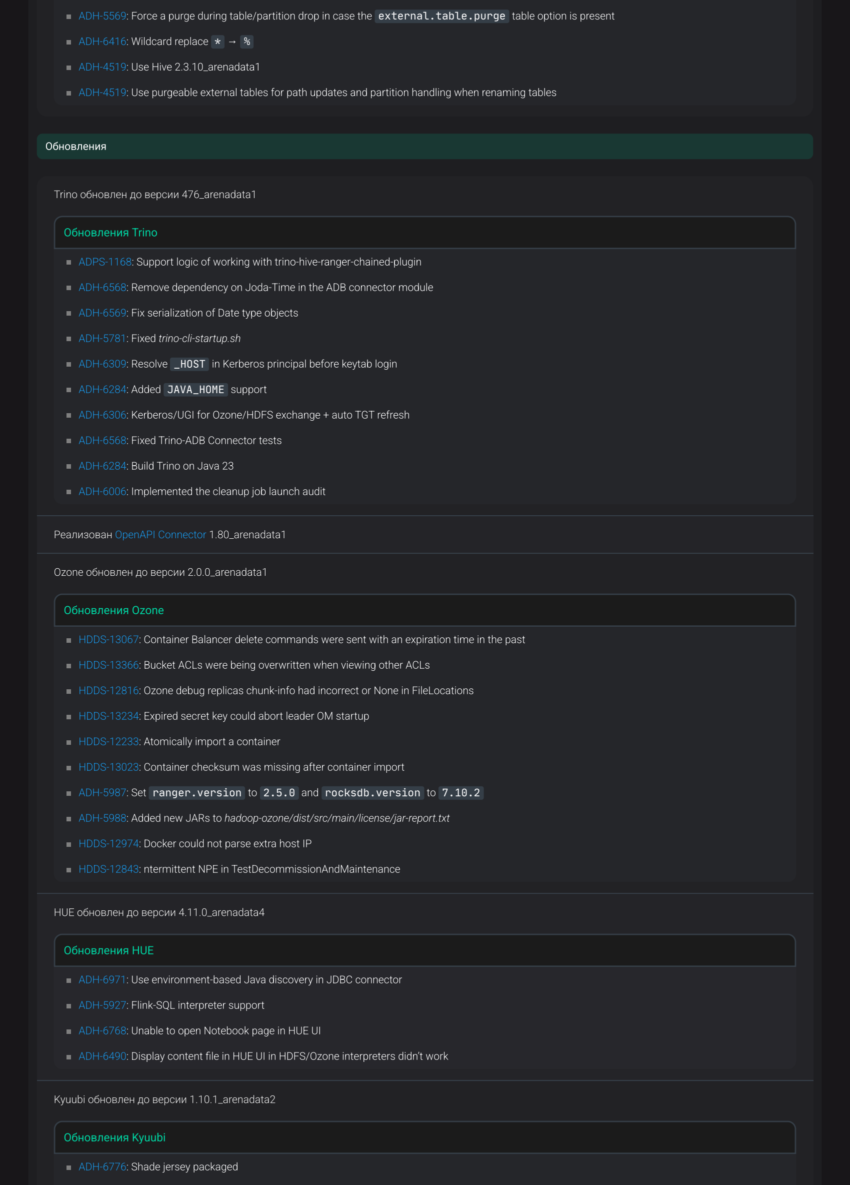
Добавлен компонент Trino Client для сервиса Trino

Добавлен компонент Ozone Client для сервиса Ozone

Добавлен сервис Spark 4.0.0_arenadata1 со следующими обновлениями:

Обновления Spark 4

- ADH-4519: Fixed scalastyle
- ADH-4519: Arenadata patches
- ADH-4519: Updated `io.grpc.version`, set `hadoop.version` and `zookeeper.version`
- ADH-4519: Added a configurable artifact directory path for Spark Connect
- ADH-4803: Added a blacklist filter to exclude certain properties
- ADH-6342: Skip rewriting stats for checkpointed LogicalRDD



- ADH-5569: Force a purge during table/partition drop in case the `external.table.purge` table option is present
- ADH-6416: Wildcard replace `*` → `%`
- ADH-4519: Use Hive 2.3.10_arenadata1
- ADH-4519: Use purgeable external tables for path updates and partition handling when renaming tables

Обновления

Trino обновлен до версии 476_arenadata1

Обновления Trino

- ADPS-1168: Support logic of working with trino-hive-ranger-chained-plugin
- ADH-6568: Remove dependency on Joda-Time in the ADB connector module
- ADH-6569: Fix serialization of Date type objects
- ADH-5781: Fixed *trino-cli-startup.sh*
- ADH-6309: Resolve `_HOST` in Kerberos principal before keytab login
- ADH-6284: Added `JAVA_HOME` support
- ADH-6306: Kerberos/UGI for Ozone/HDFS exchange + auto TGT refresh
- ADH-6568: Fixed Trino-ADB Connector tests
- ADH-6284: Build Trino on Java 23
- ADH-6006: Implemented the cleanup job launch audit

Реализован OpenAPI Connector 1.80_arenadata1

Ozone обновлен до версии 2.0.0_arenadata1

Обновления Ozone

- HDDS-13067: Container Balancer delete commands were sent with an expiration time in the past
- HDDS-13366: Bucket ACLs were being overwritten when viewing other ACLs
- HDDS-12816: Ozone debug replicas chunk-info had incorrect or None in FileLocations
- HDDS-13234: Expired secret key could abort leader OM startup
- HDDS-12233: Atomically import a container
- HDDS-13023: Container checksum was missing after container import
- ADH-5987: Set `ranger.version` to `2.5.0` and `rocksdb.version` to `7.10.2`
- ADH-5988: Added new JARs to *hadoop-ozone/dist/src/main/license/jar-report.txt*
- HDDS-12974: Docker could not parse extra host IP
- HDDS-12843: ntermittent NPE in TestDecommissionAndMaintenance

HUE обновлен до версии 4.11.0_arenadata4

Обновления HUE

- ADH-6971: Use environment-based Java discovery in JDBC connector
- ADH-5927: Flink-SQL interpreter support
- ADH-6768: Unable to open Notebook page in HUE UI
- ADH-6490: Display content file in HUE UI in HDFS/Ozone interpreters didn’t work

Kyuubi обновлен до версии 1.10.1_arenadata2

Обновления Kyuubi

- ADH-6776: Shade jersey packaged

- KYUUBI-7055: Support to filter out server only configs with prefixes
- KYUUBI-7051: Fix usage without sslTrustStore of JDBC driver

Spark обновлен до версии 3.5.4_arenadata2

Обновления Spark

- ADH-6415: Skip rewriting stats for checkpointed LogicalRDD
- ADH-6416: Wildcard replace `*` → `%`

ADB Spark connector обновлен до версии 1.1.0-3.5.4_arenadata2

Обновления ADB Spark connector

- Улучшено разрешение адреса сервера с fallback-механизмом.
- Добавлен автоматический поиск bind-адреса HTTP-сервера при помощи NIC.

Hive обновлен до версии 4.0.1_arenadata2

Обновления Hive

- ADH-6473: Bumped the HBase service version from 2.5.10 to 2.6.3
- ADH-6474: Bumped DataNucleus version from 5.2.8 to 5.2.10
- HIVE-28749: The default `hikaricp.leakDetectionThreshold` was not valid
- HIVE-28545 Removed a global lock on `HMSHandler.getMSForConf` which is prone to deadlock
- HIVE-28900: Make the objectstore-secondary connection pool size configurable
- HIVE-28839: Connection starvation in HMS if DataNucleus value generation fails
- HIVE-28788: ObjectStore failed to commit the transaction could lead to connection starvation

HBase обновлен до версии 2.6.3_arenadata1

Обновления HBase

- hadoop.guava обновлен до версии 32.0.1-jre.

Phoenix обновлен до версии 5.2.2_arenadata1

Обновления Phoenix

- ADH-6575: Added HexQualifierFunction UDF to convert VARBINARY to HEX
- ADH-6575: Downgraded `hadoop-three.version` to 3.3.6

Phoenix Query Server обновлен до версии 6.0.0_arenadata4

Обновления Phoenix Query Server

- ADH-6588: Added default config
- ADH-6588: Bumped the HBase service version from 2.5.10 to 2.6.3

SSM обновлен до версии 2.1.1

Обновления SSM

- ADH-6574: Added filtering for the `-ruleId` arg from the arguments of DistCp call
- ADH-6442: Added support for free port picking on SSM server level
- ADH-6281: Fixed cmdlet stopping logic
- ADH-6189: Fixed sorting field for file_access.access_time column
- ADH-5916: Added new Rules API integration tests

- ADH-6152: Added support for setting the number of cmdlet executors for specific node
- ADH-6071: Made search of several columns case-insensitive

Solr обновлен до версии 8.11.4_arenadata1

Обновления Solr

- ADH-6698: Detect `JAVA_HOME` via bigtop and use proper JVM path in Solr scripts

Новые функции

Добавлена поддержка Astra Linux 1.7 "Воронеж"

Добавлена поддержка AltLinux 10 SP

Реализована утилита для явного выбора Java для всех сервисов

Преднастроен интерпретатор HUE PySpark

Преднастроен интерпретатор HUE Flink 2

В мониторинг были добавлены дашборды для следующих сервисов:

- Trino
- Zeppelin
- HBase
- ZooKeeper
- Solr

Реализован компонент Observer NameNode в высокодоступном кластере HDFS

Реализован режим высокой доступности для компонента Trino Coordinator

Добавлена возможность подготовки хостов (установки на них пакетов) после их добавления в кластер

Реализована возможность хранения паролей и учетных данных в формате JCEKS для сервиса Ozone

Сервис Trino теперь поддерживает политики HadoopSQL при интеграции с Ranger

Реализован OverwriteOnCommitProtocol для Spark3

Улучшения

Улучшены настройки сервиса Trino:

- Добавлена опция `--add-opens` в параметр ***jvm.config*** каждого из компонентов
- Добавлено управление следующими методами аутентификации:
  - Password file
  - Header
  - Certificate
  - JWT
  - OAuth 2.0
- Добавлено управление следующими настройками доступа:
  - File-based
  - OPA Policy

Реализована возможность настройки прокси-сервера для подключения к репозиторию в настройках кластера

Реализована возможность добавления кастомных параметров для действия ***Manage Kerberos***

Улучшен шаблон для логов сервиса Kyuubi

Клиент `kyuubi-beeline` теперь доступен глобально

Добавлена возможность автоматически очищать каталоги Trino при использовании DBCatalogManager

Добавлено автоматическое создание пользовательских директорий из настроек для следующих сервисов:

- Ozone
- Impala
- Zeppelin
- Spark
- Hive
- HBase
- Solr

Добавлен чекбокс ***Metastore DB schema init/upgrade*** в модальном окне для действия ***Restart*** сервиса Kyuubi

Действие ***Check*** сервиса HDFS теперь принимает во внимание наличие сервиса Ozone в кластерах, где параметр ***defaultFS*** сервиса Core configuration настроен на Ozone

Свойство ***ACID Transactions*** сервиса Hive заменено активируемой группой параметров, в которой можно установить дополнительные свойства для более эффективной работы сервиса

Улучшено поведение действия ***Add/Remove components***. Теперь топология кластера передается в ADCM сразу после завершения основной задачи, даже если возникают какие-либо ошибки

Действия сервиса ZooKeeper теперь доступны во вкладке ***Host*** → ***Host-Components***

Для соответствия требованиям безопасности улучшены настройки по умолчанию для плагина Ranger Spark:

Изменения в *ranger-spark-audit.xml*:

- `xasecure.audit.destination.solr.batch.filespool.perms` = `666`
- `xasecure.audit.destination.solr.batch.filespool.dir.perms` = `777`

Изменения в *ranger-spark-security.xml*:

- `ranger.plugin.spark.policy.cache.file.perms` = `666`
- `ranger.plugin.spark.policy.cache.dir.perms` = `777`

Параметры `hadoop.proxyuser.*` теперь редактируемые

Обновлен дашборд HUE в сервисе Monitoring, также добавлена возможность фильтрации по хосту

Интеграция с Iceberg теперь доступна в Community-версии

Исправленные баги

ADB Spark3 connector: метод `DataFrame.count()` завершался с ошибкой, если было установлено количество партиций

Ozone: исправлены параметры ***ozone-httpfs.xml*** в керберизованном кластере

Некорректные JCEKS ***hbase-site.xml*** после действия ***Manage credential encryption***

Невозможность добавить компонент ***Spark3 Client*** в кластер с Ozone, если значением параметров ***spark.yarn.archive*** и ***spark.history.fs.logDirectory*** был путь к OFS

Каталоги Trino Hive/Iceberg зависели от сервиса HDFS

Core: неверные значения `any`-параметров, относящихся к Kerberos, после установки в керберизованном кластере

Ozone: HttpFS UI не работал с Kerberos

Неверные сообщения об ошибках во время работы действия ***Export configuration***, если запрашиваемый хост недоступен

Trino завершал работу с ошибкой при активированном Kerberos и retry-политике `TASK`

Проблемы с интеграцией Impala и внешнего хранилища объектов

Активация плагина Ranger для Solr завершалась с ошибкой

ADB Spark3 connector: параметр ***usehostname*** не влиял на явно указанный адрес ноды

Действие ***Manage SSL*** с активированным параметров ***force*** не обновляло конфигурационные файлы, если ранее были добавлены другие пароли от keystore/truststore

`Admprom set_ssl_certs_rights` работало некорректно с компонентом Prometheus

Изменение параметра `nginx_http_port` не освобождало занятый по умолчанию порт 8089

Ошибка при использовании DistributedShell в YARN

Rager Trino plugin: по умолчанию у пользователя `rangerlookup` не было прав доступа к некоторым политикам

Client configurations: некоторые файлы log4j не появлялись в архивах

Пароли truststore и keystore отображались после активации шифрования учетных данных

Дублирование *spark-history-env.sh* в настройках сервиса

Ошибка IllegalArgumentException при работе Spark History Server

Ошибка при изменении директорий для хранения логов HttpFS

HUE: неверный протокол в ссылке ***webhdfs_url*** после активации SSL

Дублирование действия ***Check*** в списке доступных действий сервиса HBase

Airflow: действие ***Manage SSL*** завершалось с ошибкой на шаге `Airflow check`

Прочее

Перед обновлением кластера удалите сервис MariaDB, так как он признан устаревшим

Минимальная версия ADCM — 2.7.0

4.0.0

4.0.0.b1

Дата: 25.06.2025

Новые сервисы

Добавлен компонент Flink History Server для сервиса Flink

Добавлен новый сервис Monitoring. Включает в себя Grafana, Prometheus и все необходимые экспортеры и преднастроенные дашборды для следующих сервисов и компонентов:

- HDFS (NameNode, DataNode, JournalNode)
- Ozone
- HUE
- Kyuubi
- YARN (ResourceManager, NodeManager, TimeLine Server)
- Hive Server
- Hive Metastore
- Impala (Daemon, Catalog, State Store)

Старый сервис помечен как Deprecated и будет удален в будущих релизах

Обновления

Trino обновлен до версии 468_arenadata2

Обновления Trino

- Реализован коннектор Trino-ADB.
- Реализован DBCatalogManager с версионированием и режимом обслуживания.
- Реализована функциональность миграции каталогов.
- Реализован проброс данных пользователя для JDBC-коннекторов.
- Добавлена поддержка Ozone.

Ozone обновлен до версии 1.4.1_arenadata2

Обновления Ozone

- Добавлена поддержка Trino.

SSM обновлен до версии 2.1.0

Обновления SSM

- Добавлена поддержка имперсонации пользователей.
- Проведен рефакторинг кеша локальной файловой системы.
- [Bugfix] Исправлена таблица на странице ***Cluster Info***.
- [Bugfix] Теперь настройки фильтров сбрасываются при переходе между страницами ***Rules*** и ***Actions***.
- [Bugfix] Некоторые действия, запущенные после начала синхронизации, завершались в скором времени с ошибкой `Timeout`.
- [Bugfix] У действия, завершившегося с ошибкой, время начала и завершения совпадало.
- Добавлено время начала исполнения для сущностей, связанных с действиями.
- Добавлено новое логическое состояние действия — `SCHEDULED`.

Сервис Impala обновлен до версии 4.5.0_arenadata1

Обновления Impala

- Поддержка Apache Hive 4.0.1.

Iceberg обновлен до версии 1.6.1_arenadata1

Обновления Iceberg

- Добавлен табличный параметр `drop.base-directory.enabled` для управления удалением базовой директории таблицы после удаления самой таблицы.

Flink обновлен до версии 1.20.1_arenadata1

Обновления Flink

- Применен патч FLINK-35332.

HUE обновлен до версии 4.11.0_arenadata3

Обновления HUE

- Поддержка Python 3.10.
- Поддержка Kerberos-аутентификации для интерпретатора Trino.
- [Bugfix] Каталоги не отображались в интерпретаторе Trino.
- [Bugfix] При исполнении запросов статус не изменялся, а в логах были ошибки интерпретатора Trino.

Kyuubi обновлен до версии 1.10.1_arenadata1

Обновления Kyuubi

- [Bugfix] Исправлено логирование аудита в Solr для `spark-authz`.
- Добавлен класс `OpenMetadataLineageDispatcher` для отправки lineage-событий на сервер OpenMetadata.
- Добавлено новое возможное значение `OPEN_METADATA` для параметра `spark.kyuubi.plugin.lineage.dispatchers`.

Spark обновлен до версии 3.5.4_arenadata1

Обновления Spark

- Принудительное выполнение очистки во время удаления таблицы/раздела в случае активированной опции `external.table.purge`.
- `external.table.purge` теперь учитывается для обновления путей и обработки разделов при переименовании таблиц.
- Применен патч SPARK-51795.
- Применен патч SPARK-51821.
- Применен патч SPARK-52025.
- Применен патч SPARK-43221.
- Применен патч SPARK-51638.
- Применен патч SPARK-44756.
- Применен патч SPARK-50648.
- Применен патч SPARK-50853.
- Применен патч SPARK-45991.
- Применен патч SPARK-51272.
- Применен патч SPARK-48218.
- Применен патч SPARK-43301.
- Применен патч SPARK-44247.
- Применен патч SPARK-46718.
- Применен патч SPARK-47531.

ADQM Spark connector обновлен до версии 1.0.0-3.5.4_arenadata1

ADB Spark connector обновлен до версии 1.0.5-3.5.4_arenadata1

Обновления ADB Spark connector

- Трансформация построчного итератора JDBC `ResultSet` в `Seq` перемещена в метод `JdbcService.extractData()` для правильного закрытия ресурсов.
- Добавлена поддержка режима считывания `gp_parallel_retrieve_cursor`.
- Добавлена поддержка aggregate pushdown.

- В `BatchReader` добавлена поддержка `SupportsPushDownTopN`, `SupportsPushDownOffset` и `SupportsPushDownLimit`.

Livy обновлен до версии 0.8.0_arenadata1

Hive обновлен до версии 4.0.1_arenadata1

Обновления Hive

- [Bugfix] SQL-тип параметров БД, таблицы и раздела в `removeTxnComponents` теперь указывается явно.
- [Bugfix] Добавлено сообщение об ошибке `sanitize` во время обновления очереди на сжатие.
- [Bugfix] Генерация утилитарного метода `utf8Sanitize` и его использование в `CompactionTxnHandler`.

Tez обновлен до версии 0.10.4_arenadata1

Новые функции

Межсервисные зависимости были устранены, появилась возможность устанавливать несколько кластеров с необходимыми сервисами и доступностью ручной интеграции без ограничений на обязательные сервисы

Бандл содержит в себе расширение для интеграции Flink с Kafka — `flink-kafka-connector` 3.4.0

PyFlink теперь доступен "из коробки"

В HUE теперь доступен преднастроенный интерпретатор Trino

Добавлена возможность скачать клиентские конфигурации из кластера для сервисов HDFS, YARN, Hive, Solr, Core, HBase, Ozone, Flink, Impala и Spark3

Добавлена возможность автоматически устанавливать Java из репозитория Arenadata

Добавлена поддержка HA для Impala State Store

Добавлена поддержка HA для Impala Catalog

Добавлен плагин Ranger для Trino

Расширены возможность управления Trino. В конфигурацию добавлены следующие параметры:

- настройки логирования;
- контроль бесперебойной работы Trino с предустановленными значениями для текущего хранилища (HDFS/Ozone);
- настройки свойств сессии;
- настройки групп ресурсов.

Для Kyuubi добавлена возможность генерировать SQL Lineage и интегрироваться с Open Metadata при использовании движка SparkSQL

Trino теперь поддерживает работу с Ozone через протокол `ofs`

Добавлена возможность выполнять rolling restart для HDFS, ZooKeeper, YARN и HBase

Добавлена поддержка имперсонации для выполнения действий в SSM

Реализована возможность настройки Ozone Topology awareness

Добавлена возможность настройки логирования Kyuubi через ADCM

Добавлена возможность настройки LDAP-аутентификации для Trino через ADCM

Улучшения

Действия ***Add***, ***Remove***, ***Shrink***, ***Expand*** и ***Move*** были заменены действием ***Add/Remove components*** для сервисов Kyuubi, Airflow, ADPG, Solr, Impala, SSM, Zeppelin, Zookeeper, Flink, HBase и Ozone

Все параметры настройки Trino теперь явно отображены в ADCM

Действие ***Manage SSL*** теперь также управляет состоянием сервиса SSM

Trino выполняет предустановку значения параметра `discovery.uri` при использовании хостпровайдера SSH

Улучшена работа Flink в кластере с YARN

Обновление веб-ссылок было добавлено в компонентные действия ***Start*** и ***Restart***

При добавлении/удалении сервисов/компонентов появилась возможность отключить постпроверку для более быстрого исполнения действия

Kyuubi теперь поддерживает движок Trino

Добавлены подсказки в действия ***Add/modify node-to-labels mapping*** и ***Add/modify node labels***

Добавлена возможность редактирования */etc/security/limits.d/om.conf* для сервиса Ozone из ADCM

В группу параметров ***flink-conf.yaml*** добавлен параметр `rest.port`

В действие ***Manage SSL*** для HDFS добавлена задача `Check`

ADB Spark connector теперь принимает во внимание порядок предоставленных пользователем столбцов схемы в записи ADB

Бандл теперь управляет созданием директорий из конфигурационных параметров для сервисов:

- Trino — *node.data-dir*
- YARN — *yarn.nodemanager.local-dirs*, *yarn.nodemanager.log-dirs*, *mapreduce.cluster.local.dir*, *yarn.timeline-service.leveldb-timeline-store.path*, *yarn.nodemanager.recovery.dir*, *ranger.plugin.yarn.policy.cache.dir*, *ranger-yarn-audit.xml/xasecure.audit.destination.solr.batch.filespool.dir*
- HDFS — *dfs.namenode.name.dir*, *dfs.datanode.data.dir*, *dfs.journalnode.edits.dir*, *dfs.namenode.checkpoint.dir*, *ranger.plugin.hdfs.policy.cache.dir*, *ranger-hdfs-audit.xml/xasecure.audit.destination.solr.batch.filespool.dir*

Также учитывается последующее изменение конфигураций и топологии кластера

Исправленные баги

Если компоненты Trino имели кастомные рабочие директории, `systemd` постоянно перезапускал их

Не изменялись настройки SSL для каталога Hive в Trino

Spark3: веб-ссылки не обновлялись после удаления компонентов Livy и Connect

Zeppelin: запросы в интерпретаторе Spark завершались с ошибкой, потому что Zeppelin не мог создать директорию

Файл *impalarc* отсутствовал на хостах, на которые был расширен Impala Client

Файл *log4j2-repl.xml* для Kyuubi имел неверный шаблон

Установка standalone-версии Hive Metastore завершалась с ошибкой

Hive: веб-ссылки не обновлялись после удаления компонента Hive TezUI

Ошибка при удалении выведенного из строя (decommissioned) компонента YARN NodeManager с хоста, находящегося в режиме обслуживания

Плагин Spark AuthZ не создавал логи аудита

Spark3 Livy не работал с Ozone

Неверный парсинг аргументов для флага `-q` в `impala-shell`

ADB Spark connector: сервис не работал в некоторых случаях при включенном плагине Ranger для Spark

ADB Spark connector: считывание из таблицы не возвращало результатов, если число разделов было явно установлено в значение `1`

ADB Spark connector: метод `DataFrame.count()` завершался с ошибкой, если число разделов было установлено вручную

Прочее

Начиная с релиза 4.0.0 дистрибутив меняет нейминг с Arenadata Hadoop (ADH) на Arenadata Hyperwave (ADH). Тем не менее сохраняется возможность прямого апгрейда для пользователей

Перед обновлением кластера необходимо удалить сервисы Sqoop и Spark2 в силу прекращения их поддержки в дистрибутиве. Также требуется удалить устаревший компонент Spark3 Thrift Server — рекомендуется перейти на Kyuubi

В этом релизе появилась возможность автоматической установки Java из репозиториев Arenadata. Однако Arenadata не является официальным поставщиком JDK и данная функциональность представляется для полностью автоматической установки в системах, где отсутствуют необходимые зависимости. В продуктовых средах рекомендуется использовать JDK от официальных дистрибьюторов

Минимальная версия ADCM — 2.5.0

Реализован рефакторинг пакетов для устранения межсервисных зависимостей и логики генерации настроек. Текущее мажорное обновление может потребовать больше времени для полного обновления пакетов на всех хостах. Для увеличения скорости обновления рекомендуется увеличить количество форков Ansible в конфигурации кластера

3.3.6.2

3.3.6.2.b1

Дата: 31.01.2025

Новые сервисы

Ozone 1.4.1_arenadata1

- Реализована функциональность, связанная с безопасностью: поддержка TLS, Kerberos, интеграция в ADPS посредством плагина Ranger, SPNEGO.
- Реализована интеграция Ozone HttpFS с HUE.
- Реализованы настройки Ozone topology awareness в ADCM.
- Реализована поддержка Ozone write pipeline v2.
- Реализована поддержка S3 Gateway.
- Присутствуют некоторые ограничения, см. Известные проблемы ADH.

Также были включены следующие патчи:

Патчи Ozone

- HDDS-11543: Track OzoneClient object leaks via LeakDetector framework
- HDDS-11574: Ozone client leak in TestS3SDKV1
- HDDS-8005: [disabled] Intermittent failure in TestOmSnapshot.testSnapDiffWithMultipleSSTs/
- HDDS-10148: TestOmSnapshotFsoWithNativeLib should be tagged as native test/
- HDDS-10990: Fix memory leak in native lib/
- HDDS-10560: Link rocksdb lib to Ozone rocksdb tools lib relative path instead of absolute path
- HDDS-3415: Enable TestContainerPlacement#testContainerPlacementCapacity
- HDDS-11347: Add rocks_tools_native lib check in Ozone CLI checknative subcommand
- HDDS-10118: hdds-rocks-native fails to build with Java11+

Trino 468_arenadata1

- Реализована функциональность, связанная с безопасностью: поддержка TLS, Kerberos, SPNEGO.

- Реализована преднастройка внутренних каталогов (Iceberg, Hive).
- Добавлена возможность использовать версию Java, отличную от используемой в остальном кластере (требуется ручная настройка).
- Добавлена возможность управлять управлять каталогами Trino в ADCM.

Core configuration

- Конфигурация Hadoop перемещена в сервис, независимый от HDFS.

Обновления

HUE обновлен до версии 4.11.0_arenadata2

Обновления HUE

- Интерфейс HUE JDBC не поддерживал извлечение метаданных таблицы из ADB.
- Страница с задачами Impala открывалась некорректно при включенном SSL.

Kyuubi обновлен до версии 1.9.2_arenadata2

Обновления Kyuubi

- Добавлена возможность управлять несколькими опциями Java в HiveProcessBuilder.

SSM обновлен до версии 2.0.1

Обновления SSM

- SSM 2.0.0
- SSM 2.0.1

Spark3 обновлен до версии 3.5.2_arenadata2

Обновления Spark3

- Добавлена поддержка `TRUNCATE` для Hive4.
- Добавлена поддержка `DROP` для Hive4.

Новые функции

Реализована SPNEGO-аутентификация для сервиса Impala

Реализована поддержка RedOS 7.3 в Enterprise-версии бандла ADH

Реализована возможность импортировать кластер ADH в ADS для настройки Kafka tiered storage

Улучшения

Действия ***Add***, ***Remove***, ***Shrink***, ***Expand***, ***Move*** заменены одним действием ***Add/Remove components*** для сервисов Kyuubi, Sqoop, Airflow, ADPG, Solr, Impala, SSM, Zeppelin, ZooKeeper, Flink и HBase

Удален отображаемый пароль при активированном Samba Kerberos

Из логов убраны пустые и логически ненужные задачи, связанные с Kerberos при удалении компонентов

Исправлены сервисы, в которых не используется `bigtop-detect-javahome` (ZooKeeper, SSM, Livy и jmxtrans)

Исправлено название действий ***Maintenance/Decommiss DataNodes*** и ***Decommiss/Recommiss NodeManagers*** в HDFS

Улучшено удобство использования параметра ***Enable custom ulimits***

В HUE добавлена возможность управлять *log.conf* через ADCM

В HUE добавлена возможность управлять ***impalad_flags*** для интерпретатора Impala через ADCM

Улучшена работа с файлами **-env.sh* в HBase

Для Solr Добавлена кастомная настройка файлов *ranger-solr-audit.xml*, *ranger-solr-security.xml* и *ranger-solr-policymgr-ssl.xml* в ADCM

Для Impala добавлена возможность управлять шаблонами `llma-site` и `fair-scheduler` через ADCM

Для HUE добавлено создание БД, пользователя и грантов при запуске действия ***Restart*** с активным флагом ***Metastore DB schema init/upgrade***

Добавлена возможность настраивать следующие параметры в группе настроек Hive:

- `hive.server2.support.dynamic.service.discovery`

HDFS не перезапускался после отключения шифрования с помощью действия ***Manage credential encryption***

Исправленные баги

Интерфейс JDBC не поддерживал извлечение метаданных таблиц из ADB/Greenplum в HUE

Страница задач Impala отображалась некорректно при включенном SSL в HUE

Невозможность запускать команды в Hive JDBC после перемещения Kyuubi на новый хост

Невозможность выполнить `TRUNCATE` с Hive4 в Spark3

Невозможность выполнить `DROP` с Hive4 в Spark3

Параметр `desktop.http_port` ничего не делал при настройке HUE

ZooKeeper не выключался после перемещения/удаления с хостов

Соединение JDBC не работало с режимом `THRIFT_HTTP` в Kyuubi

Keystore и truststore не скрывались после активации шифрования

Действия ***Add/Remove*** без изменения распределения компонентов по хостам запускали лишние задачи

ADCM status checker не запускался вместе с хостом

Опция `no password` не применялась в Solr после активации шифрования

Ошибка при запуске действия ***Add Server(s)*** для Solr на пустом хосте со включенным SSL

Действие ***Manage credential encryption*** завершалось с ошибкой, если один из выбранных сервисов не был установлен в кластере

Сервисы Hadoop не имели прав доступа к директории */var/lock/subsys/* во время установки

Прочее

Для Trino необходима Java 23. Требуется провести установку на хосте вручную и добавить путь в конфигурациях кластера (`TRINO_JAVA_HOME`)

Для керберизации Ozone необходимо установить его в уже керберизованный кластер

Минимальная версия ADCM — 2.4.0

Помечены как deprecated и будут удалены в следующих версиях — Spark 2, Sqoop, Spark3 Thrift Server (компонент Spark3)

При апгрейде на данную версию необходимо вручную добавить сервис Core configuration

3.3.6.1

3.3.6.1.b1

Дата: 30.10.2024

Обновления

HDFS обновлен до версии 3.3.6_arenadata1 с патчами, перечисленными ниже.

Патчи Hadoop

- HDFS-17299: HDFS is not rack failure tolerant while creating a new file
- HDFS-17453: IncrementalBlockReport can have race condition with Edit Log Tailer
- YARN-10377: Clicking on queue in Capacity Scheduler legacy ui does not show any applications
- YARN-11107: When NodeLabel is enabled for a YARN cluster, AM blacklist program does not work properly
- HDFS-17024: Potential data race introduced by HDFS-15865
- HDFS-16316: Improve DirectoryScanner: add regular file check related block
- YARN-10975: EntityGroupFSTimelineStore#ActiveLogParser parses already processed files
- YARN-9877: Intermittent TIME_OUT of LogAggregationReport
- HDFS-16676: DatanodeAdminManager$Monitor reports a node as invalid continuously
- YARN-11073: Avoid unnecessary preemption for tiny queues under certain corner cases
- YARN-10855: yarn logs cli fails to retrieve logs if any TFile is corrupt or empty
- YARN-10652: Capacity Scheduler fails to handle user weights for a user that has a "." (dot) in it
- YARN-10442: RM should make sure node label file highly available
- YARN-10448: SLS should set default user to handle SYNTH format
- HDFS-15543: RBF: Write Should allow, when a subcluster is unavailable for RANDOM mount points with fault Tolerance enabled
- HADOOP-18613: Upgrade ZooKeeper to version 3.8.3
- HADOOP-19084: prune dependency exports of hadoop-* modules
- HADOOP-18487: Make protobuf 2.5 an optional runtime dependency
- YARN-11657: Remove protobuf-2.5 as dependency of hadoop-yarn-api
- YARN-8056: Possible NPE when start Application Master

Также версия guava обновлена до 32.0.1-jre

Hive обновлен до версии 4.0.0_arenadata1 с патчами, перечисленными ниже.

Патчи Hive

- HIVE-28211: Restore hive-exec-core jar
- HIVE-28323: Iceberg: Allow reading tables irrespective whether they were created with hive engined enabled or not
- HIVE-28306: Iceberg: Return new scan after applying column project parameter
- HIVE-28291: Fix Partition spec is incorrect in getPartitionsByFilter RPC
- HIVE-28286: Add filtering support for get_table_metas API in Hive metastore
- HIVE-28282: Merging into iceberg table fails with copy on write when values clause has a function call
- HIVE-28278: Iceberg: Stats: IllegalStateException Invalid file: file length 0
- HIVE-28271: DirectSql fails for AlterPartitions
- HIVE-28270: Fix missing partition paths bug on drop_database
- HIVE-28266: Iceberg: select count(*) from data_files metadata tables gives wrong result

- HIVE-28260: CreateTableEvent wrongly skips authorizing DFS_URI for managed table
- HIVE-28225: Iceberg: Delete on entire table fails on COW mode
- HIVE-28202: Incorrect projected column size after ORC upgrade to v1.6.7
- HIVE-28224: Upgrade Orc version in Hive to 1.9.3
- HIVE-28248: Upgrade opencsv to v5.9
- HIVE-27653: Iceberg: Add conflictDetectionFilter to validate concurrently added data and delete files
- HIVE-28364: Iceberg: Upgrade iceberg version to 1.5.2
- HIVE-28119: Iceberg: Allow insert clause with a column list in Merge query not_matched condition
- HIVE-28352: Schematool fails to upgradeSchema on dbType=hive
- HIVE-28369: LLAP proactive eviction fails with NullPointerException
- HIVE-28254: CBO (Calcite Return Path): Multiple DISTINCT leads to wrong results
- HIVE-28196: Preserve column stats when applying UDF upper/lower
- HIVE-28190: Materialized view rebuild lock heart-beating is broken
- HIVE-28166: Iceberg: Truncate on branch operates on the main table
- HIVE-28121: Use direct SQL for transactional altering table parameter
- HIVE-28098: Fails to copy empty column statistics of materialized CTE
- HIVE-28082: HiveAggregateReduceFunctionsRule could generate an inconsistent result
- HIVE-27847: Prevent query Failures on Numeric <→ Timestamp
- HIVE-26018: The result of UNIQUEJOIN on Hive on Tez is inconsistent with that of MR
- HIVE-23772: Relocate calcite-core to prevent NoSuchFiledError

Также исправлена ошибка нахождения `java.sql.Driver: Provider org.apache.calcite.jdbc.Driver` и реализована поддержка Impala

---

Tez обновлен до версии 0.10.3_arenadata1

Сервис Impala обновлен до версии 4.4.0_arenadata2 с патчами, перечисленными ниже.

Патчи Impala

- IMPALA-13272: Analytic function of collections can lead to crash
- IMPALA-13252: Filter update log message prints TUniqueId in non-standard format
- IMPALA-13519: Running queries been cancelled after statestore failover
- IMPALA-12712: INVALIDATE METADATA <table> should set a better createEventId
- IMPALA-13034: Add logs for slow HTTP requests dumping the profile
- IMPALA-13270: Bug when comparing ExprSubstitutionMap.size()
- IMPALA-13138: Never smallify existing StringValue objects, only new ones during DeepCopy
- IMPALA-13107: Invalid TExecPlanFragmentInfo received by executor with instance number as 0
- IMPALA-13129: Hit DCHECK when skipping MIN_MAX runtime filter
- IMPALA-13130: Under heavy load, Impala does not prioritize data stream operations
- IMPALA-13161: impalad crash — impala::DelimitedTextParser<true>::ParseFieldLocations
- IMPALA-13150: Possible buffer overflow in StringVal::CopyFrom()
- IMPALA-13170: InconsistentMetadataFetchException due to database dropped when showing databases
- IMPALA-9441: TestHS2.test_get_schemas is flaky in local catalog mode
- IMPALA-13203: ExprRewriter did not rewrite 'id = 0 OR false' as expected
- IMPALA-13119: CostingSegment.java is initialized with wrong cost
- IMPALA-13077: Equality predicate on partition column and uncorrelated subquery doesn't reduce the cardinality estimate
- IMPALA-12800: Queries with many nested inline views see performance issues with ExprSubstitutionMap
- IMPALA-8042: Better selectivity estimate for BETWEEN

Также реализована поддержка Hadoop 3.3.6, Hive 4.0.0 и Hive Driver 4.0.0

Spark 3 обновлен до версии 3.5.2_arenadata1 с патчами, перечисленными ниже.

Патчи Spark 3

- SPARK-47679: Use `HiveConf.getConfVars` or Hive conf names directly
- SPARK-47468: Exclude `logback` dependency from SBT like Maven
- SPARK-49197: Redact `Spark Command` output in `launcher` module
- SPARK-46037: When Left Join build Left, ShuffledHashJoinExec may result in incorrect results

Также реализована поддержка Hive 4.0.0, в группу ***spark-defaults.conf*** конфигурации сервиса добавлен параметр фильтрации в виде "черного списка" `spark.sql.security.confblacklist` для исключения некоторых свойств, добавлен конфигурируемый путь для Spark Connect, исправлено логирование драйвера

ADQM Spark Connector обновлен до версии 1.0.0_3.5.2

ADB Spark Connector обновлен до версии 1.0.5_3.5.2

Flink обновлен до версии 1.19.1_arenadata1 с поддержкой Hive 4.0.0

HBase обновлен до версии 2.5.10_arenadata1 с поддержкой Hive 4.0.0

hbase-operator-tools обновлен до версии 1.2.0_arenadata3

phoenix-queryserver обновлен до версии 6.0.0_arenadata3

Phoenix обновлен до версии 5.2.0_arenadata1

Iceberg обновлен до версии 1.5.2_arenadata1

SSM обновлен до версии 2.0.0-alpha со следующими улучшениями:

- Представлен новый UI и удалена зависимость от Zeppelin.
- Добавлена поддержка HDFS 3.3.6.
- Оптимизирована стратегия подсчета доступов.

Сервис Kyuubi обновлен до версии 1.9.2_arenadata1 со следующими улучшениями:

- Реализована поддержка Hive 4.0.0.
- Добавлена дополнительная переменная среды KYUUBI_ADDITIONAL_CLASSPATH.
- Исправления AuthZ.
- Обновление версии Spark.
- Предварительный вариант модуля движка Flink теперь поддерживает сборку со Scala 2.13.

Solr обновлен до версии 8.11.3_arenadata1

ZooKeeper обновлен до версии 3.8.4_arenadata1

Sqoop обновлен до версии 1.4.7_arenadata3

ADPG обновлен до версии 16

Новые функции

Добавлена поддержка Ubuntu 22.04.2 LTS

Добавлено новое кластерное действие — Manage Credential Encryption. Оно позволяет шифровать конфиденциальные данные в конфигурационных файлах сервисов HDFS, YARN, Hive, HBase, Spark, Impala, Zeppelin, Kyuubi и Solr

Реализована возможность активации протокола SSL/TLS для межкомпонентной коммуникации внутри сервиса Flink

Реализована поддержка SPNEGO для HUE

Дистрибутив включает в себя следующие расширения и форматы для Flink:

- Расширения Flink Kafka.
- Расширения Flink Iceberg.
- Расширения Flink Hive.
- Формат parquet.
- Формат orc.
- Формат avro.
- Формат protobuf.

В ADCM добавлена настройка логов для Livy

Добавлен параметр-чекбокс ***Custom nginx.conf*** с шаблоном для сервиса Hive

Теперь все параметры HDFS `proxyusers` отображаются явным образом

Улучшения

Тип конфигурационного параметра `mem_limit` в Impala Daemon изменен на `string`

Свойство classpaths было изменено на массив для Flink, Spark, Hive, Kyuubi и Hive

Действия ***Add***, ***Remove***, ***Shrink***, ***Expand*** и ***Move*** заменены на одно действие ***Add/Remove components*** для Spark2, Spark3, Hive и YARN

Теперь Hive использует ADPG в качестве metastore по умолчанию во время установки

`kyuubi.authentication.ldap.binddn` более не является необходимым параметром

Актуализирована конфигурация Flink:

- Следующие параметры были добавлены:
  - `taskmanager.memory.flink.size`
  - `taskmanager.memory.process.size`
  - `jobmanager.memory.flink.size`
  - `jobmanager.memory.process.size`
  - `jobmanager.memory.heap.size`
  - `flink.yarn.appmaster.vcores`
  - `flink.yarn.container.vcores`
- Следующие параметры были убраны:
  - `jobmanager.heap.size`
  - `taskmanager.heap.size`

Добавлены новые секции параметров YARN в ADCM: ***mapred-env.sh***, ***Custom mapred-env.sh***

Параметр YARN `HADOOP_JOB_HISTORYSERVER_OPTS` перемещен из ***yarn-env.sh*** в ***mapred-env.sh***

Действие ***Remove*** для Airflow и Zeppelin было добавлено в community-версию дистрибутива

Добавлен параметр ***Custom yarn-env.sh*** в конфигурацию YARN

Добавлены уведомления для сервисов, которые сообщают пользователям о необходимости перезапуска при изменении топологии зависимого сервиса

Исправленные баги

- Ошибки при подключении внешней базы данных к сервису SSM
- Сервис SSM записывал неверный ID процесса в PID-файл
- Ошибка, связанная с необъявленной переменной при включении плагина Ranger YARN
- Не перезапускался Spark3 Thrift Server
- Kyuubi: действие ***Check*** завершалось с ошибкой, если был включен только REST-протокол
- Ошибка `Install CE` при проверке Spark3 Connect
- Неверные права у бинарного файла container-executor
- Impala: плагин Ranger не активировался для некоторых топологий
- HUE: бандл не изменял значение `spark_history_server_security_enabled`
- Flink: лишние параметры на хосте при выключенном режиме HA
- Неверные правила auth-to-local в работе Spark3 History Server
- Неверная настройка LDAP-аутентификации для Airflow

Прочее

- Мажорная версия SSM обновилась и стала несовместима с предыдущей. Во время обновления все настройки безопасности будут изменены, а хранилище метаданных очищено и мигрировано
- Минимальная совместимая версия ADPS теперь 1.2.0_b1
- Минимальная совместимая версия ADCM теперь 2.2.0

3.2.4.3

3.2.4.3.b1

Дата: 23.07.2024

Новые функции

- Добавлен новый сервис HUE
- Версия Zeppelin обновлена до 0.11.1_arenadata1
- Версия Spark3 обновлена до 3.4.3_arenadata1, и исправлено регулярное выражение для версии Hive
- Версия Kyuubi обновлена до 1.9.0_arenadata1, также был добавлен алиас Impala для движка JDBC
- Версия Impala обновлена до 4.4.0_arenadata1, включая следующие патчи:
  - IMPALA-13035: Querying metadata tables from non-Iceberg tables throws IllegalArgumentException
  - IMPALA-13040: SIGSEGV in QueryState::UpdateFilterFromRemote
- Добавлен плагин Kyuubi AuthZ для Spark3

- Введена поддержка доменного контроллера Samba
- Введена поддержка Apache Iceberg для Spark3
- Введена поддержка Apache Iceberg для Hive
- Введена поддержка Spark3 и PySpark для Zeppelin
- Разработана LDAP-аутентификация для Kyuubi
- Разработана LDAP-аутентификация для Impala

Улучшения

- Вместо ***Enable/Disable SSL*** добавлено новое действие ***Manage SSL***, позволяющее более удобно управлять SSL-шифрованием для сервисов кластера ADH
- Значения, установленные в ***Custom***-параметрах, теперь переписывают существующие, даже если они read-only
- Добавлена возможность создавать дополнительный Livy Server для Spark3
- Добавлена возможность указывать несколько директорий в параметре `scratch_dirs` для Impala
- Теперь сервис Arenadata PostgreSQL нельзя удалить при установленном Airflow2
- Теперь параметр `hadoop.proxyuser` регистрируется при добавлении нового сервиса Kyuubi
- Теперь параметр `hadoop.kms.proxyuser.kyuubi.*` в группе *kms-site.xml* конфигурации Ranger регистрируется для сервиса Kyuubi
- Более нет необходимости устанавливать системную переменную `IMPALA_SHELL_HOME` для запуска Impala shell

Исправленные баги

- Не было возможности изменить веб-порт для Solr
- Не применялись изменения конфигурационного файла log4j для сервиса HBase
- Параметры Kerberos пропадали из файла конфигурации после перезапуска компонента Spark3 History Server
- Действие ***Restart YARN*** не работало при недоступных `local-dirs`
- Ошибки во время установки Impala на Red Hat
- Конфигурация сервиса Impala не применялась при запуске кластера
- Установка значения `true` параметру ***Precheck packages*** приводила к ошибкам обновления

3.2.4.2

3.2.4.2.b2

Дата: 27.04.2024

Улучшения

- Устранена необходимость устанавливать Axiom JDK при использовании Astra Linux
- Добавлена возможность устанавливать пользовательское значение переменной `JAVA_HOME`

Исправленные баги

- При конфигурации Shiro Simple Authentication явно отображались пароли
- Конфликт пакетов при обновлении ADH и установке сервиса SSM

3.2.4.2.b1

Дата: 27.03.2024

Новые функции

- Добавлен новый сервис SSM
- Добавлен новый сервис Kyuubi
- Добавлены JDBC-проверки
- Сервисное действие ***Manage Ranger plugin*** теперь более кастомизируемое и понятное
- ADQM Spark connector теперь включен в бандл ADH
- Добавлен компонент Spark Connect для сервиса Spark3
- Изменены права на директории и файлы в бандле для поддержки установки продукта на файловую систему со значением umask равным `027`
- Возможность управления системными переменными в Hive
- Возможность управления системными переменными в HBase
- Обновлена версия Spark до 3.4.2_arenadata1
- Обновлена версия Hive до 3.1.3_arenadata6
- Обновлена версия ADB Spark connector до 1.0.5-spark-3.4.x

Исправленные баги

- Неверно генерировался файл *hdfs-site.xml* во время действия ***Manage Ranger plugin*** в YARN
- Spark3 не подхватывал настройки SSL после действий ***Remove***/***Install***
- Различные ошибки ADB PySpark connector
- LDAP не работал в Airflow2
- Ошибка удаления Spark2 в ADH 3.2.4
- Ошибка балансировщика HDFS на Astra Linux
- Неправильные пути логов Impala в ADH v3.1.2_arenadata1_b1-2
- HiveServer2 не запускался после обновления ADH до версии 3.2.4 из-за SAN-секции в сертификатах

Прочее

- Сервис SSM находится на стадии технологического превью и не предназначен для промышленного использования. Он все еще находится в разработке и представлен пользователям для ознакомления

Путь к интерпретатору Python для сервисов Spark 2/3 изменен с */opt/python3.10/bin/python3* на */opt/pyspark3-python/bin/python3*. Следует принять это во внимание при установке параметра `PYSPARK_PYTHON`

Для установки Impala на Red Hat необходимо вручную установить пакет `cyrus-sasl-sql`. Необходимость в этом будет исключена в следующей версии

Теперь минимально поддерживаемая версия ADCM — 2.0

3.2.4.1

3.2.4.1.b3

Дата: 27.04.2024

Улучшения

Устранена необходимость устанавливать Axiom JDK при использовании Astra Linux

3.2.4.1.b2

Дата: 16.01.2024

Исправленные баги

Ошибочная конфигурация Spark 2/3 при керберизации

3.2.4.1.b1

Дата: 26.12.2023

Новые функции

Flink обновлен до версии 1.17.1_arenadata1 с патчем FLINK-32976

Tez обновлен до версии 0.10.1_arenadata1

ADB Spark Connector обновлен до версии 1.0.5-spark-3.3.x с улучшением производительности и безопасности, а также исправлением ошибок

Добавлен пакет jdbc-tools версии 1.0

Spark2 обновлен до версии 2.3.2_arenadata2 с патчем SPARK-31644

Hadoop обновлен до версии 3.2.4_arenadata1 со следующими патчами:

- HDFS-14768: Busy DN replica should be consider in live replica check
- HDFS-15186: Erasure Coding: Decommission may generate the parity block's content with all 0 in some case
- HADOOP-17340: TestLdapGroupsMapping failing -string mismatch in exception validation
- HADOOP-15783: TestSFTPFileSystem.testGetModifyTime fails
- YARN-9554: TimelineEntity DAO has java.util.Set interface which JAXB can't handle

Добавлена поддержка zstd в HDFS

Добавлена поддержка PMDK в HDFS

Spark3 обновлен до версии 3.3.2_arenadata1 с патчем SPARK-39910 (внутренняя разработка)

Обновлен Spark3 Livy до версии 0.7.2_arenadata5

Обновлен Sqoop до версии 1.4.7_arenadata2

Обновлен Phoenix Query Server до версии 6.0.0_arenadata2

Обновлен Phoenix до версии 5.1.3_arenadata2

Обновлен HBase operator tools до версии 1.2.0_arenadata2

Обновлен Hive до версии 3.1.3_arenadata5 со следующими патчами:

- HIVE-20344: PrivilegeSynchronizer for SBA might hit AccessControlException
- HIVE-21844: HMS schema Upgrade Script is failing with NPE

Обновлен HBase до версии 2.4.17_arenadata1

Обновлен Airflow до версии 2.6.3

Исключены уязвимости в библиотеке log4j вплоть до версии 2.15

В ADCM добавлены настройки логирования для Solr

В ADCM добавлены настройки логирования для ZooKeeper

Airflow1 удален из бандла

Добавлен компонент Thrift Server для Spark3

Добавлен шаблон изменения Ulimits на странице конфигурации компонентов в ADCM

Добавлена поддержка AstraLinux 1.7 SE "Орел"

Улучшения

Изменены настройки подключения Hive к базе данных Metastore для повышения гибкости

Обновлен драйвер Postgres JDBC, поставляемый в дистрибутиве

Существующие keytab-файлы больше не пересоздаются полностью при установке/расширении сервисов

Исправленные баги

Ошибка при добавлении ролей в *shiro.ini* при настройке Zeppelin

Spark3 Server не запускался после обновления

Прочее

Из-за прекращения поддержки Airflow 1 данный сервис требуется удалить из кластера при обновлении. Вместо него можно установить Airflow 2

Минимальная версия ADPS — 1.1.0

3.1.2.1

3.1.2.1.b2

Дата: 30.11.2023

Исправленные баги

- Проблема реконфигурации при долгом нахождении HDFS в режиме SafeMode
- Ошибка обновления URL-адреса репозитория в некоторых случаях при offline-обновлении
- При обновлении HBase учтены изменения upstream, связанные с Phoenix Query Server
- Ошибки обновления кластера в Community Edition

3.1.2.1.b1

Дата: 20.10.2023

Новые функции

- Flink обновлен до версии 1.16.2
- Добавлены следующие обновления из upstream:
  - HIVE-5312: Let HiveServer2 run simultaneously in HTTP (over thrift) and Binary (normal thrift transport) mode
  - HIVE-20187: Incorrect query results in hive when hive.convert.join.bucket.mapjoin.tez is set to true
  - HIVE-21940: Metastore: Postgres text <-> clob mismatch for PARTITION_PARAMS/PARAM_VALUE
  - HIVE-22914: Make Hive Connection ZK Interactions Easier to Troubleshoot
  - HIVE-19825: HiveServer2 leader selection shall use different zookeeper znode
- Solr обновлен до версии 8.11.2 с исправленными уязвимостями библиотеки log4j
- Sqoop обновлен до версии 1.4.7 с исправленными уязвимостями библиотеки log4j
- HBase обновлен до версии 2.2.7
- Phoenix обновлен до версии 5.13
- Добавлена поддержка Impala
- ADB Spark3 Connector теперь включен в бандл ADH
- Реализован HBase-компонент hbck2
- Представлен Maintanence mode, позволяющий удалить любую ноду из кластера
- Представлено автоматическое управление High Availability для сервисов ADH
- Добавлен компонент SQL Gateway для Flink
- Добавлен PySpark 3 для пользовательской установки
- Добавлена поддержка авторизации через Knox SSO для Zeppelin

Улучшения

- Изменено управление Kerberos (отключение, включение, конфигурация)
- В ADCM добавлены настройки логирования для Spark
- В конфигурацию кластера добавлен новый параметр ***Precheck packages***, который включает/отключает проверку пакетов. По умолчанию проверка пакетов отключена

В ADCM добавлены настройки логирования для Sqoop

Исправленные баги

Сервисы Airflow 2 не запускались после керберизации и перезапуска

Проблемы со статистикой и метаданными в Hive

Конфигурация Hive не происходила без установленного ZooKeeper в Community Edition

Проблема с config groups в Hive

Не происходила проверка наличия Spark 3, если не установлен Spark 2

Spark 3 не удалялся без распределения компонент

Проблема с HDFS-балансировщиком при включенном Kerberos

Прочее

Минимальная версия ADPS — 1.0.5

Минимальная версия ADCM — 2023.10.10.08

2.1.10

2.1.10.b1

Дата: 21.06.2023

Новые функции

Добавлена возможность выбора версии TLS для сервисов ADH

Добавлена поддержка пользовательских интерпретаторов Zeppelin

Добавлен новый компонент Spark History Server для Spark3

Добавлены следующие обновления из upstream:

- HIVE-20192: HS2 with embedded metastore is leaking JDOPersistenceManager objects
- HIVE-20209: Metastore connection fails for first attempt in repl dump
- HIVE-20511: REPL DUMP is leaking metastore connections
- HIVE-20522: HiveFilterSetOpTransposeRule may throw assertion error due to nullability of fields
- HIVE-20627: Concurrent async queries intermittently fail with LockException and cause memory leak
- HIVE-20682: Async query execution can potentially fail if shared sessionHive is closed by master thread
- HIVE-21018: Grouping/distinct on more than 64 columns should be possible
- HIVE-26743: Backport HIVE-24694 to 3.1.x
- HIVE-21206: Bootstrap replication is slow as it opens a lot of metastore connections
- HIVE-22393: HiveStreamingConnection: Exception in beginTransaction causes AbstractRecordWriter to throw NPE, covering up real exception
- HIVE-22841: ThriftHttpServlet#getClientNameFromCookie should handle CookieSigner IllegalArgumentException on invalid cookie signature
- HIVE-24552: Possible HMS connections leak or accumulation in loadDynamicPartitions
- HIVE-25830: Hive::loadPartitionInternal occur connection leak
- HIVE-25268: date_format udf returns wrong results for dates prior to 1900 if the local timezone is other than UTC

- HIVE-25075: Hive::loadPartitionInternal establishes HMS connection for every partition for external tables

Улучшения

Режим высокой доступности для ResourceManager активируется автоматически

Добавлена возможность использования кастомного Hive Metastore для Spark

Добавлена поддержка SSL для Hive Metastore

Добавлено действие ***Remove*** для сервиса MariaDB со статусом `created`

Версия Spark3 обновлена до 3.3.2

На странице сервиса ZooKeeper в ADCM добавлены ссылки на Admin Server endpoints

Улучшено выполнение действия ***Move***

Исправленные баги

Исправлено: при наличии Airflow 2 невозможно удалить сервис Airflow 1, если установка последнего завершилась с ошибкой

Zeppelin: исправлена JDBC-строка подключения для сервиса Hive при включенном режиме высокой доступности

Hive: исправлена временная зона для `date_format()`

Исправлена ошибка `Unable to close` для NiFi hive3streaming

Исправлены ограничения GROUPING/DISTINCT для таблиц Hive с 64 столбцами и более

Hive: исправлена ошибка `AssertionError` для запросов с union

Прочее

ZooKeeper: изменен порт по умолчанию для Admin Server

2.1.8

2.1.8.b3

Дата: 02.03.2023

Новые функции

Airflow2: добавлен режим высокой доступности

Airflow2: добавлена поддержка аутентификации/авторизации с помощью LDAP

Airflow2: добавлена возможность внешней конфигурации брокера

Hive обновлен до версии 3.1.3

Добавлены следующие обновления из upstream:

- HIVE-20007: Hive carries out timestamp computations in UTC
- HIVE-20221: Increase column width for partition_params
- HIVE-20437: Handle schema evolution from float, double and decimal

- HIVE-20833: package.jdo needs to be updated to conform with HIVE-20221 changes
- HIVE-20839: Fix the "Cannot find field" error during dynamically partitioned hash join
- HIVE-21050: Use Parquet LogicalTypes
- HIVE-21215: Read Parquet INT64 timestamp
- HIVE-21837: MapJoin throws an exception when selected column has completely null values
- HIVE-21987: Hive is unable to read Parquet int32 annotated with decimal
- HIVE-22476: Hive datediff function provided inconsistent results when `hive.fetch.task.conversion` is set to none
- HIVE-22540: Vectorization: Decimal64 columns don't work with `VectorizedBatchUtil.makeLikeColumnVector(ColumnVector)`
- HIVE-22589: Add storage support for ProlepticCalendar in ORC, Parquet, and Avro
- HIVE-22648: Upgrade Parquet to 1.11.0
- HIVE-23345: INT64 Parquet timestamps cannot be read into bigint Hive type
- HIVE-24074: Incorrect handling of timestamp in Parquet/Avro when written in certain time zones in versions before Hive 3.x
- HIVE-25054: Upgrade jodd-core due to CVE-2018-21234
- HIVE-25093: `date_format()` UDF is returning output in UTC time zone only
- HIVE-25104: Backward incompatible timestamp serialization in Parquet for certain timezones
- HIVE-25299: Casting timestamp to numeric data types is incorrect for non-UTC timezones
- HIVE-25458: `unix_timestamp()` with string input returns wrong result
- HIVE-25559: `to_unix_timestamp()` udf result is incorrect
- HIVE-25577: `unix_timestamp()` is ignoring the time zone value
- HIVE-26233: Problems reading back PARQUET timestamps above 10000 years
- HIVE-26270: Wrong timestamps when reading Hive 3.1.x Parquet files with the vectorized reader
- HIVE-26320: Incorrect results for IN UDF on Parquet column of CHAR/VARCHAR type
- HIVE-26612: INT64 Parquet timestamps cannot be read into BIGINT Hive type
- HIVE-26658: INT64 Parquet timestamps cannot be mapped to most Hive numeric types
- HIVE-26955: Select query fails when decimal column data type is changed to string/char/varchar in Parquet

Улучшения

Дефолтное значение `hive.server2.transport.mode` изменено на `binary`

Убрана переменная `cluster_version_before_upgrade` из *config.yaml.j2*

HDFS: удалены RW-права из свойства `dfs.permissions.superusergroup`

Spark3: добавлен ADH python пакет в installation routine

Реализована кастомная конфигурация файла *container-executor.cfg*

Исправлены CORS-ошибки в ResourceManager UI2

Исправленные баги

Исправлено: отсутствие повторных попыток для задачи ***Disable balancer*** при сжатии HBase

Исправлено: отсутствие меток времени в ansible-логе в ADCM

Исправлено: настройки для checker-thriftserver не перезаписываются после повторной конфигурации

Исправлено: действие `livy_spark3_move` завершается неожиданным результатом

Исправлено: status checker не запускается для сервиса Spark3 после повторной установки

- Исправлен конфликт checker-thriftserver с другим сервисами
- Исправлено некорректное значение параметра `dfs.datanode.https.address` при выполнении действия ***Disable Kerberos***
- Исправлен процесс остановки NodeName для действия JournalNode expand
- Исправлено: HDFS не останавливается, если включен Kerberos
- Исправлены проблемы с Hive-Main JDBC
- Исправлены разрешения для *.jceks*-файлов
- Исправлено: импорт Sqoop Hive не работает при включенном Ranger-плагине
- Исправлено поведение при остановке NameNode
- Airflow2: исправлена поддержка соединения с внешними базами данных
- Airflow2: исправлена конфигурация Redis
- Airflow2: исправлено использование шаблонов для *airflow.cfg* (*cfg_properties_template*)
- Airflow2: исправлены права для каталога конфигураций Redis
- Airflow2: исправлен список уровней логирования
- Исправлена неудачная проверка в случаях, если TEZ submit создает запись в Timelineserver
- Исправлена ошибка в ZooKeeper *zkCli.sh* при включенном Kerberos
- Исправлено: неудачные проверки Spark после выполнения операции YARN expand
- Исправлены ошибки с отключением Kerberos

Прочее

- ADH MySQL Service: изменено отображаемое имя и версия
- Исправлено описание параметра `name.dirs` в файле *config.yml*
- Исправлено описание параметра `spark.ssl.trustStore`
- Обновлено поведение `hadoop.http.authentication.signature.secret.file`
- Обновлены версии offline-пакетов

2.1.7

2.1.7.b1

Дата: 20.12.2022

Новые функции

- Компонент livy-spark3 добавлен в сервис Spark3
- Добавлен токен делегирования Hive
- Добавлена опция ***Apply configs from ADCM*** для всех сервисов

Доступна сборка Flink 1.15.1

Добавлена возможность подключения к Flink JobManager в режиме высокой доступности (high availability)

Улучшения

Выполнена оптимизация по проверке пакетов для процесса установки

Добавлена очистка учетных данных MIT после отключения Kerberos в кластере ADH для следующих сервисов:

- Hive
- YARN
- HDFS

Спрятаны пароли для действий ***Config check***, ***Enable SSL*** для следующих сервисов:

- YARN
- HBase
- HDFS

Выполнен рефакторинг параметров имперсонации Livy

Добавлена возможность настройки имперсонации Livy

Добавлен дополнительный параметр `xasecure.add-hadoop-authorization`

Действие ***Disable HA HiveServer2*** использует команду `deleteall` вместо `rmr`

Выполнена оптимизация действий

Добавлена возможность удаления сервиса в состоянии `created`

Исправленные баги

Исправлена активация режима высокой доступности для YARN ResourceManager

Исправлено: сбой при включении SSL для Airflow2 на AltLinux

Исправлено: Flink TaskManager не запускается в режиме высокой доступности

Исправлено: отсутствие метрик при установке сервиса мониторинга в неправильном порядке

Исправлено: отсутствие действий в Spark3 ***Add Spark3 Livy***, ***Remove Spark3 Livy***

Исправлено: сбой при обновлении Flink с версии 2.1.6 на 2.1.7, если Job Manager и HDFS клиент расположены не на одном узле

Исправлено: отсутствует параметр `container-executor.class` в *yarn-site.xml*

Исправлено: сбой при установке Flink

Исправлено действие ***Manage ranger plugin*** в HBase

Исправлены настройки SSL для следующих сервисов:

- HDFS
- YARN
- Hive

Исправлено: сбой действия ***Enable SSL*** из-за отсутствия ranger-hbase-plugin

- Исправлен сбой при обновлении с версии 2.1.6.b4-1 на 2.1.7.b1-pre_rc
- Исправлено: неверная минимальная версия для обновления
- Исправлено: действия ***Enable HA***/***Disable HA*** приводят к несовпадению файлов конфигурации *hive-site.xml* на серверах
- Исправлен сбой при действии ***Disable HA HiveServer2***
- Исправлена ошибка HBase SASL при подключении к керберизированному ZooKeeper (без *hbase-jaas.conf*)
- Исправлена ошибка при расширении Hive Server 2 на хосте с Hive Metastore
- Исправлено действие ***Add Hive Metastore***
- Исправлено: отсутствует описание для поля ***Reliability Control*** → ***timeout***
- Исправлено: тип Scheduler не передается в фильтр yarn_scheduler_jmx
- Исправлено: отображение паролей в Ansible-логе во время действия ***Enable SSL***
- Исправлено: неверное значение Spnego keytab при расширении Hive Server 2
- Исправлен список компонентов на странице ***Hosts - Components***
- Исправлено: необходимость установки драйвера jdbc-mysql на узлы, где присутствует сервис hive-client при использовании внетренней MariaDB
- Действие ***Enable Ranger plugin*** для Hive приводит к сбою, если в Ranger не определены сервис Yarn и политики
- Исправлено: повторное включение SSL приводит к сбою
- Исправлено: не удается выполнить проверку Spark Thriftserver из-за использования FairScheduler
- Исправлено: реконфигурация YARN приводит к сбою, если включен FairScheduler
- Исправлены ошибки с порядком вывода хостов из эксплуатации (decommissioning)
- Исправлен баг, связанный с невозможностью запуска Flink на YARN

Прочее

- Добавлен флаг `install_flag: true` в действие ***Upgrade***
- Исправлено наименование конфигураций сервисов (HDFS, HBase, Hive)
- Исправлено наименование действий ADH-сервисов (HDFS, HBase, Hive)
- Исправлены наименование и опечатки для действий по установке ADH
- Изменен порядок действий при установке кластера, Kerberos и SSL

2.1.6

2.1.6.b4

Дата: 03.11.2022

Исправленные баги

- Исправлено: неверное значение Spnego keytab при расширении Hive Server 2

2.1.6.b3

Дата: 28.10.2022

Исправленные баги

- Исправлено: ошибка операции расширения (Expand) с принципалом http
- Исправлено: действия ***Disable/Enable SSL/Kerberos*** доступны в любом состоянии
- Исправлено: ошибка Spnego при отключении Kerberos для Hive

2.1.6.b2

Дата: 17.10.2022

Новые функции

- Добавлена возможность кастомизации *krb5.conf* через ADCM

Исправленные баги

- Исправлено: Нарушается работа HDFS из-за отсутствия файлов Spark
- Исправлена проблема с кастомными конфигурациями Hive

2.1.6.b1

Дата: 16.09.2022

Новые функции

- Добавлена поддержка AltLinux 8.4
- Добавлена поддержка настройки *ldap.conf* через ADCM
- Добавлена возможность керберизации кластера с помощью FreeIPA

Улучшения

- Улучшена обработка ошибок с помощью cURL
- Выполнен рефакторинг процесса логирования для Hive

Исправленные баги

- Исправлено: невозможно изменить параметры в *ranger-yarn-policymgr-ssl.xml* (YARN)
- Исправлено: невозможно изменить свойство `hive.metastore.sasl.enabled` (Hive)
- Исправлено имя конфигурации после перезапуска Flink
- Исправлена ошибка отсутствия пользователя Zeppelin при включении Ranger для YARN

Исправлено: невозможно изменить nameservice при установке с ошибкой (HDFS)

Исправлено: обновление кластера всегда переключает состояние на `Installed`

Исправлено: обновление Spark 3 не отключает старые репозитории в Alt Linux

Исправлено: для действия *Reconfigure Kerberos* отсутствует атрибут `allow_to_terminate`

Исправлено: YARN Resource Manager не запускается на Alt 8.4

Исправлено `ClassNotFoundException` при выполнении spark-sql с включенным плагином Ranger

Исправлено несоответствие между настройками безопасности в конфигурациях Spark и Spark3

Прочее

Изменен код проверки в связи с обновлением версии Scala

2.1.4

2.1.4.b11

Дата: 08.08.2022

Новые функции

Добавлена возможность указания внешних nameservices

Добавлена возможность подключения к HiveServer2 в отказоустойчивом режиме

Улучшения

Выполнен рефакторинг состояний компонентов кластера

Изменен порядок действий ***Stop***, ***Start*** и ***Restart*** для сервиса HDFS

Улучшен процесс сбора метрик мониторинга для YARN-очередей

Удален read-only атрибут из свойства `hadoop.security.auth_to_local`

Исправленные баги

Исправлена ошибка статуса керберизации кластера после обновления бандла

Исправлено: переменная Ansible недоступна во время установки HDFS

Параметр `hive.zookeeper.quorum` сбрасывается при выполнении действия Hiveserver2 ***Disable***

Исправлены разрешения для каталога `dfs.datanode.data.dir`

2.1.4.b10

Дата: 16.06.2022

Новые функции

- Добавлена возможность установки компонентов ADH из кастомного Docker registry
- Добавлена опция ***Rewrite current service SSL parameters*** для действия ***Enable SSL***
- Добавлены новые параметры для интерпретатора Zeppelin Hive — для включения SSL и Kerberos
- Реализованы повторные попытки создания принципалов Kerberos
- Пользовательская аутентификация (LDAP/AD) включена для Hive2Server
- Добавлено действие ***Move*** для Spark Livy Server
- Добавлено действие ***Move*** для Spark History Server
- Добавлено действие ***Move*** для Flink Job Manager
- Добавлено действие ***Move*** для Sqoop Metastore
- Добавлено действие ***Move*** для YARN Timeline Server
- Добавлено действие ***Move*** для YARN MapReduce History Server
- Добавлен Ranger-плагин для авторизации Solr
- Добавлена возможность удалять сервисы из кластера
- Добавлена возможность кастомизации конфигурационных файлов через ADCM
- Добавлена поддержка Kerberos REALM

Улучшения

- Соединение Solr для аудита изменено с сервера Solr на узел ZK
- Изменение типа свойства с `string` на `option` при обновлении бандла больше не сбрасывает значение свойства
- Параметры конфигурации SSL по умолчанию изменены с невидимых на доступные только для чтения

Исправленные баги

- Исправлено: повторяющиеся ключи словаря в файле *config.yaml*, которые не позволяли пройти новую проверку YAML в ADCM
- Исправлено: ошибка установки Zeppelin без Hive
- Исправлено: пользователи могли изменять в сервисах некоторые параметры, связанные с Kerberos, доступные только для чтения
- Исправлены сбои задач при включении свойства ***GPU on YARN***
- Исправлена ошибка с применением действия ***Remove service*** к Spark3
- Исправлена ошибка с некорректным значением в *interpereter.json* в Zeppelin при отключенном SSL
- Исправлено: после включения SSL политики не работали в Ranger
- Исправлено: формат Container DN при использовании ***Enable Kerberos***
- Исправлена ошибка с некорректным сохранением значения `HBASE_MASTER_OPTS` в HBase
- Исправлено: YARN не удается скопировать *container-executor.cfg*

- Исправлено: статус задачи и результат не совпадали при удалении необязательных (опциональных) настроек в конфигурации сервиса ZooKeeper
- Исправлено: действие ***Check***, применяемое к Flink, завершалось неудачно, если хосты в ADCM имели заглавные буквы
- Исправлено: сервисы не собирали политики Ranger в режиме SSL
- Исправлено: применение действия ***Remove internal database*** фактически удаляло сам сервис

Прочее

- Добавлен фиксированный репозиторий плагинов Ranger Solr для ADH 2.1.4
- Порядок обновления бандлов изменен с частного на общий
- Реализованы зависимости между компонентами и сервисами на уровне ADCM
- Плагины Ranger обновлены до версии 1.0.3
- Добавлена возможность загрузки offline-пакетов ADH из исходного каталога Arenadata в прокси-репозиторий клиента

2.1.4.b9

Дата: 31.03.2022

Новые функции

- Аутентификация Kerberos включена для веб-интерфейса
- Добавлена поддержка SSL для Ranger-плагинов
- Добавлена поддержка SSL для Flink
- Добавлена поддержка SSL для Sqoop
- Операция отката доступна при сбое процесса керберизации
- Добавлена поддержка SSL для Zeppelin
- Добавлена поддержка SSL для Airflow
- Добавлена поддержка SSL для Spark
- Добавлена поддержка SSL для Solr
- Добавлена поддержка SSL для Hive
- Добавлена поддержка SSL для HBase
- Добавлена поддержка SSL для YARN
- Добавлена возможность настройки SSL в кластерах Hyperwave
- Добавлена поддержка SSL для HDFS

Улучшения

- Блок настроек ***Custom hive-site.xml*** размещается после блока ***hive-site.xml*** в настройках конфигурации
- Ссылки на NameNodes и HttpFS перемещены в начало списка веб-ссылок HDFS

- Порядок действий по остановке кластера изменен на обратный
- Действие ***Reconfig and restart*** заменено действием ***Restart***, которое выполняет три операции: останавливает сервис, применяет параметры конфигурации и запускает сервис
- Возможность выполнения действия ***resourcemanager_enable_ha*** без изменения `hc_map` отключена для Resource Manager
- Возможность выполнения действия ***resourcemanager_expand*** без включения высокой доступности (HA) отключена для Resource Manager, так как он не работает без включения режима высокой доступности

Исправленные баги

- Исправлено: параметры из конфигурационного файла *httpfs-site.xml* не применялись к сервису HttpFS
- Исправлено: SQL-запросы, запущенные из Spark3 или Spark2, работали некорректно при включенном плагине Ranger Hive
- Исправлено: параметры `keystore/truststore`, указанные в настройках сервиса, не переопределяли настройки кластера по умолчанию при per-service установке
- Исправлено: Phoenix Query Server не мог работать в thin-режиме в керберизированной среде
- Исправлено: ошибка запуска ***spark-thrift-server-checker*** после включения и отключения Kerberos
- Исправлено: ошибка сохранения параметров конфигурации Flink после установки в керберизированной среде
- Возможность работы с кластерами ADH с поддержкой Kerberos исправлена для операционной системы Windows
- Исправлена ошибка `Cluster Monitoring imported, but not installed` во время установки кластера
- Исправлена ошибка с проверкой доступа к схеме `public` базы данных `ranger` на этапе preconfigure-check
- Исправлены ошибки установки кластера на этапе ***HBase check***
- Исправлено: логи приложений были недоступны в устаревшем интерфейсе Resource Manager в кластерах с поддержкой Kerberos
- Исправлено: сбой керберизации в AD, если были установлены два экземпляра Ranger Admin

Прочее

- Добавлена настройка параметров SSL перед включением SSL в автотестах
- Веб-ссылки переписаны для поддержки изменения схемы http/https

2.1.4.b6

Дата: 21.12.2021

Улучшения

- Выполнен рефакторинг базы данных для проверок Hive Metastore

Исправленные баги

- Исправлена ошибка с установкой параметра `kdc_type` в `null` после обновления кластера
- Исправлена ошибка, возникавшая в керберизированном кластере при запуске MapReduce-задач не под пользователем `yarn`
- Исправлено: подключенные, но не установленные сервисы вызывали ошибки при установке других сервисов

2.1.4.b5

Новые функции

- Режим HTTP добавлен для HiveServer2
- Добавлена авторизация AD/LDAP/SIMPLE для Zeppelin
- Компонент HBase REST Server добавлен для HBase
- Реализована возможность использования Active Directory в качестве хранилища Kerberos

Улучшения

- Добавлена возможность установить Kerberos-принципал для запуска Spark-задач через YARN. До этого Spark всегда запускал задачи с использованием принципала `yarn`

Исправленные баги

- Исправлена ошибка с расширением DataNodes с помощью действия ***Add DataNode*** в керберизированной среде
- Исправлена ошибка работы интерпретатора Livy в керберизированной среде
- Исправлена ошибка работы интерпретатора Hive в керберизированной среде
- Исправлена ошибка с проверками Zeppelin после активации Kerberos
- Исправлена ошибка с удалением компонента мониторинга jmxtrans
- Исправлено: действие ***Enable Resource Manager HA*** не выполнялось в кластерах с включенным плагином Kerberos и Ranger
- Исправлена ошибка установки Airflow в керберизированной среде
- Исправлена ошибка с включением Kerberos после его отключения (включение → отключение → включение)

Прочее

- Добавлено полное тестирование стека для использования корпоративной лицензии RedHat 7.9 в ADH

2.1.4.b4

Дата: 01.11.2021

Новые функции

- Реализовано действие ***Reinstall status-checker***. Запускает скрипты деплоя *status-checker* для сервисов, а также для контейнеров Docker

Улучшения

- Изменена проверка Solr: вместо списков сравнивается количество живых Nodes
- Увеличено количество тайм-аутов/повторных попыток для проверки Zeppelin

Исправленные баги

- Исправлено: ошибка обновления версии бандла
- Исправлено: Ranger-плагин работал некорректно при использовании определенных символов в имени кластера
- Исправлено: ошибка с непостоянным состоянием обслуживания DataNodes после обновления ADH с 2.1.3 до 2.1.4
- Исправлено: Keytab-разрешения изменялись при определенных действиях
- Исправлено: ошибка при per-service установке в керберизированных кластерах ADH
- Исправлено: ошибка парсинга списка контейнеров при использовании Docker status checker в Airflow

Прочее

- Исправлена ошибка с тестом размера heap
- Исправлена несовместимость с текущей версией ADCM для разработчиков
- Изменена тестовая логика per-service установки в керберизированных кластерах ADH: перед установкой каждого сервиса необходимо добавить сервис в кластер и добавить компоненты сервиса на хосты (вместо добавления всех компонентов на все хосты)

2.1.4.b3

Дата: 30.09.2021

Новые функции

- Интеграция MIT Kerberos реализована в ADCM
- Реализована возможность добавления кастомного порта для Kerberos Server
- Интегрированы Ranger-плагин и YARN с поддержкой Kerberos
- Интегрированы Ranger-плагин и Hive с поддержкой Kerberos
- Интегрированы Ranger-плагин и HBase с поддержкой Kerberos
- Интегрированы Ranger-плагин и HDFS с поддержкой Kerberos
- Ranger-плагин теперь работает в сервисах с поддержкой Kerberos
- Добавлен параметр разделения памяти (split) для сервиса Hive: параметры управления ресурсами можно настроить отдельно для HiveMetastore и HiveServer2
- Добавлена опция редактирования размера памяти для компонентов Flink
- Добавлена опция редактирования размера памяти для компонентов Solr
- Добавлена опция редактирования размера памяти для компонентов Sqoop
- Добавлена опция редактирования размера памяти для компонентов Spark
- Добавлена опция редактирования размера памяти для компонентов Zeppelin
- Добавлена опция редактирования размера памяти для компонентов HBase
- Добавлена опция редактирования размера памяти для компонентов YARN
- Действия ***Add/Remove*** добавлены для YARN Timeline server

Действия ***Add/Remove*** добавлены для Sqoop Metastore

Добавлена опция редактирования размера памяти для компонентов HDFS

Добавлена опция управления памятью ADH

Действия ***Add/Remove*** добавлены для Flink Job Manager

Действия ***Add/Remove*** добавлены для Spark Thrift Server

Действия ***Add/Remove*** добавлены для Spark Livy Server

Действия ***Add/Remove*** добавлены для Spark History Server

Действия ***Add/Remove*** добавлены для Hive Tez UI

Действие ***Move*** добавлено для YARN MapReduce History Server

Добавлена реализация Kerberos для ADH в ADCM

Добавлена возможность перемещения любого сервисного компонента на другую Node или удаления его из кластера

Улучшения

Удалена ненужная проверка репозитория/пакетов на этапе установки HDFS для ADH EE

Путь к файлам *docker-status-checker* изменен

Исправленные баги

Исправлена ошибка с задачей `refreshUserToGroupsMappings` в керберизированном кластере

Исправлена ошибка, из-за которой Solr не работал после применения действий хоста

Исправлена ошибка с действием ***Enable Resource Manager HA*** в керберизированной среде

Исправлена ошибка с расширением/сжатием Spark в керберизированной среде

Исправлена ошибка со сжатием Solr в керберизированной среде

Исправлена ошибка с расширением YARN Node Manager в керберизированной среде

Исправлена ошибка `Keytab does not exist: $user.home/hadoop.keytab`

Исправлена ошибка с расширением/сжатием Flink Server

Исправлена ошибка с доступностью порта 6123 Flink JobManager Server

Исправлена ошибка с расширением Sqoop в керберизированной среде

Исправлена ошибка с расширением/сжатием сервера YARN

Исправлено: роль Solr пыталась импортировать отсутствующую роль мониторинга даже если мониторинг-сервис не установлен

Исправлена ошибка с запуском сервисного действия ***Install*** для Solr

Исправлена ошибка с расширением сервера Spark Livy в керберизированной среде

Исправлена ошибка с отключением Spark Thrift Server в керберизированной среде

- Исправлена ошибка резервирования общей памяти Solr
- Исправлена ошибка с керберизацией Solr без добавления сервисов Hadoop
- Исправлена ошибка запуска jmxtrans после перезагрузки хоста
- Исправлен неверный URL-адрес веб-интерфейса Hive Server
- Исправлена ошибка с открытием ссылки на HiveServer2 UI после установки ADH
- Исправлена ошибка с доступностью действий хоста после обновления кластера с 2.1.3.0 до 2.1.4.b2
- Исправлено: сбой установки Enterprise-кластера при per-service действиях
- Исправлено: сбой действия ***Reconfig and restart*** для сервиса мониторинга с установленным Airflow
- Исправлено: процесс Spark ThriftServer не останавливался после удаления контекста Spark

Прочее

- Для ускорения автотестов и процесса разработки проверка пакетов сделана необязательной для определенных сред
- Версии http и реестра перенесены на текущую версию ET-релиза
- Добавлены спецификации для новых действий Spark и YARN MapReduce History Server

2.1.4.b2

Дата: 20.07.2021

Новые функции

- Добавлена возможность использовать внешний MySQL в AirFlow
- Добавлена возможность использовать внешний PostgreSQL в AirFlow
- Действия хоста добавлены для сервиса Spark3. Действия хостов здесь и ниже означают действия, управляемые на уровне хоста
- Действия хоста добавлены для сервиса мониторинга
- Действия хоста добавлены для сервиса Sqoop
- Действия хоста добавлены для сервиса Airflow
- Действия хоста добавлены для сервиса Solr
- Действия хоста добавлены для сервиса Flink
- Действия хоста добавлены для сервиса Zeppelin
- Действия хоста добавлены для сервиса Spark
- Действия хоста добавлены для сервиса MySQL
- Действия хоста добавлены для сервиса Hive
- Действия хоста добавлены для сервиса HBase
- Действия хоста добавлены для сервиса YARN

- Действия хоста добавлены для сервиса HDFS

Улучшения

- Действие Sqoop ***Check*** изменено в соответствии с новыми переменными внешней базы данных Hive
- Действия хоста переименованы
- Удален файл *solr-tools.jar* из подмодуля Solr в бандле, так как он вызывал ошибки в CI

Исправленные баги

- Исправлена ошибка с offline-установкой в операционной системе RH 7.9
- Исправлена ошибка с применением действия ***Reconfig and restart*** к сервису мониторинга
- Исправлена ошибка с установкой MySQL на хост, с которого этот сервис был удален ранее
- Исправлена ошибка с метрикой YARN `CPU utilization` после установки кластера ADH
- Исправлена ошибка с действием ***Maintenance DataNode***, возникавшая из-за некорректного содержимого файла *dfs.hosts* (если ранее другой DataNode был переведен в состояние обслуживания)

Прочее

- Для отладки возможных проблем через Allure-отчеты реализован сбор логов для сервиса Airflow
- Исправлено неверное описание в автотесте, реализующем миграцию на внешнюю базу данных MySQL
- Изменены спецификации для тестирования действий хоста
- Добавлены тесты для действий хоста
- Добавлены спецификации и автотесты для сценариев сжатия ADH

2.1.4.b1

Дата: 22.06.2021

Новые функции

- Добавлена возможность определять кастомные переменные для среды HBase
- Добавлено действие по удалению MySQL из кластера ADH
- Добавлена возможность использовать внешний PostgreSQL в Hive Metastore
- Добавлена возможность изменения значений `host:port` Hive Metastore
- Добавлена возможность настройки Java Heap для HiveServer2
- Добавлена возможность добавлять/изменять/удалять параметры конфигурации из файла *httpfs-site.xml* через ADCM
- Добавлены стартовые проверки для JournalNodes и NameNodes
- Spark 3.1.1 реализован для ADH 2.X
- Добавлена возможность offline-установки для ADH

Улучшения

- Действие ***Check*** улучшено для Sqoop
- Изменен процесс сборки Solr: вместо внешних репозиториев Maven используются репозитории Arenadata
- Реализована возможность использования Docker Registry из репозитория Arenadata
- Для установки сервисов (например, Airflow) без DNS реализовано разрешение имен хостов в Docker-контейнерах
- Реализована установка Airflow без DNS
- Реализована проверка/ожидание принадлежности DN к кластеру от самого DN
- Spark History Server стал обязательным
- Проведен рефакторинг сервиса ZooKeeper

Исправленные баги

- Исправлено: сбой установки Hive при добавлении компонента Tez без TezUI
- Исправлено: повторяющийся ключ `type: list` удален из файла конфигурации бандла
- Исправлено: YARN-приложения не могли запускать задачи после включения Ranger-плагина
- Исправлена ошибка с действием ***Enable Resource manager HA*** в ADH Community Edition
- Исправлена ошибка установки Sqoop после установки Ranger
- Исправлены проблемы с HBase-логированием

Прочее

- Образы Docker в спецификациях пакетов изменены в соответствии с новым соглашением об именовании
- Пакеты для релиза 2.1.4 загружены в репозиторий Google
- Исправлен сбор логов для HttpFS во время автотестов
- Создан репозиторий для версии продукта 2.1.4
- Добавлена возможность обновления бандла без пересборки пакетов
- Ненужные файлы удалены из архива сборки пакета
- Для решения проблемы ansible с `lint-issue` добавлены опции pipefail для shell-задач
- Добавлены тесты для проверки интеграции ADH и Ranger
- Создана сборка ADH 2.1.3.1

2.1.3

2.1.3.0

Дата: 14.01.2021

Новые функции

- Добавлены действия ***Remove/Add Hive Tez***
- Добавлены действия ***Add diamond*** и ***Remove diamond***

Улучшения

- Переработана логика расширения YARN Resource Manager
- Логика проверки для клиента Spark изменена с `parallel` на `in series`
- Интеграция с ADPS: новый бандл ADPS, содержащий Ranger, необходимо повторно интегрировать с ADH после перемещения Ranger из него

Исправленные баги

- Исправлена ошибка с закрытием задач Hive после завершения проверок
- Проверки Livy временно отключены
- Исправлена ошибка с неверным именем кластера, возникавшая при создании сервиса HDFS через Ranger Admin
- Исправлена ошибка с действием HDFS ***Remove Client***
- Исправлены неудачные проверки Hive CLI после включения плагина Ranger
- Исправлена ошибка с подключением нескольких кластеров к одному ADPS

Прочее

- Репозиторий для плагинов добавлен в бандл релиза
- Пакеты для релиза 2.1.3.0 загружены в репозиторий Google
- Доступна сборка ADH 2.1.3.0
- ADH обновлен до версии 2.1.3.0
- Добавлены новые репозитории для плагинов Ranger
- Отредактированы спецификации обхода ошибки с расширением HBase после расширения HDFS
- Разработаны спецификации по расширению сервисов ADH

2.1.2

2.1.2.5

Дата: 19.11.2020

Новые функции

- Добавлены клиентские компоненты для Flink
- Добавлены клиентские компоненты для HDFS
- Добавлены клиентские компоненты для YARN

Улучшения

- В автотестах увеличено время ожидания для `docker_container`. Также добавлены проверки правильности порядка запуска сервисов в контейнерах
- Изменен номер порта по умолчанию для MySQL в Airflow Metastore
- Увеличено пространство для конфигураций Hadoop (например, */etc/hadoop/conf/*, */etc/hive/conf* и так далее) в образах Docker с Airflow

Исправленные баги

- Исправлено состояние гонки в проверках Sqoop (часть 2 — с несколькими клиентами)
- Исправлена ошибка установки кластера, возникавшая при установке MySQL
- Исправлена ошибка установки кластера, возникавшая после проверки установки Spark

Прочее

- Пакеты релиза 2.1.2.5 загружены в репозиторий Google
- Добавлена сборка offline-пакета для ADH в репозиторий ADH
- ADH обновлен до версии 2.1.2.5
- Добавлены тесты для клиентов YARN/HDFS
- Добавлены спецификации для автотестов для клиентов YARN/HDFS
- Внесены изменения в тесты Hadoop, связанные с загрузкой бандлов
- Созданы автотесты для Airflow
- В автотестах все обращения к файлам стали независимыми в текущей рабочей директории
- Dev-репозиторий для ADH 2.1.2.5 инициализирован

2.1.2.4

Дата: 18.09.2020

Исправленные баги

- Исправлены неправильные репозитории в релизе ADH и ET-пакетах

Прочее

- Исправлена ошибка с обработкой флага `release_version`
- Пакеты скопированы из 2.1.2.3 в 2.1.2.4

2.1.2.3

Дата: 15.09.2020

Новые функции

- Бандлы ADH разделены на версии Community и Enterprise
- Реализован режим высокой доступности (High Availability, HA) для NameNode

Исправленные баги

- Исправлена ошибка, возникавшая на этапе ***Restart NameNodes*** при выполнении действия ***Remove NameNode***
- Исправлена ошибка с проверкой Hive Tez на нескольких хостах
- Исправлена ошибка с включением динамического выделения ресурсов
- Исправлено: ZKFC игнорировал параметр `dfs.namenode.rpc-bind-host` и использовал параметр `dfs.namenode.rpc-address` для привязки к адресу хоста

Прочее

- Пакеты для релиза 2.1.2.3 загружены в репозиторий Google
- Все спецификации, относящиеся к ADH20, перемещены в файл *prj_adh*. Изменен процесс публикация артефактов
- Релизные и develop-репозитории разделены в бандлах

2.1.2.2

Дата: 05.06.2020

Улучшения

- Установка `epel-release` отключена

Исправленные баги

- Исправлено состояние гонки в проверках Sqoop
- Исправлена ошибка с запуском кластерного действия ***Check***

Прочее

- Пакеты релиза 2.1.2.2 загружены в репозиторий Google
- ADH обновлен до версии 2.1.2.2
- Nginx скопирован из репозитория Epel в репозиторий ADH2

2.1.2.1

Дата: 21.05.2020

Новые функции

- Развертывание Sqoop перенесено на ALT Linux

- Развертывание Solr перенесено на ALT Linux
- Развертывание Flink перенесено на ALT Linux
- Создан публичный репозиторий ALT Linux для ZooKeeper 3.4.14
- Развертывание Airflow перенесено на ALT Linux
- Добавлена возможность устанавливать лимиты `nproc` для HBase
- Sqoop добавлен в бандлы ADH
- Созданы пакеты ADH 2.x для ALT Linux
- Solr 8.2.0 добавлен для ADH 2.2

Улучшения

- Проведен рефакторинг процесса развертывания ADH для ALT Linux

Исправленные баги

- Исправлена ошибка ввода/вывода Nodes из эксплуатации через ADCM
- Исправлена ошибка `Port already in use: 10102`, возникшая с HBase `hbck`
- Исправлен порядок общих компонентов
- Исправлено: веб-ссылки в ADCM не обновлялись после сжатия HDFS DataNodes или YARN Node Manager
- Исправлена ошибка, возникавшая с YARN 3.1.2 в ALT Linux при выполнении ansible-задач
- Исправлено отсутствие `org.json.JSONObject` для Sqoop Metastore
- Каталог */var/run/sqoop* создан для Sqoop Metastore
- Добавлена отсутствующая зависимость для пакетов, связанных с Flink
- Исправлена ошибка установки HBase и Solr при использовании внешнего ZooKeeper

Прочее

- Развертывание Airflow отключено (отображаемое в ADCM и только для ALT Linux)
- Изменен релизный публичный репозиторий
- Пакеты для релиза 2.1.2.1 загружены в репозиторий Google
- Изменения для libisal смерджены
- Изменения для bigtop-groovy, bigtop-jsvc, bigtop-tomcat и bigtop-utils смерджены
- Изменения для Bigtop смерджены
- Изменения для Livy смерджены
- Изменения для ZooKeeper смерджены
- Изменения для Zeppelin смерджены

- Изменения для Spark смерджены
- Изменения для Phoenix смерджены
- Изменения для Tez смерджены
- Изменения для Hive смерджены
- Изменения для HBase смерджены
- Изменения для Hadoop смерджены
- Ветки Bigtop для CentOS и ALT linux смерджены вручную
- URL репозитория изменен на 2.1.2
- Проверены автотесты для ADH-сервисов в соответствии с текущей версией стека

2.1.2.0

Дата: 19.02.2020

Новые функции

- Добавлена возможность настройки Hive ACID
- SELinux отключен для всех компонентов во время установки
- Реализована поддержка Flink 1.8.0 для ADCM
- Flink добавлен в бандл ADH
- Улучшена логика действия ***Shrink***
- Поддержка GPU включена для YARN
- Airflow добавлен в бандл ADH
- Ссылка на пользовательский интерфейс Solr добавлена на главную страницу ADCM
- Действия ***Shrink/Expand*** реализованы для HDFS HttpFS
- Реализованы проверки HDFS HttpFS
- Реализован облачный режим для Solr
- Реализовано развертывание Solr
- Solr добавлен в бандл ADH

Улучшения

- Библиотеки Tez установлены на клиентских узлах Hive

Исправленные баги

- Исправлено: невозможность использования Hive с Tez из-за несоответствия конфигурации
- Исправлена ошибка с сохранением конфигураций для HDFS и YARN

- Исправлена ошибка с проверками HBase после установки
- Исправлена ошибка с проверками YARN в режиме высокой доступности

Прочее

- Добавлены тесты/примеры DAG для проверки функциональности Airflow
- Добавлены тесты для проверки работоспособности Solr

2.1.1

2.1.1.1

Дата: 02.12.2019

Прочее

- Преобразование в кастомные единицы измерения systemd

2.1.1.0

Дата: 21.11.2019

Новые функции

- Реализована конфигурация планировщика YARN
- Реализован HDFS-mover
- В пользовательский интерфейс ADCM добавлена кнопка ***Install*** на уровне кластера
- Добавлена возможность определения внешнего ZooKeeper в файле _core-site.xml_
- Добавлена возможность добавления кастомных/расширенных параметров конфигурации в файлы _*-site.xml_
- Реализованы метки (labels) YARN Node
- Реализован HDFS HttpFS
- Реализована функция локального чтения для HDFS (HDFS Short-Circuit Local Reads)
- Реализован балансировщик дисков HDFS (HDFS Disk Balancer)
- Реализован балансировщик HDFS (HDFS Balancer)

Улучшения

- Файлы _*-site.xml_ приведены к общему виду
- Функции assert и fail заменены на _adcm_check_
- Выполнен рефакторинг мониторинга: унифицированы код/панели инструментов, переработаны метрики и так далее
- Переменная имени хоста удалена из определения PID Zeppelin

Дашборд HDFS разделен на панели HDFS и YARN в Grafana

Изменены имена файлов Hadoop PID

Прочее

Выполнено ручное тестирование установки ADH 2.1 согласно документации

2.1.0

2.1.0.0

Дата: 10.10.2019

Новые функции

Добавлена возможность получения статуса для следующих сервисов:

- Zeppelin
- Livy Server
- Thrift Server
- Spark Server
- Zeppelin
- Phoenix
- HBase Thrift
- HBase
- MySQL
- YARN
- HDFS
- Hive

Добавлено управление следующими сервисами:

- Livy Server
- Zeppelin
- Spark Thrift Server
- Spark Server
- Phoenix Server
- HBase Thrift
- HBase Region Server
- HBase Master
- Node Manager
- Resource Manager
- Timeline Service
- WebHCat
- MySQL
- Hive Metastore
- Hive Server
- DataNodes
- Secondary NameNodes
- NameNodes

Добавлены скрипты развертывания для следующих сервисов:

- Livy Server
- Spark Thrift
- Spark Server
- MySQL
- HBase Thrift
- Phoenix Server
- Hive
- HDFS

Реализованы проверки для следующих сервисов:

- Zeppelin
- Spark Thrift Server
- Spark Server
- Livy Server
- Hive
- MySQL
- YARN
- HDFS
- ZooKeeper

Реализовано развертывание следующих сервисов:

- Zeppelin
- Livy Server
- Spark Server
- HBase Thrift
- Phoenix Server
- HBase Region Server
- HBase Master
- WebHCat
- Hive Metastore
- MySQL
- Hive Server
- Hive Client
- Node Manager
- Resource Manager
- Timeline Server
- DataNode
- Secondary NameNode
- NameNode

Доступны следующие сборки:

- Tez 0.9.2
- Livy 0.6
- Pig 0.17
- Flume 1.8

- YARN UI 2.0
- Sqoop 1.4.7

Для следующих сервисов реализованы функции мониторинга:

- Spark
- YARN
- HBase
- HDFS
- Hive

Добавлены необходимые настройки для Hive/Tez

В файл *config.yml* добавлен блок `hbase-site`

Добавлены необходимые настройки для сервисов Hadoop

Добавлены быстрые ссылки на сервисы

HDFS rack awareness реализовано с помощью кастомных скриптов

Сервисы YARN и MapReduce объединены в один

Реализован режим высокой доступности для Resource Manager

Реализованы проверки на вывод из эксплуатации/повторный ввод в эксплуатацию (Decommission/Recommission) для NodeManagers

Реализованы проверки на вывод из эксплуатации/повторный ввод в эксплуатацию (Decommission/Recommission) для DataNodes

Zeppelin обновлен до версии 0.8.1

Поддержка Zeppelin реализована для ADCM

Поддержка Tez UI реализована для ADCM

Реализована возможность добавления нового Node Manager в ADH

Реализована возможность добавления новых DataNodes в кластеры ADH

Поддержка Spark реализована для ADCM

Ranger обновлен до версии 1.1

Добавлена конфигурация кворума ZooKeeper

Улучшения

Роль MySQL добавлена в бандл ADH (в виде сервиса)

Реализовано несколько каталогов конфигурации для Nodes

Включена агрегация журналов YARN

Были пересмотрены роли Spark и Hive

Роль Hadoop разделена на HDFS, YARN, MapReduce

Роль Hadoop для ADCM изменена

Добавлены отдельные роли для Hadoop

- Роль ZooKeeper-сервиса перенесена из ADS-бандла
- Выполнен рефакторинг базовых функций сервиса YARN

Исправленные баги

- Исправлена ошибка с отсутствием параметра `hbase.zookeeper.quorum` после установки

Прочее

- Выполнена предрелизная подготовка для ADH 2.1.0
- Файл *EULA.txt* добавлен в корень бандла
- Репозиторий для пакетов ZooKeeper добавлен в ADH
- Все подмодули ADH-бандла переключены в Master
- Подготовлена документация по выведению из эксплуатации/восстановлению/режиму высокой доступности (Decommission/Recomission/HA)
- Подготовлена документация по развертыванию HBase через ADCM
- Подготовлена документация по развертыванию Spark через ADCM
- Подготовлена документация по развертыванию Hive через ADCM
- Подготовлена документация по развертыванию YARN через ADCM
- Подготовлена документация по развертыванию HDFS через ADCM
- Подготовлена документация для ADH-бандла
- Реализованы автотесты для Spark
- Реализованы автотесты для Hive
- Реализованы автотесты для YARN
- Реализованы автотесты для HDFS
- Подготовлены smoke-тесты для проверки сервиса Livy Server
- Подготовлены smoke-тесты для проверки сервиса Spark Thrift Server
- Подготовлены smoke-тесты для проверки сервиса Livy Server
- Подготовлены smoke-тесты для проверки сервиса Spark Server
- Подготовлены smoke-тесты для проверки сервиса MySQL
- Подготовлены smoke-тесты для проверки сервиса HBase
- Подготовлены smoke-тесты для проверки сервиса Phoenix
- Подготовлены smoke-тесты для проверки сервиса Hive
- Подготовлены smoke-тесты для проверки сервиса HDFS
- Подготовлены последние стабильные сборки для ADH

Поддерживаемые сервисы

Константин Алпашкин, Сергей Тихомиров

Назад к содержанию

В этой статье описывается список сервисов, включенных в текущую версию ADH. Столбец ***Версия*** содержит ссылки на release notes каждого компонента.

Сервис	Компонент	Версия
Apache Airflow2	Broker (Redis)	6.2.7
	Scheduler	2.6.3
	Server	2.6.3
	Worker	2.6.3
Apache Flink	Client	1.20.1_arenadata1
	Job Manager	1.20.1_arenadata1
	Task Manager	1.20.1_arenadata1
	SQL Gateway	1.20.1_arenadata1
Apache HBase	Client	2.6.3_arenadata1
	Master server	2.6.3_arenadata1
	Region server	2.6.3_arenadata1
	Thrift2 server	2.6.3_arenadata1
	Phoenix query server	6.0.0_arenadata4
	REST server	2.6.3_arenadata1
Apache HDFS	Client	3.3.6_arenadata1
	DataNode	3.3.6_arenadata1
	NameNode	3.3.6_arenadata1
	JournalNode	3.3.6_arenadata1
	ZKFC	3.3.6_arenadata1
	HttpFS	3.3.6_arenadata1
Apache Hive	Client	4.0.1_arenadata2
	HiveServer2	4.0.1_arenadata2
	Metastore	4.0.1_arenadata2
	Tez	0.10.4_arenadata1
	TezUI	0.10.4_arenadata1
HUE	Server	4.11.0_arenadata4
Apache Impala	Daemon	4.5.0_arenadata1
	Statestore	4.5.0_arenadata1
	Catalog Service	4.5.0_arenadata1
	Client	4.5.0_arenadata1
Apache Kyuubi	Kyuubi Server	1.10.1_arenadata2
Apache Ozone	Ozone Datanode	2.0.0_arenadata1
	Ozone Manager	2.0.0_arenadata1
	Ozone Recon	2.0.0_arenadata1
	Ozone S3G	2.0.0_arenadata1
	Ozone Storage Container Manager	2.0.0_arenadata1
Apache Solr	Server	8.11.4_arenadata1
Apache Spark3	Client	3.5.4_arenadata2
	History server	3.5.4_arenadata2
	Livy	0.8.0_arenadata1
SSM	Smart Server	2.1.1
	Smart Agent	2.1.1
Trino	Trino Coordinator	476_arenadata1
	Trino Worker	476_arenadata1
Apache YARN	Client	3.3.6
	MapReduce History server	3.3.6
	NodeManager	3.3.6
	ResourceManager	3.3.6
	Timeline server v.1	3.3.6
Apache Zeppelin	Server	0.11.2_arenadata2
Apache ZooKeeper	Server	3.8.4
Arenadata Monitoring	Monitoring	3.2
Arenadata Postgres	Arenadata Postgres	16

Матрица совместимости версий

Сергей Тихомиров

Назад к содержанию

Версия ADH	Минимальная версия ADPS	Максимальная версия ADPS	Минимальная версия ADCM
4.1.0	2.0.0_b1	—	2.7.0
4.0.0	1.2.2	—	2.5.0
3.3.6.2	1.2.1_b1	—	2.4.0
3.3.6.1	1.2.0_b1	—	2.2.0
3.2.4.3	1.1.0_b1	1.1.2_b2	2.0.0
3.2.4.2	1.1.0_b1	1.1.2_b2	2.0.0
3.2.4.1	1.1.0_b1	1.1.2_b2	2023.10.10.08
3.1.2.1	1.0.5_b1	1.0.5_b2	2023.10.10.08
2.1.10	1.0.4_b1	1.0.4_b6	2023.04.05.14
2.1.8	1.0.4_b1	1.0.4_b6	2022.08.08.14
2.1.7	1.0.4_b1	1.0.4_b6	2022.08.08.14

Известные проблемы ADH

Сергей Тихомиров

Назад к содержанию

Сервис/компонент	Проблема	С версии	Fixed	Обходное решение
Ozone	Невозможно активировать Kerberos-аутентификацию после установки	3.3.6.2	—	Установите кластер без сервиса Ozone, активируйте SSL и Kerberos, только затем проводите установку и настройку Ozone
Ozone	Клиенты не работают при активированной безопасной коммуникации между компонентами. Не активируйте `hdds.grpc.tls.enabled`	3.3.6.2	—	—
Trino, Smart Storage Manager	Сервис Trino не поддерживает клиент SSM. Другими словами, Trino не работает при включенной опции ***Enable SmartFileSystem for Hadoop*** (конфигурационный параметр сервиса SSM)	3.3.6.2	—	Не активируйте SmartFileSystem для Hadoop при использовании Trino
Flink и режим высокой доступности YARN	Если YARN работает в режиме высокой доступности (т.е. в кластере YARN имеется несколько компонентов ResourceManager), для корректного взаимодействия Flink с YARN необходимо вручную установить определенные параметры	4.0.0	4.1.0	В качестве временного решения необходимо указать следующие параметры в настройках сервиса Flink (***Clusters → <ADH_cluster> → Services → Flink → Primary configuration → Custom flink-conf.yaml***):

```
flink.yarn.resourcemanager.ha.enabled: True ①
flink.yarn.resourcemanager.ha.rm-ids: <adh_rm-1>,<adh_rm-2>,<adh_rm-3> ②
flink.yarn.resourcemanager.hostname.adh_rm-1: <adh_rm_fqdn-1> ③
flink.yarn.resourcemanager.hostname.adh_rm-2: <adh_rm_fqdn-2>
flink.yarn.resourcemanager.hostname.adh_rm-3: <adh_rm_fqdn-3>
```

① Информирует Flink об активном HA-режиме YARN.

② Список ID компонентов ResourceManager, доступных в кластере YARN.

③ FQDN каждого компонента ResourceManager.

Актуальное значение для параметра `flink.yarn.resourcemanager.ha.rm-ids` можно найти в */etc/hadoop/conf/yarn-site.xml*, например:

```
$ grep "yarn.resourcemanager.ha.rm-ids" /etc/hadoop/conf/yarn-site.xml -A1
```

Пример вывода:

```
<name>yarn.resourcemanager.ha.rm-ids</name>
<value>rm_ka-adh-2,rm_ka-adh-3</value>
```

Сервис/компонент	Проблема	С версии	Fixed	Обходное решение
Impala	Проблема заключается в интеграции Impala со внешним хранилищем объектов. Во время настройки и создания таблицы в локации S3 возникает следующая ошибка: `Error reading from HDFS file: s3://…`	3.3.6.2	4.1.0	Удалите или переместите следующие файлы: */usr/lib/impala/lib/aws-java-sdk-core-1.12.765.jar*, */usr/lib/impala/lib/aws-java-sdk-logs-1.12.765.jar*
Trino, Ozone	Trino не работает с Ozone через протоколы `osf` и `o3sf`. Используйте протокол `s3` (S3-шлюз). Должен быть определен в расположении объектов	3.3.6.2	4.0.0	`ALTER … SET LOCATION 's3://…'`
Impala	При перемещении клиента Impala на другой хост не создаются все необходимые файлы	3.3.6.1	4.0.0	Перезапустите сервис после миграции компонентов
Smart Storage Manager	При активации SSL в кластере с помощью действия Manage SSL сервис продолжает работать через HTTP	3.3.6.1	4.0.0	Проведите ручную настройку согласно руководству