AD.PS

Arenadata Platform Security

Arenadata Platform Security (ADPS) is a combination of two security components: Apache Ranger and Apache Knox. Together, ADPS provides a comprehensive security framework that includes policy-based access control, fine-grained authorization, and secure access to the platform and its services. This helps organizations protect sensitive data and ensure compliance with regulations.

Version CURRENT Language: EN

Contents

1 Get started
- Online installation
  - Install ADCM
  - Prepare hosts
  - Install ADPS cluster
    - Create a cluster
    - Add services
    - Add hosts to a cluster
    - Add components
    - Configure services
    - Configure a cluster
    - Install a cluster
  - Install monitoring
    - Create a cluster
    - Add services
    - Add hosts to a cluster
    - Add components
    - Configure services
    - Install a cluster
    - Integrate with ADPS cluster
- Offline installation
  - Install ADCM
  - Prepare hosts
  - Install Enterprise Tools cluster
    - Create a cluster
    - Add services
    - Add hosts to a cluster
    - Add components
    - Install a cluster
  - Install ADPS cluster
    - Create a cluster
    - Add services
    - Add hosts to a cluster
    - Add components
    - Configure services
    - Configure a cluster
    - Import ET settings
    - Install a cluster
  - Install monitoring
    - Configure Enterprise Tools cluster
    - Integrate with ADPS cluster

2 How to
- Knox
  - Connect to Knox
    - Admin API
    - CLI
  - Knox UI overview
  - Enable SSL on ADPS cluster with Knox
  - Authorization
    - Ranger Knox plugin
    - ACL
  - Logging
  - Configure
    - Topologies
  - Usage examples
    - ADH services
    - ADS services
  - Service management via ADCM
- MariaDB
  - Service management via ADCM
- OpenBao
  - OpenBao UI overview
  - Use command-line interface
  - Service management via ADCM
- Ranger
  - Connect to Ranger
    - REST API
    - KMS REST API
    - Web interface
  - Ranger Admin UI
    - Service Manager
      - Manage services
      - Manage policies
    - Audits
    - Security Zone
    - Settings
      - Manage users
      - Manage groups
      - Manage roles
      - Manage permissions
  - Access management
    - Configure Ranger UserSync
    - Configure authentication with LDAP/Active Directory
  - Configure HA
    - Ranger Admin HA
    - Ranger KMS HA
  - Configure an external database
    - PostgreSQL
    - MySQL
  - Logging
  - Usage examples
    - Configure policies
  - Service management via ADCM
- Solr
  - Ranger Solr plugin
  - Service management via ADCM
- ZooKeeper
  - Service management via ADCM

3 Tutorials
- Cluster management via ADCM
  - Cluster actions
    - Check
    - Manage SSL
    - Manage Kerberos
    - Reinstall status-checker
    - Start
    - Stop
    - Validate configuration
    - Upgrade
  - Service actions
- Security
  - Kerberos
    - MIT Kerberos
      - Configure via ADCM
    - MS Active Directory
      - Configure via ADCM
    - FreeIPA
      - Configure via ADCM
    - Samba
      - Configure via ADCM
  - Password encryption

4 References
- Configuration parameters

5 Planning guide
- Hardware requirements
- Network requirements
- Software requirements
- Custom Java settings

6 Concepts
- ADPS overview
- Services
  - Knox
    - Knox architecture
    - Topology descriptors
  - Ranger
    - Ranger architecture
    - Policy model
    - Plugins
  - OpenBao
    - OpenBao architecture

7 Release notes
- ADPS releases
- Supported services

Get started with Arenadata Platform Security

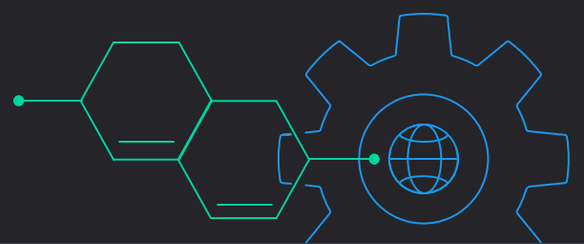
To Table of Contents

This tutorial guides you through the process of installing an Arenadata Platform Security (ADPS) cluster. The easiest way to install it is to use Arenadata Cluster Manager (ADCM) — a separate Arenadata product designed for simple, convenient, and fast software deployment and exploitation.

You can install ADPS via ADCM in two ways.

Online installation

on hosts with access to the Internet



Offline installation

on hosts without access to the Internet

Online installation

To Table of Contents

This topic describes **online** installation of ADPS. It can be applied to the hosts with access to the Internet.

The process of online installation via ADCM includes the following steps.

Step 1.

Install ADCM

Step 2.

Prepare hosts

Step 3.

Install ADPS cluster

Step 4.

Install monitoring

Install ASCM

Author: [ECLIPS](https://eclips.cloud)

Contents

- Step 1. Check prerequisites
  - Software and hardware requirements
  - Check if Docker is installed
  - Install Docker
  - Check if Docker Compose is installed
  - Install Docker Compose
  - Check the Docker version
  - Check the Docker Compose version
- Step 2. Install ASCM
  - Check the Docker container status
  - Check the ASCM port availability
- Step 3. Start ASCM
- Step 4. Check installation
  - Check the Docker container status
  - Check the ASCM port availability
  - Check the ASCM web interface
  - Set the ASCM URL
- Step 5. Update ASCM

Step 1. Check prerequisites

Software and hardware requirements

Check if Docker is installed

```
docker -v
```

Install Docker

```
curl -fsSL https://get.docker.com | bash -s docker
```

Check if Docker Compose is installed

```
docker compose version
```

Install Docker Compose

```
curl -SL https://github.com/docker/compose/releases/download/v2.20.2/docker-compose-linux-x86_64 -o /usr/local/bin/docker-compose
chmod +x /usr/local/bin/docker-compose
```

Check the Docker version

```
docker -v
```

Check the Docker Compose version

```
docker compose version
```

Step 2. Install ASCM

Check the Docker container status

```
docker ps -a | grep ascm
```

Check the ASCM port availability

```
netstat -tunlp | grep 8080
```

Step 3. Start ASCM

Check the Docker container status

```
docker ps -a | grep ascm
```

Check the ASCM port availability

```
netstat -tunlp | grep 8080
```

Check the ASCM web interface

Open the following URL in your browser:

- **IP**: `http://IP:8080`
- **Example**: `http://192.168.1.100:8080`

Set the ASCM URL

1. Click **Settings** in the left navigation pane.
2. Click **System Settings** > **Basic Settings**.
3. Enter the **ASCM URL** in the **Access Address** field.

Step 4. Check installation

Check the Docker container status

```
docker ps -a | grep ascm
```

Check the ASCM port availability

```
netstat -tunlp | grep 8080
```

Check the ASCM web interface

Open the following URL in your browser:

- **IP**: `http://IP:8080`
- **Example**: `http://192.168.1.100:8080`

Set the ASCM URL

1. Click **Settings** in the left navigation pane.
2. Click **System Settings** > **Basic Settings**.
3. Enter the **ASCM URL** in the **Access Address** field.

Step 5. Update ASCM

Check the Docker container status

```
docker ps -a | grep ascm
```

Check the ASCM port availability

```
netstat -tunlp | grep 8080
```

Check the ASCM web interface

Open the following URL in your browser:

- **IP**: `http://IP:8080`
- **Example**: `http://192.168.1.100:8080`

Set the ASCM URL

1. Click **Settings** in the left navigation pane.
2. Click **System Settings** > **Basic Settings**.
3. Enter the **ASCM URL** in the **Access Address** field.

Prepare hosts

To Table of Contents

To manage a cluster and its nodes via ADCM, it is necessary to prepare hosts for a cluster and link these hosts to ADCM. This is the function of ADCM **hostproviders** — special plugins for ADCM, which encapsulate the complex mechanism of interaction between ADCM and the hosts controlled by it.

ADCM supports several types of hostproviders. For more information on how to configure hostproviders and use them to create hosts, refer to the following documentation:

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

CAUTION

When creating ADPS hosts via ADCM, you should specify their FQDNs. The maximum FQDN length is 49 symbols.

Create a cluster

Sergei Tikhomirov, Daria Barysheva

Contents

To Table of Contents

Step 1. Get the Enterprise version of a cluster bundle

Step 2. Upload a cluster bundle to ADCM

Step 3. Create a cluster based on the uploaded bundle

In ADCM a **cluster** means a set of services running on a set of hosts.

Clusters distributions for ADCM are made in the form of so-called bundles. Generally a bundle is a regular archive that includes a description and software logic of cluster deployment and exploitation.

The steps for adding a cluster to ADCM are listed below.

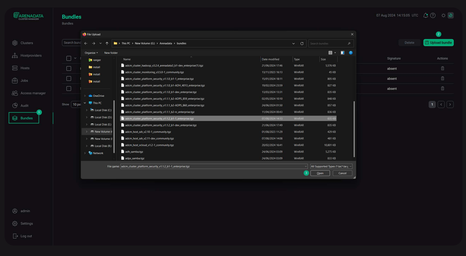
Step 1. Get the Enterprise version of a cluster bundle

Please get a cluster bundle from the Arenadata support team.

Step 2. Upload a cluster bundle to ADCM

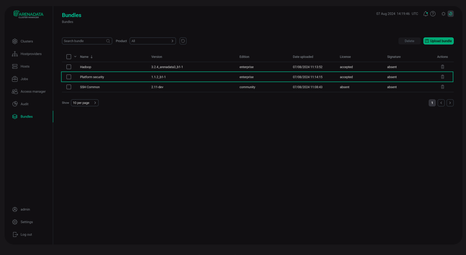
To upload a bundle to ADCM, follow the steps:

1. Select the ***Bundles*** item in the left navigation menu and click ***Upload bundle***.
2. Select a bundle in the Open File Dialog.



Upload a bundle

3. As a result of the performed actions, a bundle is displayed on the ***Bundles*** page.



The result of successful uploading a bundle

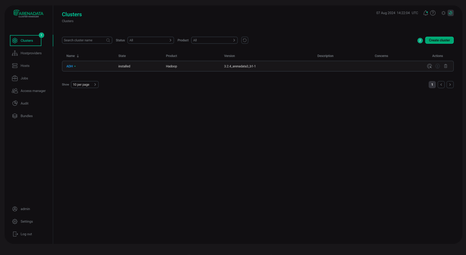
Step 3. Create a cluster based on the uploaded bundle

IMPORTANT

When creating a cluster via the ADCM web interface, you generate a new ADCM cluster instance. It means only adding the cluster to ADCM — at this stage no services are deployed.

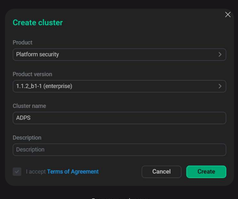
The steps for cluster creation are given below:

1. Select the ***Clusters*** item in the left navigation menu and click ***Create cluster***:



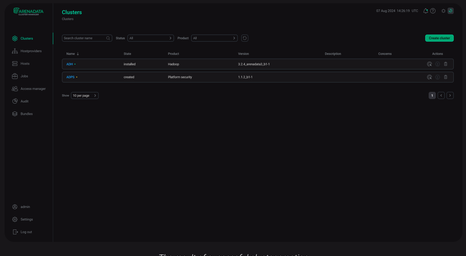
Switch to cluster creation

2. In the opened dialog, describe a new cluster instance:
   - Select an uploaded cluster bundle in the ***Product*** field.
   - Select a bundle version in the ***Product version*** field. Several versions become available in the case of different versions of the same bundle being uploaded.
   - Enter a cluster name in the ***Cluster name*** field.
   - Enter a cluster description in the ***Description*** field if necessary.
   - Set the ***I accept Terms of Agreement*** flag to sign the license agreement. This flag is displayed only for Enterprise bundles. To read the agreement text, refer to the ***Terms of Agreement*** link. You can also sign the license agreement on the ***Bundles*** page by clicking the desired product bundle.
   - Click ***Create***.



Create a cluster

3. As a result, the created cluster instance is displayed on the ***Clusters*** page.



The result of successful cluster creation

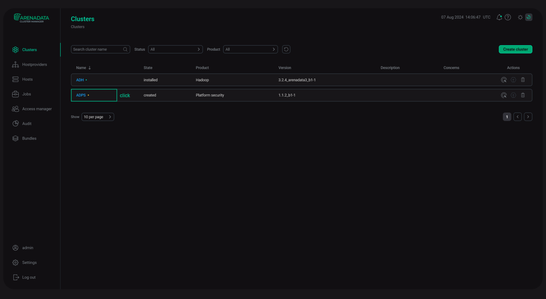
Add services

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

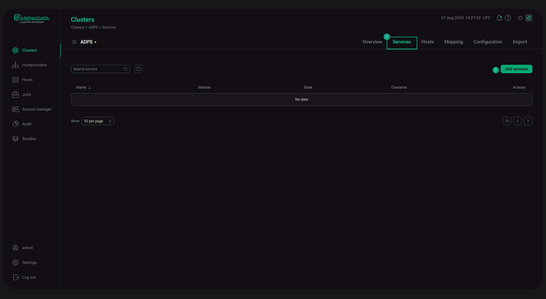
In ADCM a **service** means a software that performs some function. Examples of services for ADPS clusters: Ranger, Knox, MariaDB, etc. The steps for adding services to a cluster are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



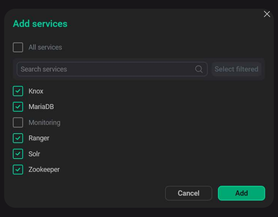
Select a cluster

2. Open the ***Services*** tab on the cluster page and click ***Add service***.



Switch to adding services

3. In the opened dialog, select services that should be added to the cluster and click ***Add***.



Select services

The brief description of available services is listed below.

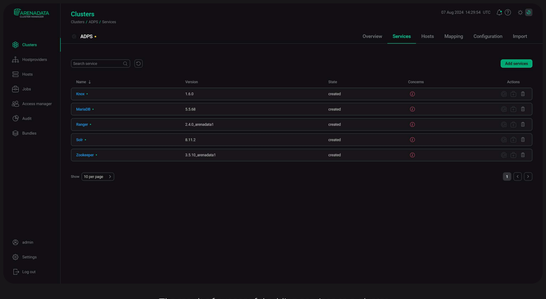
Services that can be added to the ADPS cluster

Service	Purpose
Knox	Reverse proxy service that acts a single access point for HTTP interactions with resources
Maria DB	Maria DB — a relational database created on the base of MySQL and compatible with it. Some MariaDB commands and interfaces are closer to NoSQL, than to SQL. For example, it provides such data storage types as: ColumnStore — for column data storage and distributed architecture support, OQGRAPH — for storing tree and graph structures, etc. Used as a database for Ranger
Monitoring	A service that should be added if monitoring of the ADPS cluster is planned
Ranger	A data security framework used for authorization and policy management
Solr	A search platform based on the Apache Lucene project. Its main features include full-text search, faceted search, highlighting search results, distributed indexing, integration with databases, processing documents with a complex format (Word, PDF, etc.), load-balanced querying, centralized configuration, and others. Used as an audit storage for Ranger
Zookeeper	A centralized coordination service for distributed applications. It is used for failure detection, health monitoring, session management, consensus, etc.

The minimal set of services recommended for ADPS clusters is described below:

- Knox;
- MariaDB;
- Ranger;
- Solr;
- Zookeeper.

4. As a result, the added services are displayed on the ***Services*** tab.



The result of successful adding services to a cluster

NOTE

You can also add services later. The process of adding new services to already running cluster does not differ from installing a service from scratch.

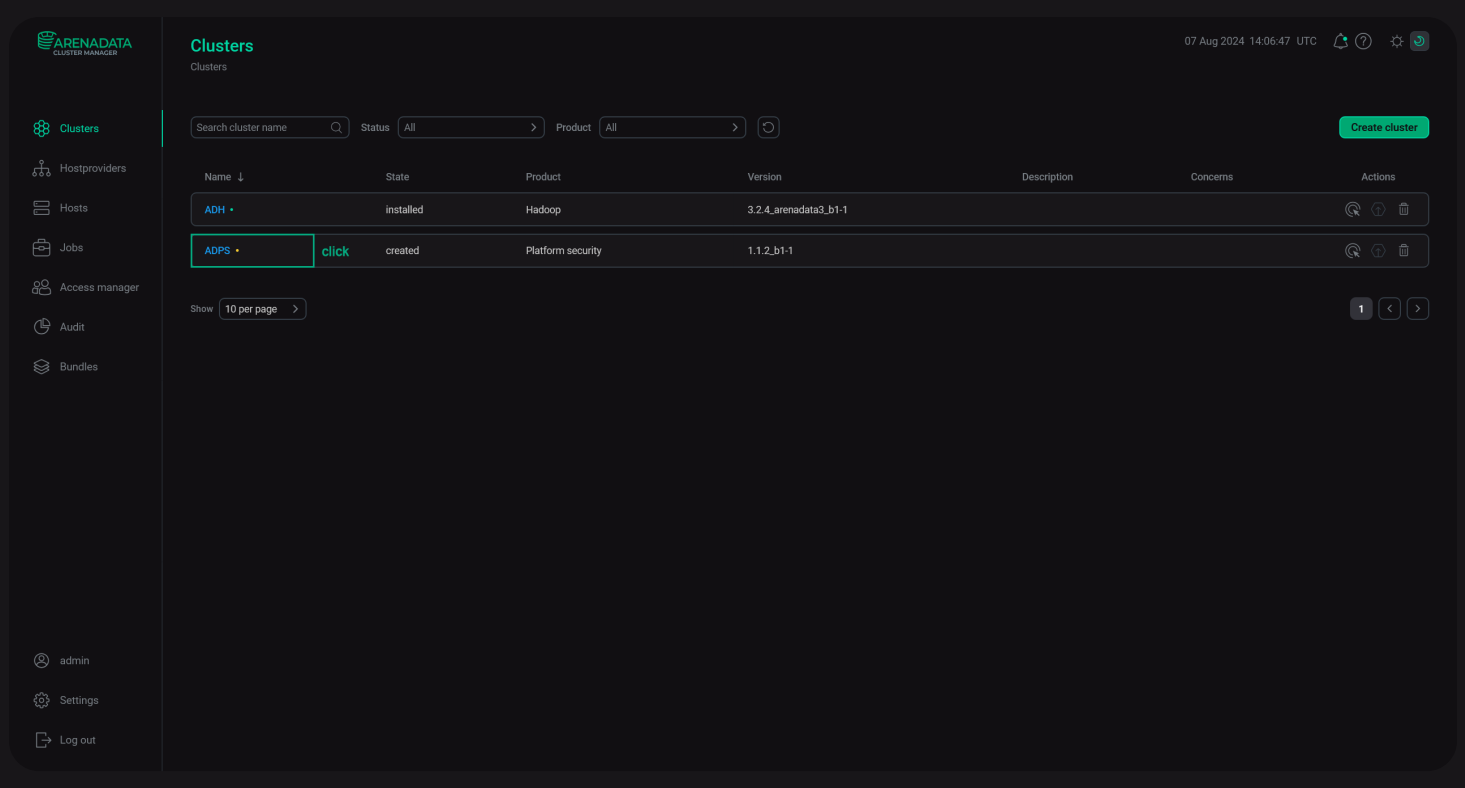
Add hosts to a cluster

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

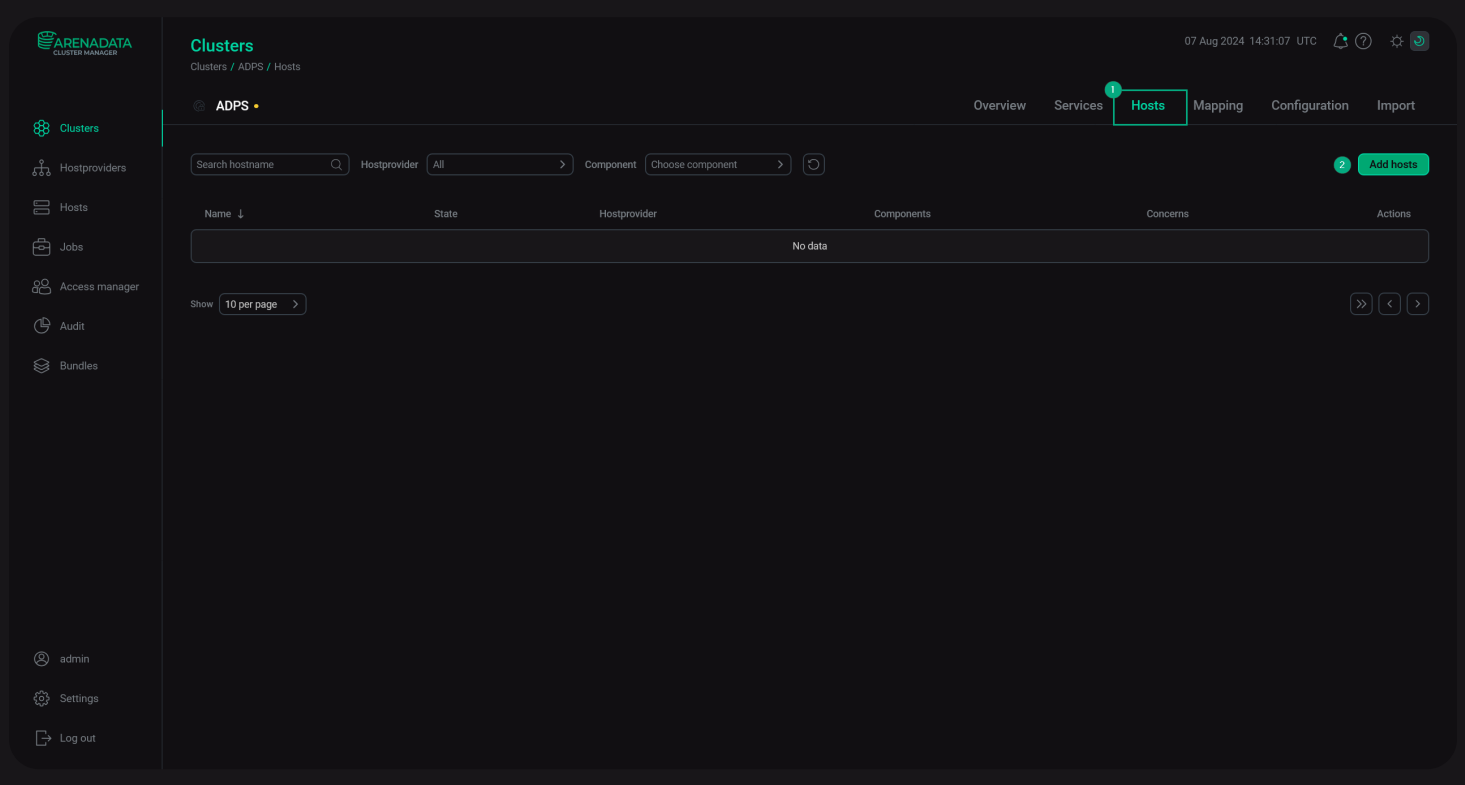
The steps for adding hosts to a cluster are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



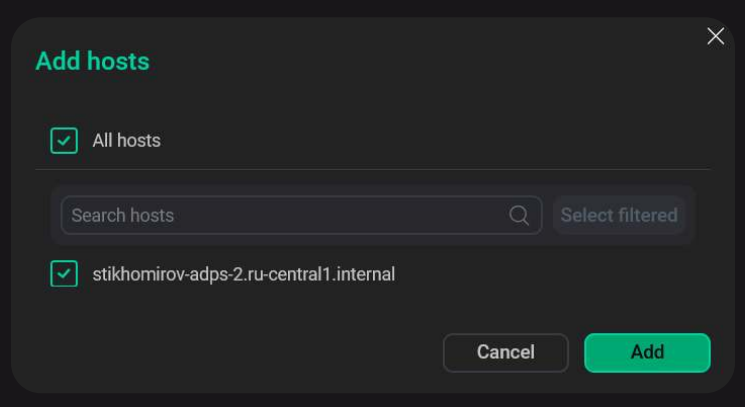
Select a cluster

2. Open the ***Hosts*** tab on the cluster page and click ***Add hosts***.



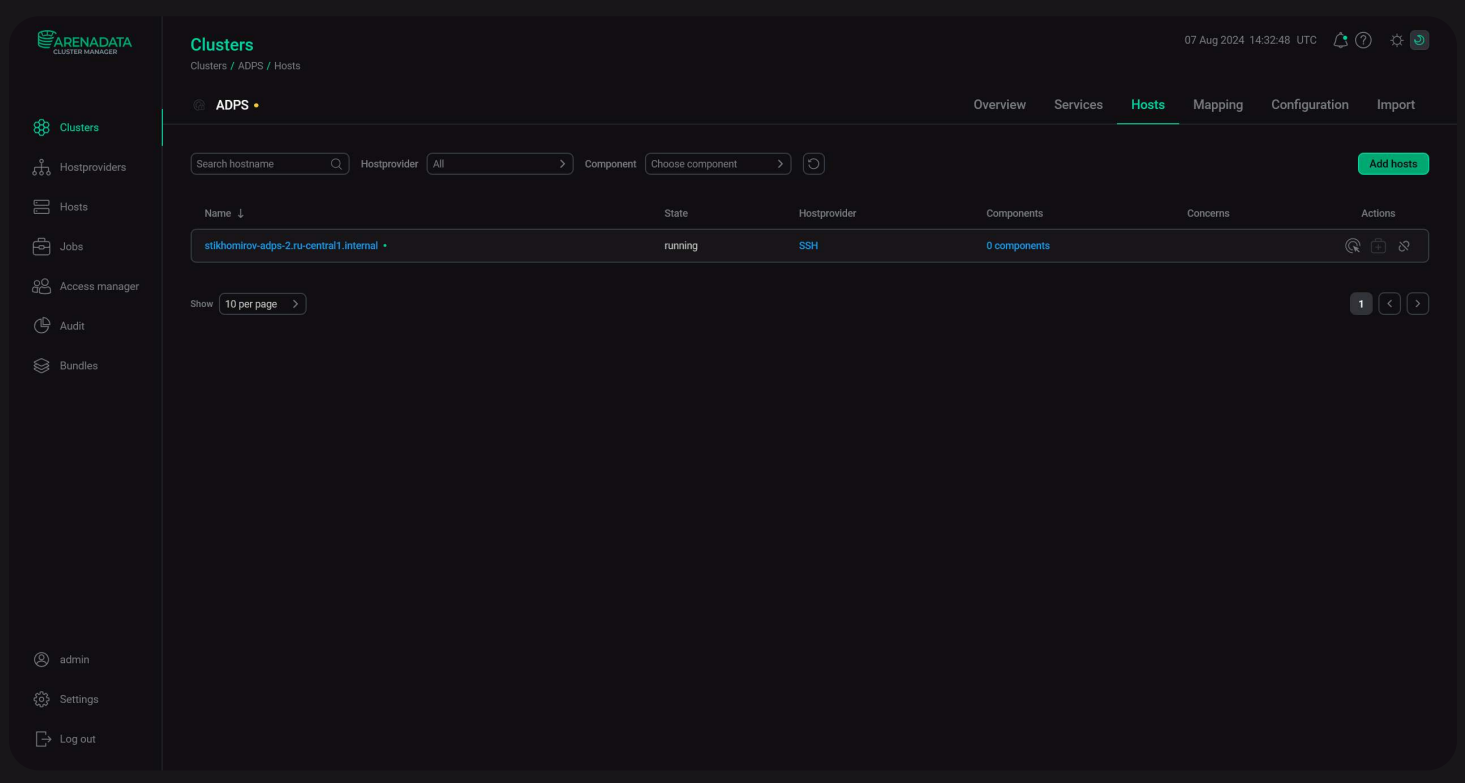
Switch to adding hosts

3. In the opened dialog, select hosts that should be added to the cluster and click ***Add***.



Select hosts

4. As a result, the added hosts are displayed on the ***Hosts*** tab.



The result of successful adding hosts to a cluster

Add components

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

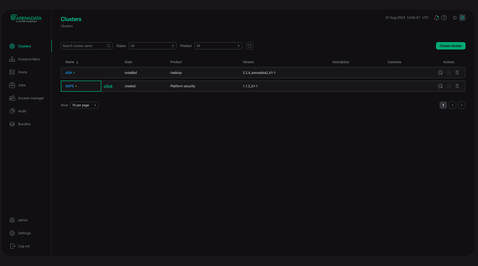
In ADCM a **component** means a part of a service that should be deployed on one or several cluster hosts.

IMPORTANT

Initially, there are no components on any of cluster hosts. You should map components to hosts manually.

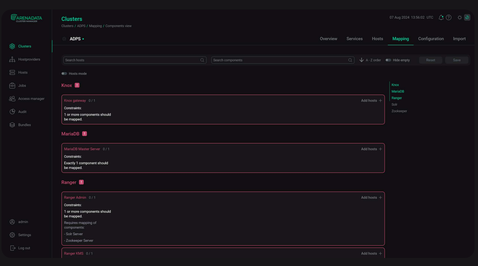
The steps for adding components to hosts are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



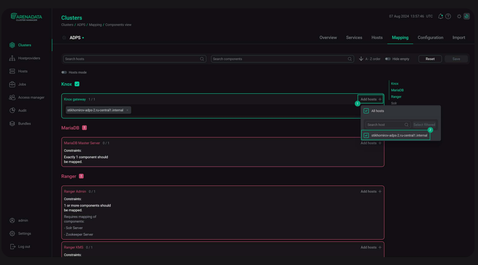
Select a cluster

2. Open the ***Mapping*** tab on the cluster page.



Switch to mapping service components

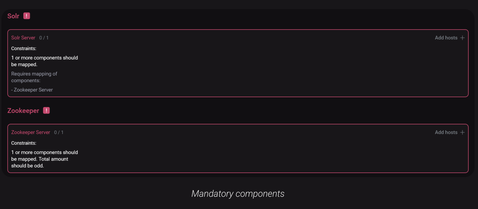
3. In the window that opens, map components to hosts. To assign hosts for each component, click ***Add hosts*** and select hosts in the pop-up window.



Select hosts for a component

Mandatory components

Each service can have mandatory and optional components. Mandatory components are highlighted in red. The ADCM UI also displays how many mandatory components should be installed.



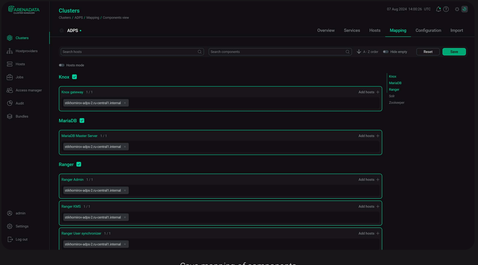
Mandatory components

CAUTION

Without assigning a necessary number of hosts to mandatory components, the component mapping cannot be saved.

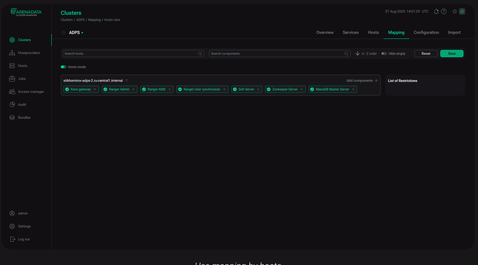
For safety reasons it's recommended to distribute the services among several hosts to ensure fault tolerance, but it is acceptable to configure everything to just one host.

4. After component distribution is completed, click ***Save***.



Save mapping of components

Starting with ADCM 2.2.0, you can use another mapping mode – by hosts. To do this, switch on the ***Hosts mode*** toggle and map components to hosts by clicking ***Add components*** for each host sequentially.



Use mapping by hosts

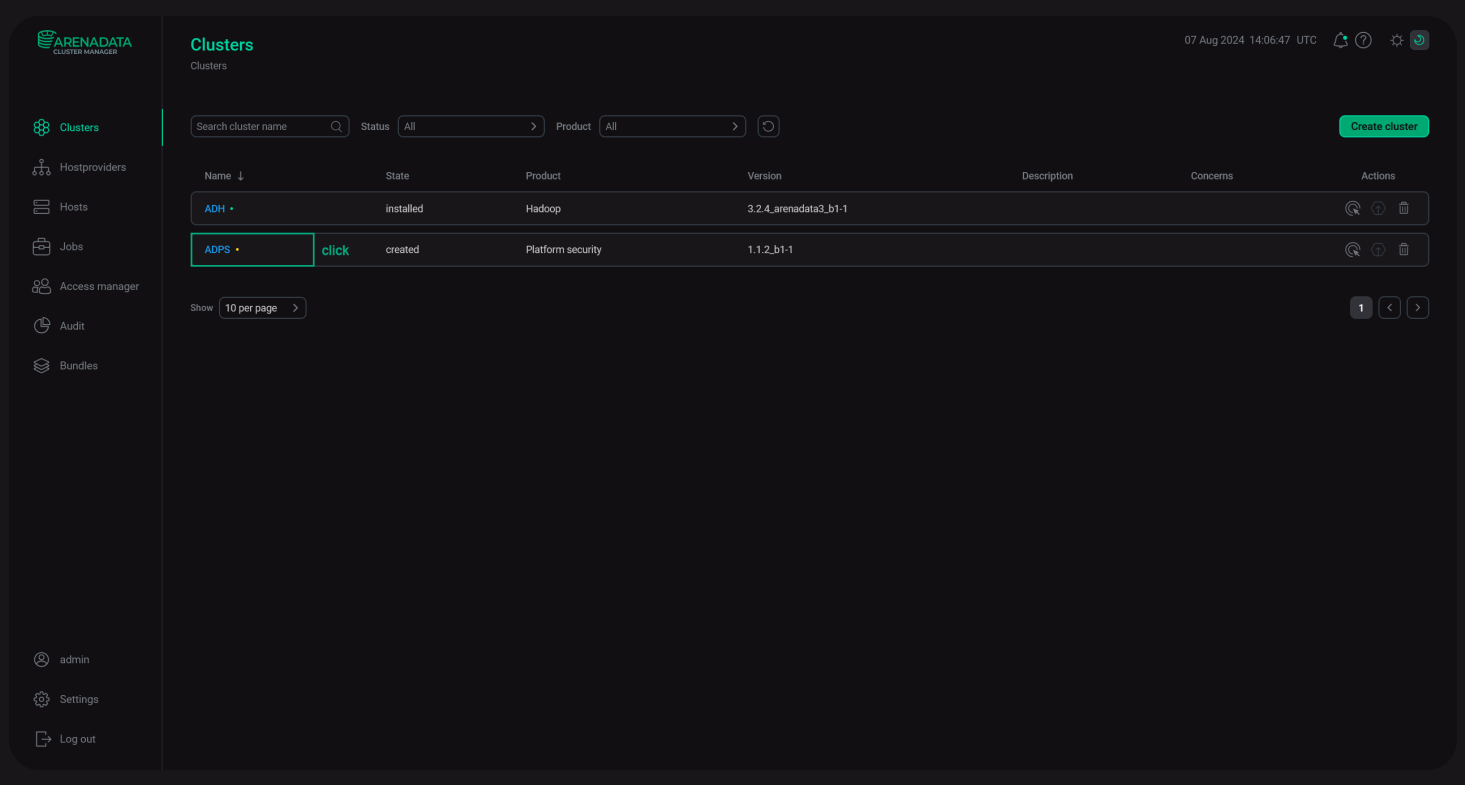
Configure services

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

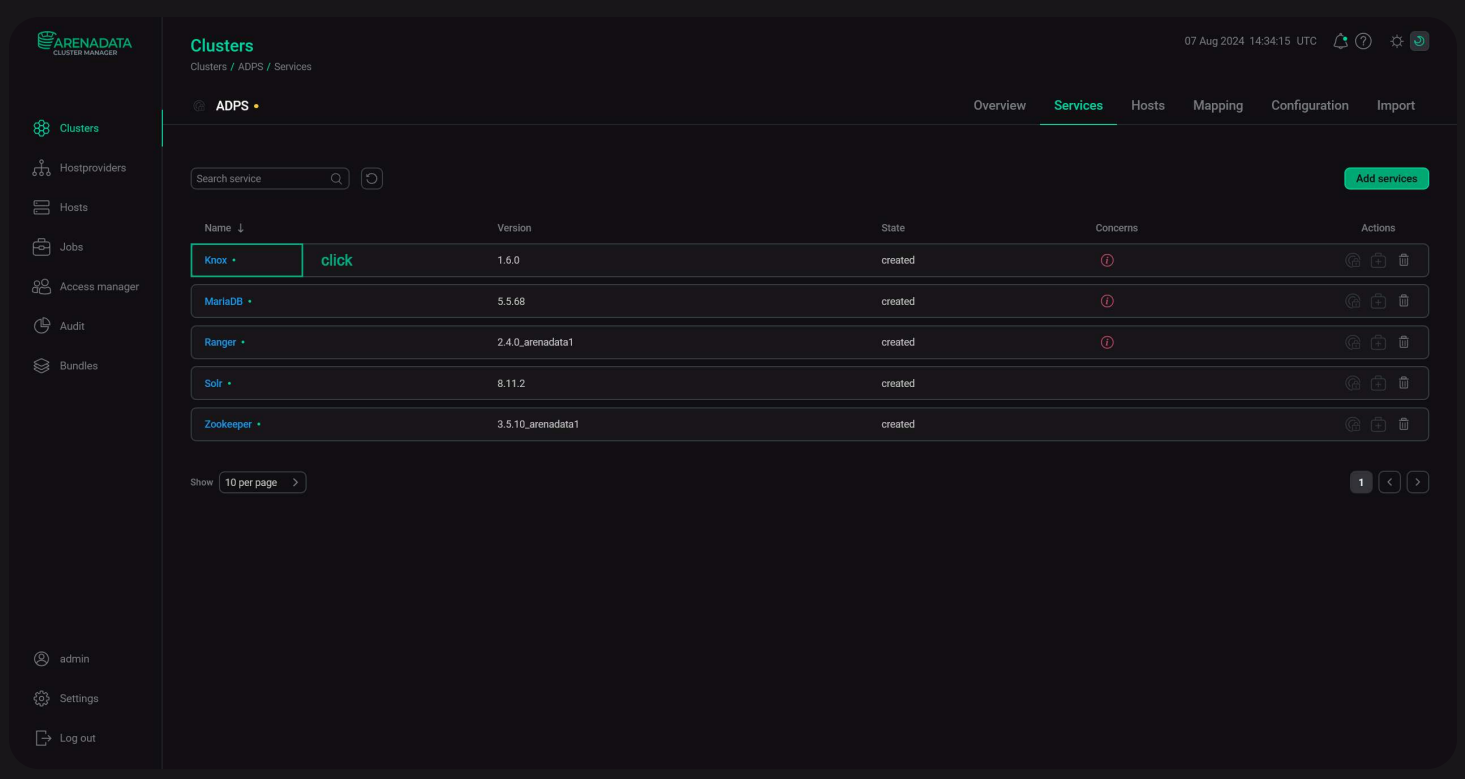
After adding all necessary services, you can configure its parameters. The steps for configuring cluster services are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



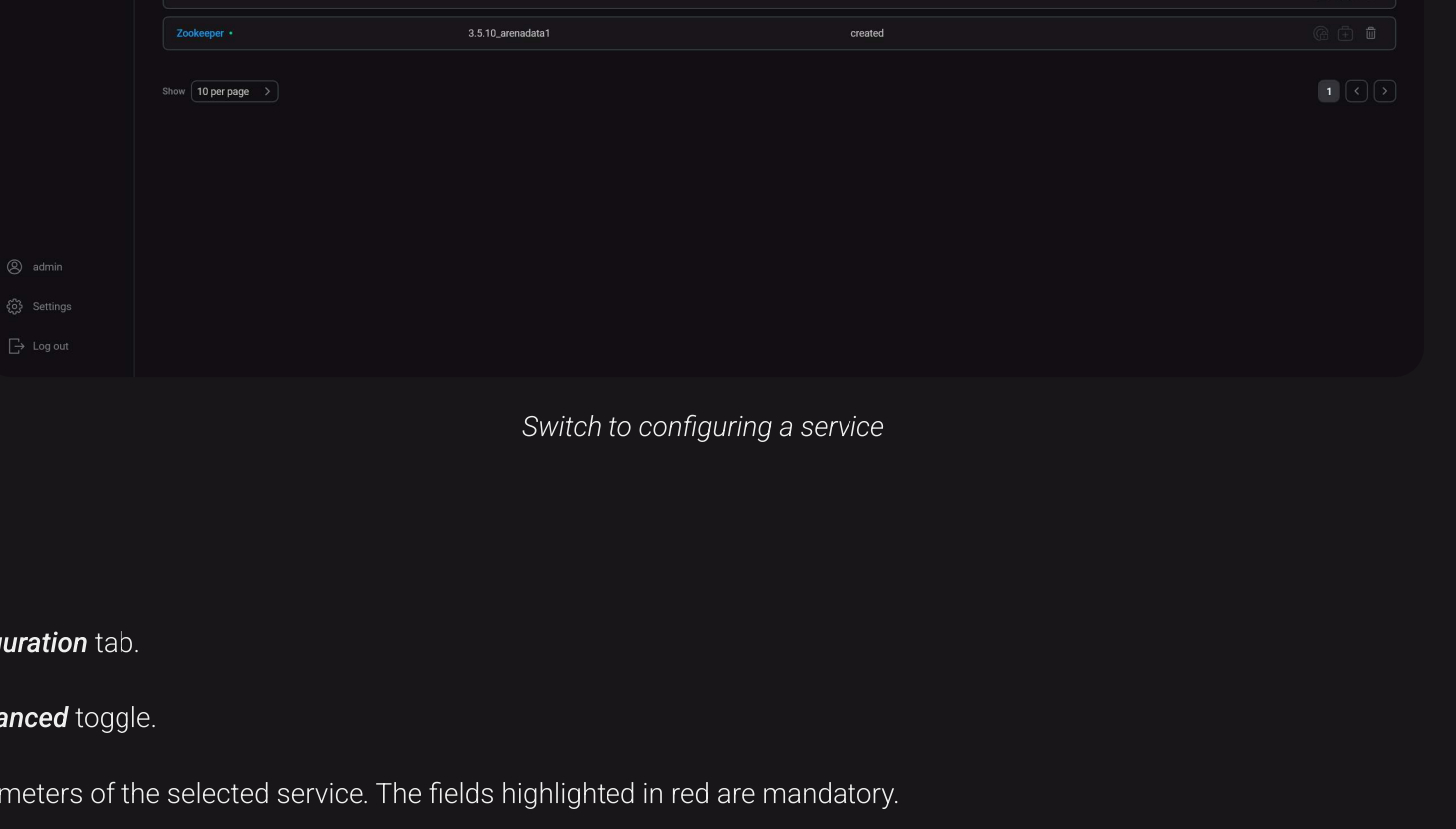
Select a cluster

2. Open the ***Services*** tab on the cluster page, find the row that contains the desired service, and click its name in the ***Name*** column.



Switch to configuring a service

3. In the window that opens:
   - Open the ***Primary Configuration*** tab.
   - Switch on the ***Show advanced*** toggle.
   - Fill in configuration parameters of the selected service. The fields highlighted in red are mandatory.
   - Click ***Save***.



Configure a service

NOTE

For more information about services configuration parameters, see Configuration parameters.

Configure a cluster

Sergei Tikhonov, Daria Barysheva

AD.PS

Contents

To Table of Contents

Primary configuration

Configuration groups

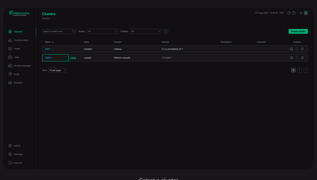
Ansible settings

IMPORTANT

There is usually no need to change cluster configuration parameters. You can leave all parameters at the default values.

After creation a new cluster, you can configure it by performing the following steps:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



Select a cluster

2. Open the ***Configuration*** tab on the cluster page. The ***Configuration*** tab includes the following sections: ***Primary configuration***, ***Configuration groups***, ***Ansible settings***.
3. Fill in all necessary parameters on the selected tab and click ***Save***.

Primary configuration

The ***Configuration → Primary configuration*** tab allows you to configure the general cluster settings.

Available configuration parameters are listed below:

- Paths to the repositories for CentOS, Red Hat, ALT Linux 8, and Astra Linux that are used in the process of installing — depending on the cluster operation system:
  - ***monitoring*** — monitoring components repository;
  - ***zookeeper_repo*** — Arenadata Zookeeper YUM repository;
  - ***arenadata*** — Arenadata YUM repository.
- The ***Kerberos*** section that includes settings listed in the table below.

Parameter	Description	Default value
Enable Kerberos	Enables the Kerberos authentication	False
Authentication on WEB UIs	Enables the Kerberos authentication on WEB UIs	False
SPNEGO Signature Value	A secret key that is used to generate and verify the signature in the HTTP authentication protocol SPNEGO (in the Base64 format)	—
Kerberos KDC type	A KDC type. For more information, see Kerberos	MIT
KDC hosts	One or more KDC hosts with running FreeIPA server(s)	—
Realm	A Kerberos realm to connect to the FreeIPA server	—
Domains	One or more domains associated with FreeIPA	—
Kadmin server	A host where `kadmin` is running	—
Kadmin principal	A principal name used to connect via `kadmin`, for example `admin@RU-CENTRAL1.INTERNAL`	—
Kadmin password	An IPA Admin password	—
Keytabs directory	A directory with keytab files	/etc/security/keytabs
Additional realms	Additional Kerberos realms	—
Delay between kinit invocation attempts	Time to wait before attempting another kinit operation if the previous one has failed	5
Number of retries for kinit invocation attempts	Number of attempts to launch the kinit command if it fails	10
Trusted Active Directory server	An Active Directory server for one-way cross-realm trust from the MIT Kerberos KDC	—
Trusted Active Directory realm	An Active Directory realm for one-way cross-realm trust from the MIT Kerberos KDC	—
Custom krb5.conf	Enables adding custom parameters to the *krb5.conf* file	False
krb5.conf	Additional parameters to write to the *krb5.conf* file	—
Admin DN	A full DistinguishedName of the admin user with rights to `create/modify/delete/pwdchange` user accounts in the target Organizational Unit	—
LDAP URL	LDAP URL that consists of the `ldap://` or `ldaps://` protocol prefix, hostname or IP address and port of an AD server. For example, `ldaps://192.168.4.2:636`	—
Container DN	A container distinguished name	—
TLS CA certificate Path	A CA certificate path on the host filesystem that is pre-located or should be written from the ***TLS CA certificate*** field	—
TLS CA certificate (optional)	A CA certificate that is written on a host by the path from ***TLS CA certificate Path*** during the execution of the ***Enable Kerberos*** action. The certificate is not saved in ADCM	—
Custom ldap.conf	Enables adding custom parameters to the *ldap.conf* file	False
ldap.conf	Additional parameters that are written to the *ldap.conf* file	—
IpaClient No NTP Autoconf	Disables the NTP configuration during the IPA client installation	False
IpaClient No DNS Lookup	Disables the DNS lookup for the FreeIPA server during the IPA client installation	True
Set up Kerberos utils	Defines whether to install or remove the Kerberos client and utils	True
Configure Kerberos on hosts	Cluster configuration including *krb5.conf* and *ldap.conf*	True
Set up principals and keytabs	Defines whether to create, re-create, or remove principals and keytabs	True
Configure services and clients	Defines whether to update the services and clients configuration	True
Run service checks	Defines whether to run the service checks	True

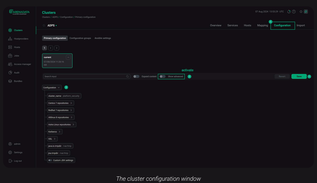
- The ***SSL*** section with the settings listed in the table below.

Parameter	Description	Default value
Enable SSL	Indicates whether SSL is enabled or not	False
Keystore path	A path to the keystore file	/tmp/keystore.jks
Keystore password	A password for the keystore file	—
Truststore path	A path to the truststore file	/tmp/truststore.jks
Truststore password	A password for the truststore file	—
TLS Version	Version of the TLS protocol	TLSv1.2
Run service checks	Defines whether to run the service checks	True

- The ***Advanced SSL*** section with the settings listed in the table below.

Parameter	Description	Default value
Java cacerts file location	Path to the *cacerts* file location of your Java distribution	/etc/pki/ca-trust/extracted/java/cacerts
Java cacerts file password	Password for the *cacerts* file of your Java distribution	—

- The ***java.io.tmpdir*** parameter that specifies an environment variable to control `java.io.tmpdir` for Solr.
- The ***jna.tmpdir*** parameter that specifies an environment variable to control `jna.tmpdir` for Solr.
- The ***Custom JDK settings*** group of parameters specifies the custom Java path.



The cluster configuration window

Configuration groups

The ***Configuration → Configuration groups*** tab is designed to set cluster configuration groups.

Ansible settings

The ***Configuration → Ansible settings*** tab allows you to set Ansible configuration options at the cluster level. The tab is available starting with ADCM 2.2.0.

Ansible settings

Name	Description	Default
forks	The number of parallel processes to spawn when communicating with remote hosts	5

Install a cluster

Sergei Tikhonkov, Daria Barysheva

AD.PS

Contents

To Table of Contents

Step 1. Installation

- Whole cluster
- Single services

Step 2. View installation results

Step 3. Check the cluster state after the installation

Step 1. Installation

There are two ways to install cluster services:

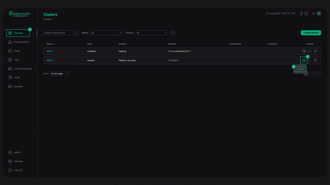
1. **Whole cluster**. In this method all services are installed automatically one by one after applying the ***Install*** action to a cluster.
2. **Single services**. In this method each service is installed manually after applying one or more actions to it. This way is also suitable for installing new services in the already deployed cluster.

In both cases each service starts automatically after installation.

Whole cluster

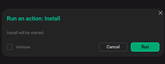
To install all cluster services within a single action, follow the steps:

1. Apply the ***Install*** action to the cluster selected on the **Clusters** page by clicking the icon in the ***Actions*** column.



Switch to the whole cluster installation

2. Verify the requested action in the opened window.



Verify the action

3. Wait until the job is completed.

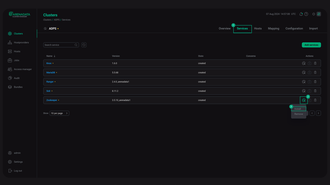
Single services

To install one or several cluster services manually, add these services to the cluster and then perform the actions described below.

CAUTION

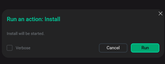
Notice that services are installed one by one. Do not install another service until the installation of the previous one is completed.

1. Open the **Services** tab on the cluster page. For each service that should be installed, click the icon in the **Actions** column and select the ***Install*** action.



Switch to installation of single services

2. Verify the action in the opened window.



Verify the action

3. Wait until the job is completed. Then check that the service state has changed from `created` to `installed`.
4. Repeat the previous actions for other services that should be installed.

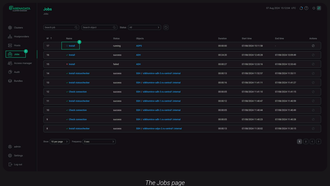
IMPORTANT

When you install services manually, one by one, use the following order:

1. Zookeeper
2. Solr
3. MariaDB
4. Ranger
5. Knox
6. Monitoring

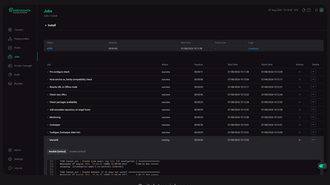
Step 2. View installation results

ADCM starts a single job for installation process, as well as for any other task. You can find out about the status of jobs more specifically on the **Jobs** page.



The Jobs page

The successful completion of the service installation is determined by the transition of the job from the `running` status to the `success` status on the **Jobs** page. If the job fails, it switches to the `failed` status. In this case it is possible to see more detailed information about the errors that occurred by clicking the failed job on the **Jobs** page.



Switch to a job page

A job page contains two sections displayed at the left side of the screen: ***ansible [stdout]*** and ***ansible [stderr]***. These are technical logs of the job that include information from standard I/O streams stdout/stderr. These logs can help to understand what caused the problems.

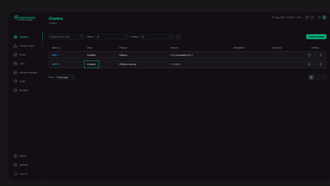
There can also be the optional third section ***ansible [check]***. These are the logs of the most frequent errors checks. The description of these errors is more simple and specific, than in two previous technical logs.

The contents of all three sections should be studied, if errors occur.

Step 3. Check the cluster state after the installation

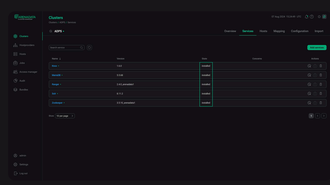
As a result of successful installation, the cluster and its services change their state according to the following rules:

- The cluster state, displayed in the ***State*** field on the ***Clusters*** page, is being changed from `created` to `installed`.



The cluster state after successful installation

- The state of cluster services, displayed on the **Services** tab of the cluster page, is being changed from `created` to `installed`.



The state of cluster services after successful installation

Install monitoring

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

Monitoring is installed optionally (yet strongly recommended). ADCM monitoring is based on three services: Diamond, Graphite, and Grafana.

For **online** installation, you need a separate monitoring cluster. The steps for installing monitoring in this case are listed below. You can perform them both before and after the ADPS cluster installation, except the last step Integrate with ADPS cluster, that can be executed only when the ADPS cluster already exists.

- Create a cluster
- Add services
- Add hosts to a cluster
- Add components
- Configure services
- Install a cluster
- Integrate with ADPS cluster

Create a cluster

Daria Barysheva

Contents

To Table of Contents

Step 1. Download a cluster bundle

Step 2. Upload a cluster bundle to ADCM

Step 3. Create a cluster based on the uploaded bundle

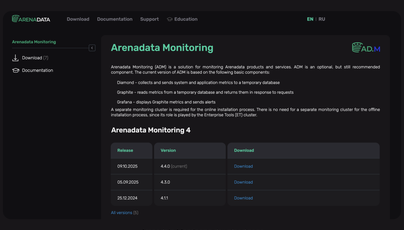
In ADCM a **cluster** means a set of services running on a set of hosts.

Clusters distributions for ADCM are made in the form of so-called bundles. Generally a bundle is a regular archive that includes a description and software logic of cluster deployment and exploitation.

The steps for adding a cluster to ADCM are listed below.

Step 1. Download a cluster bundle

1. Go to https://network.arenadata.io/ and select ***Arenadata Monitoring***.
2. Download a bundle of the desired version.

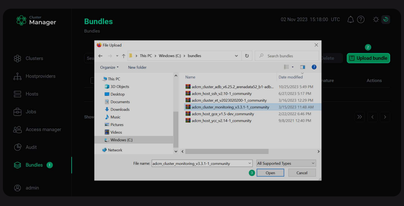


Arenadata Monitoring download page

Step 2. Upload a cluster bundle to ADCM

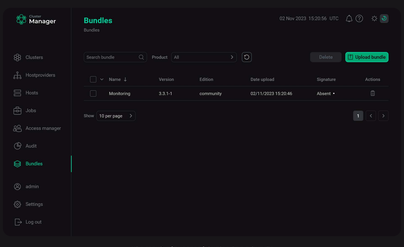
To upload a bundle to ADCM, follow the steps:

1. Select the ***Bundles*** item in the left navigation menu and click ***Upload bundle***.
2. Select a bundle in the Open File Dialog.



Upload a bundle

3. As a result of the performed actions, a bundle is displayed on the ***Bundles*** page.



The result of successful uploading a bundle

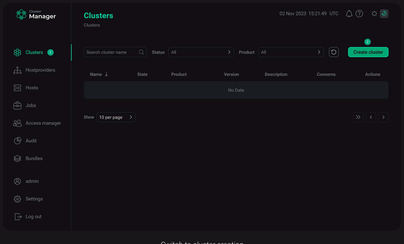
Step 3. Create a cluster based on the uploaded bundle

IMPORTANT

When creating a cluster via the ADCM web interface, you generate a new ADCM cluster instance. It means only adding the cluster to ADCM — at this stage no services are deployed.

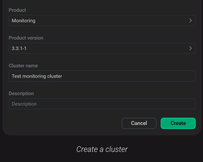
The steps for cluster creation are given below:

1. Select the ***Clusters*** item in the left navigation menu and click ***Create cluster***.



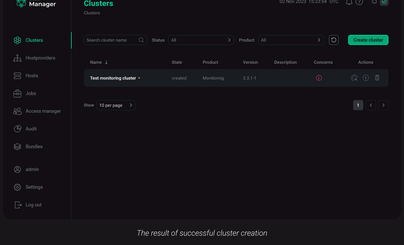
Switch to cluster creation

2. In the opened dialog, describe a new cluster instance:
   - Select an uploaded cluster bundle in the ***Product*** field.
   - Select a bundle version in the ***Product version*** field. Several versions become available in the case of different versions of the same bundle being uploaded.
   - Enter a cluster name in the ***Cluster name*** field.
   - Enter a cluster description in the ***Description*** field if necessary.
   - Set the ***I accept Terms of Agreement*** flag to sign the license agreement. This flag is displayed only for Enterprise bundles. To read the agreement text, refer to the ***Terms of Agreement*** link. You can also sign the license agreement on the ***Bundles*** page by clicking the desired product bundle.
   - Click ***Create***.



Create a cluster

3. As a result, the created cluster instance is displayed on the ***Clusters*** page.



The result of successful cluster creation

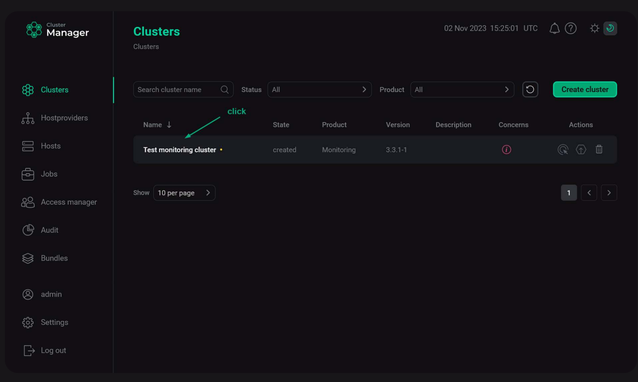
Add services

Daria Barysheva

To Table of Contents

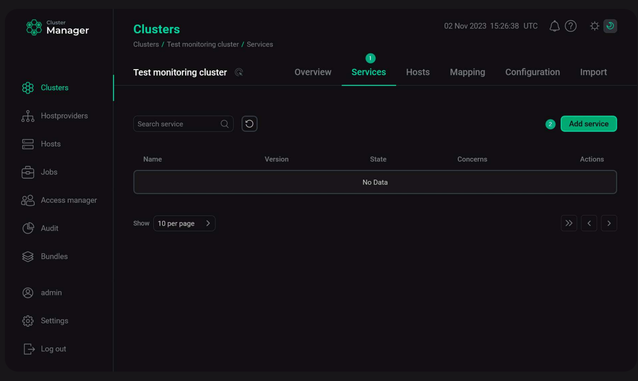
In ADCM a **service** means a software that performs some function. The steps for adding services to a cluster are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



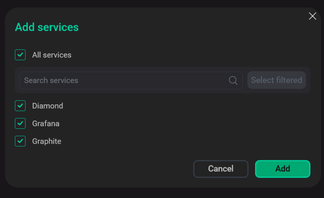
Select a cluster

2. Open the ***Services*** tab on the cluster page and click ***Add service***.



Switch to adding services

3. In the opened dialog, select services that should be added to the cluster and click ***Add***.



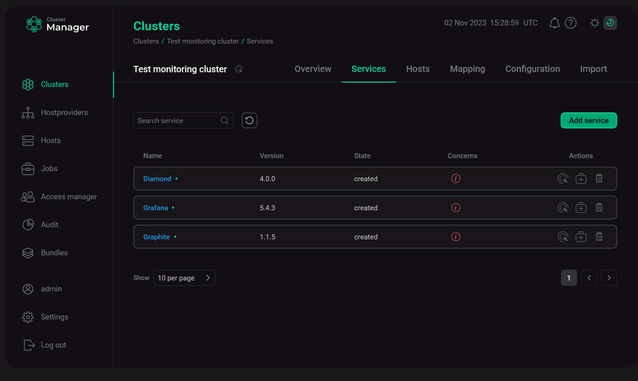
Select services

The brief description of available services is listed below.

Services that should be added to the monitoring cluster

Service	Purpose
Diamond	A Python service used for collecting system metrics from the hosts that are connected to the monitoring system. The collected metrics are processed by Carbon (the backend services of Graphite)
Graphite	Includes several components: ■ **Carbon** — a set of services that are used to process metrics collected by Diamond. ■ **Whisper DB** — a temporary file database where collected metrics are stored. ■ **Web service** — allows to get and visualize metrics from Whisper DB.
Grafana	A web service that allows users to create convenient sets of graphs, alerts, and other visualization tools for metrics received from Graphite. By default, uses the port `3000`

4. As a result, the added services are displayed on the ***Services*** tab.



The result of successful adding services to a cluster

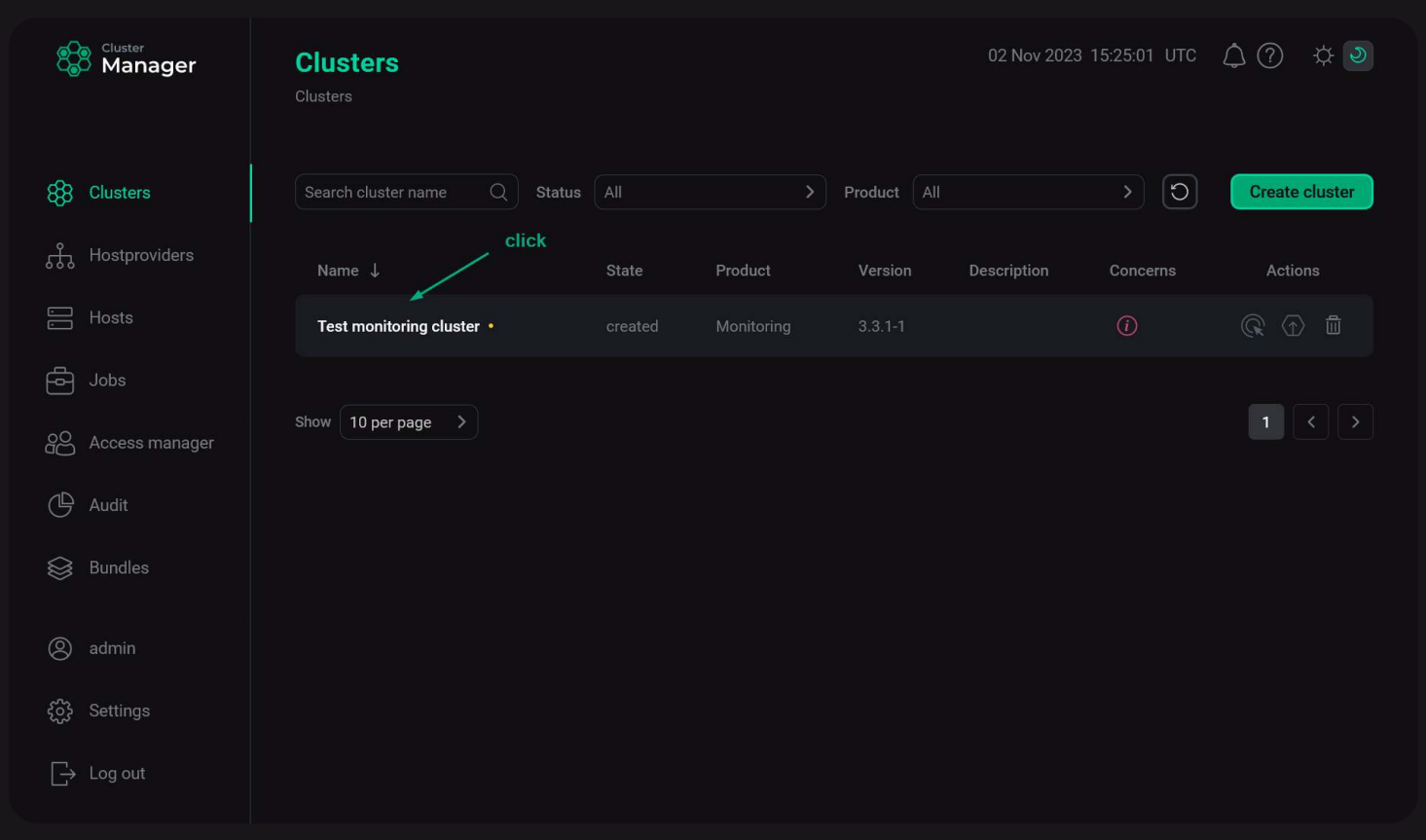
Add hosts to a cluster

Daria Barysheva

To Table of Contents

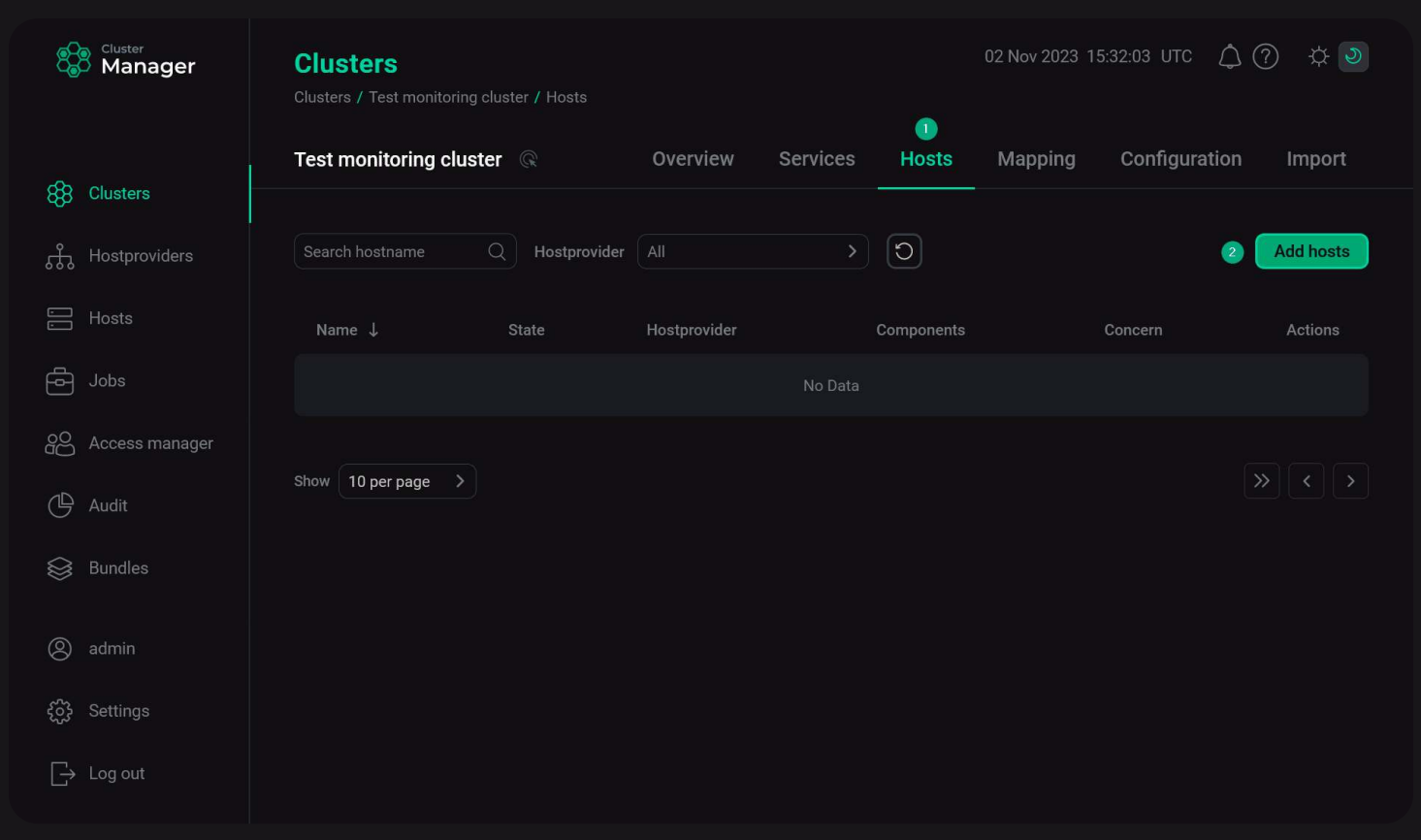
The steps for adding hosts to a cluster are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



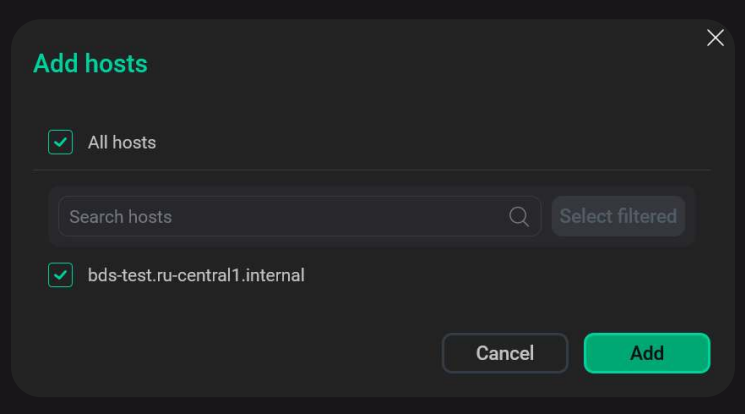
Select a cluster

2. Open the ***Hosts*** tab on the cluster page and click ***Add hosts***.



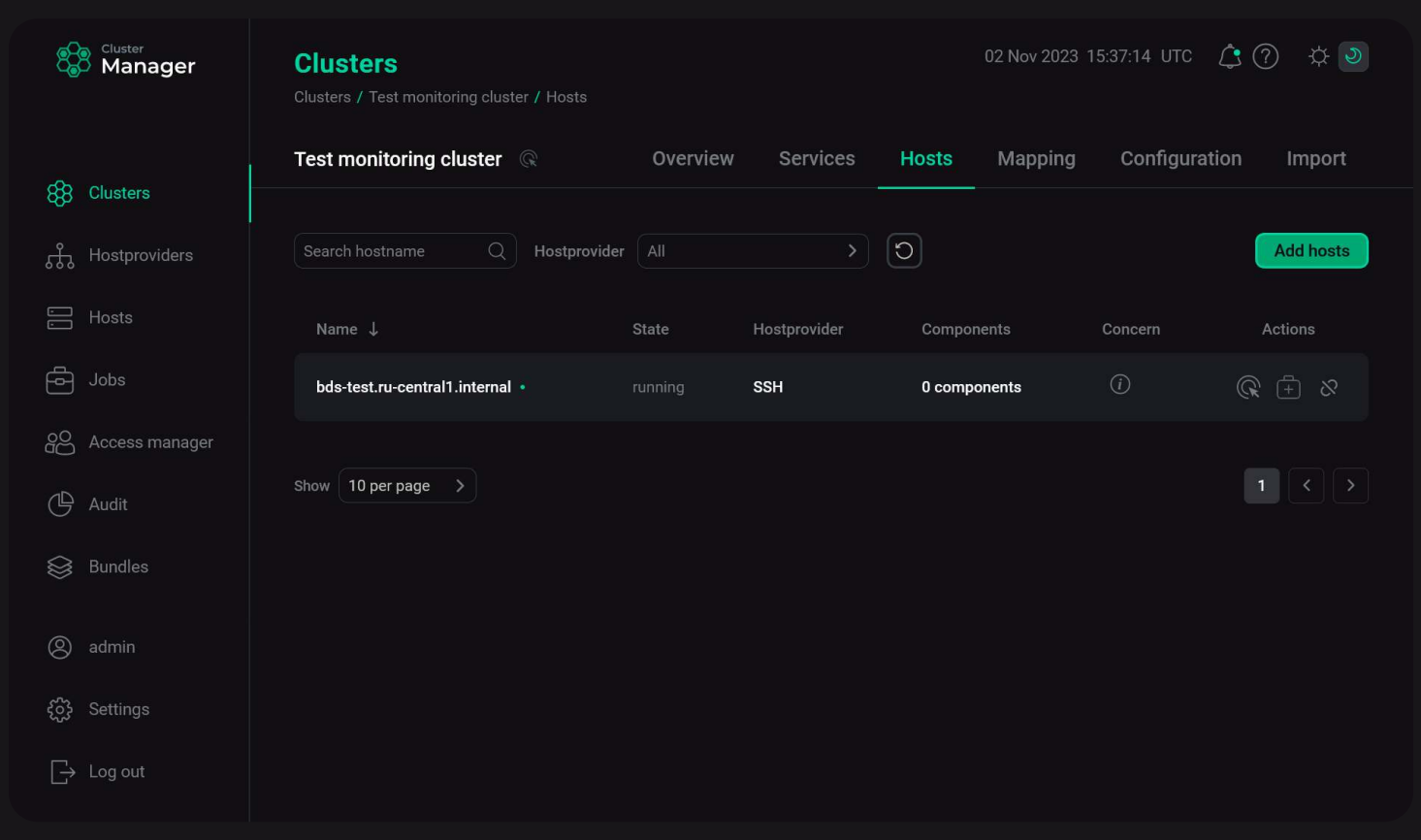
Switch to adding hosts

3. In the opened dialog, select hosts that should be added to the cluster and click ***Add***.



Select hosts

4. As a result, the added hosts are displayed on the ***Hosts*** tab.



The result of successful adding hosts to a cluster

Add components

Daria Barysheva

To Table of Contents

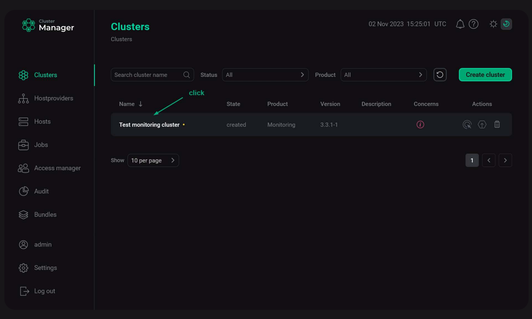
In ADCM a **component** means a part of a service that should be deployed on one or several cluster hosts.

IMPORTANT

Initially, there are no components on any of cluster hosts. You should map components to hosts manually.

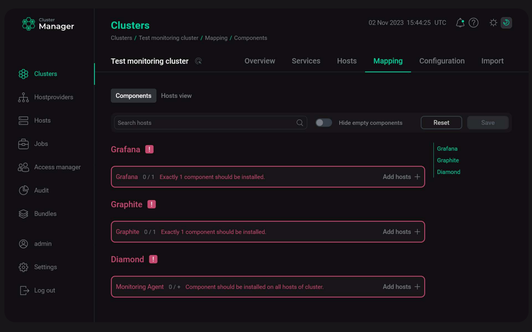
The steps for adding components to hosts are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



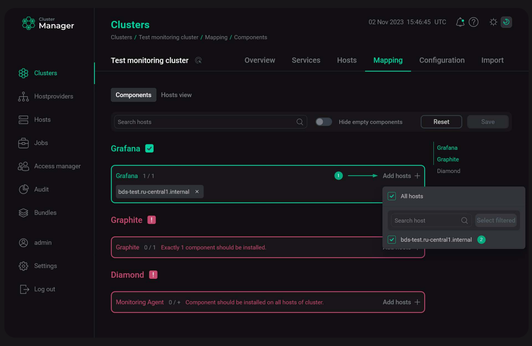
Select a cluster

2. Open the ***Mapping*** tab on the cluster page.



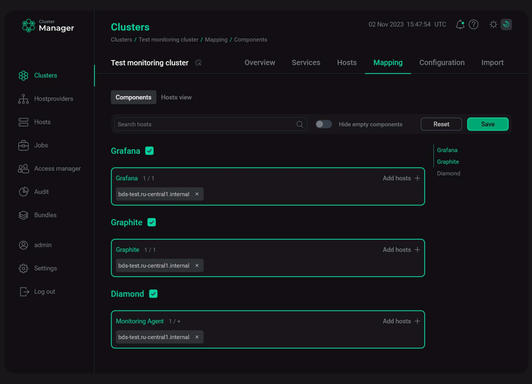
Switch to mapping service components

3. In the window that opens, map components to hosts. To assign hosts for each component, click ***Add hosts*** and select hosts in the pop-up window.



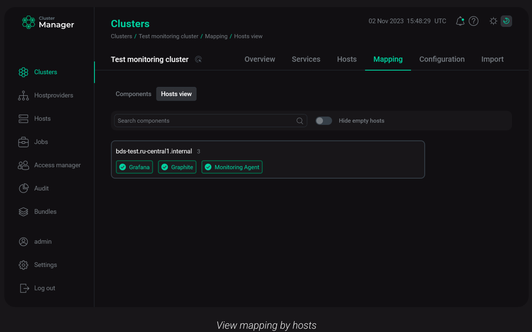
Select hosts for a component

4. After component distribution is completed, click ***Save***.



Save mapping of components

5. To open an alternative mapping view (by hosts), select the ***Hosts view*** tab.



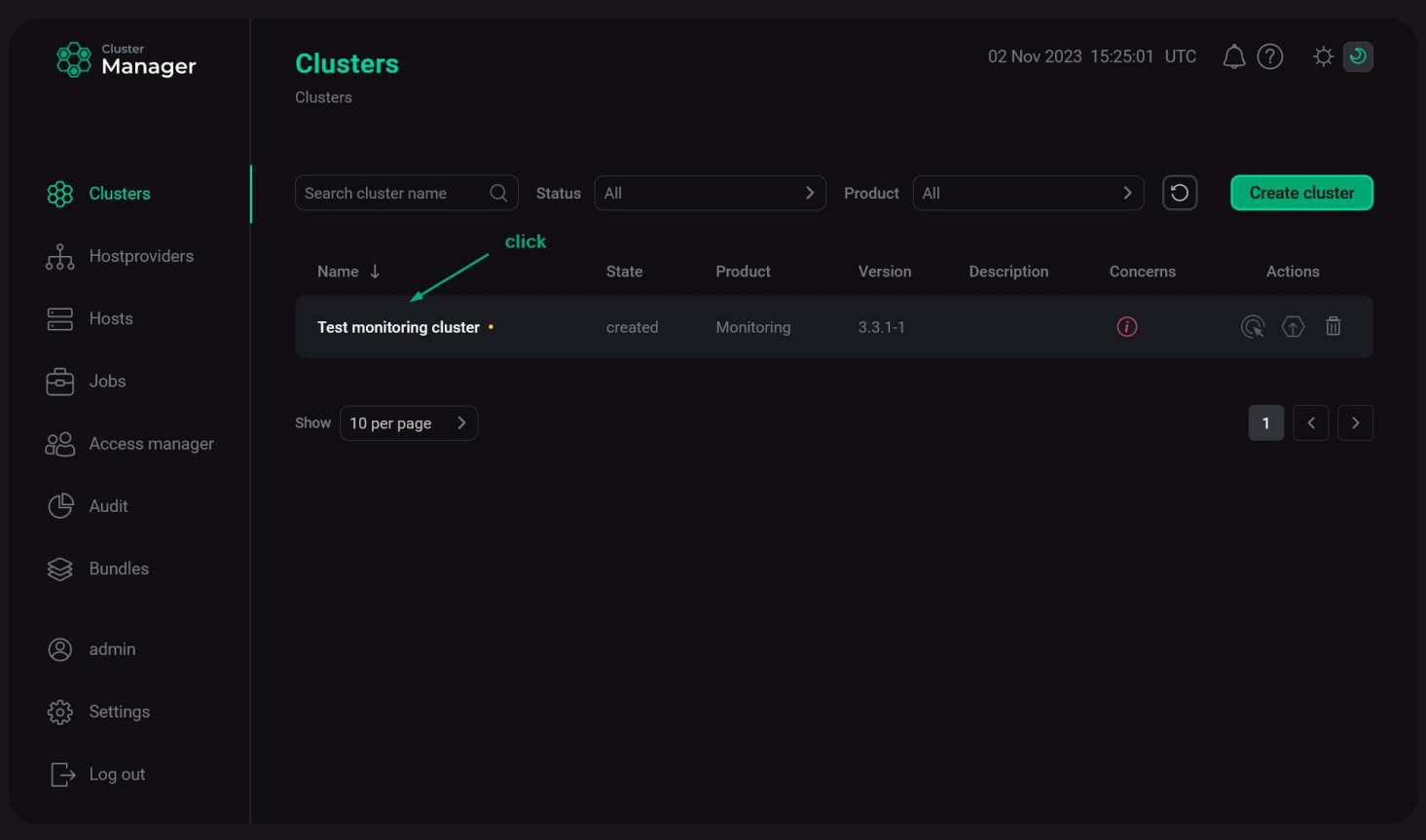
View mapping by hosts

Configure services

Daria Barysheva

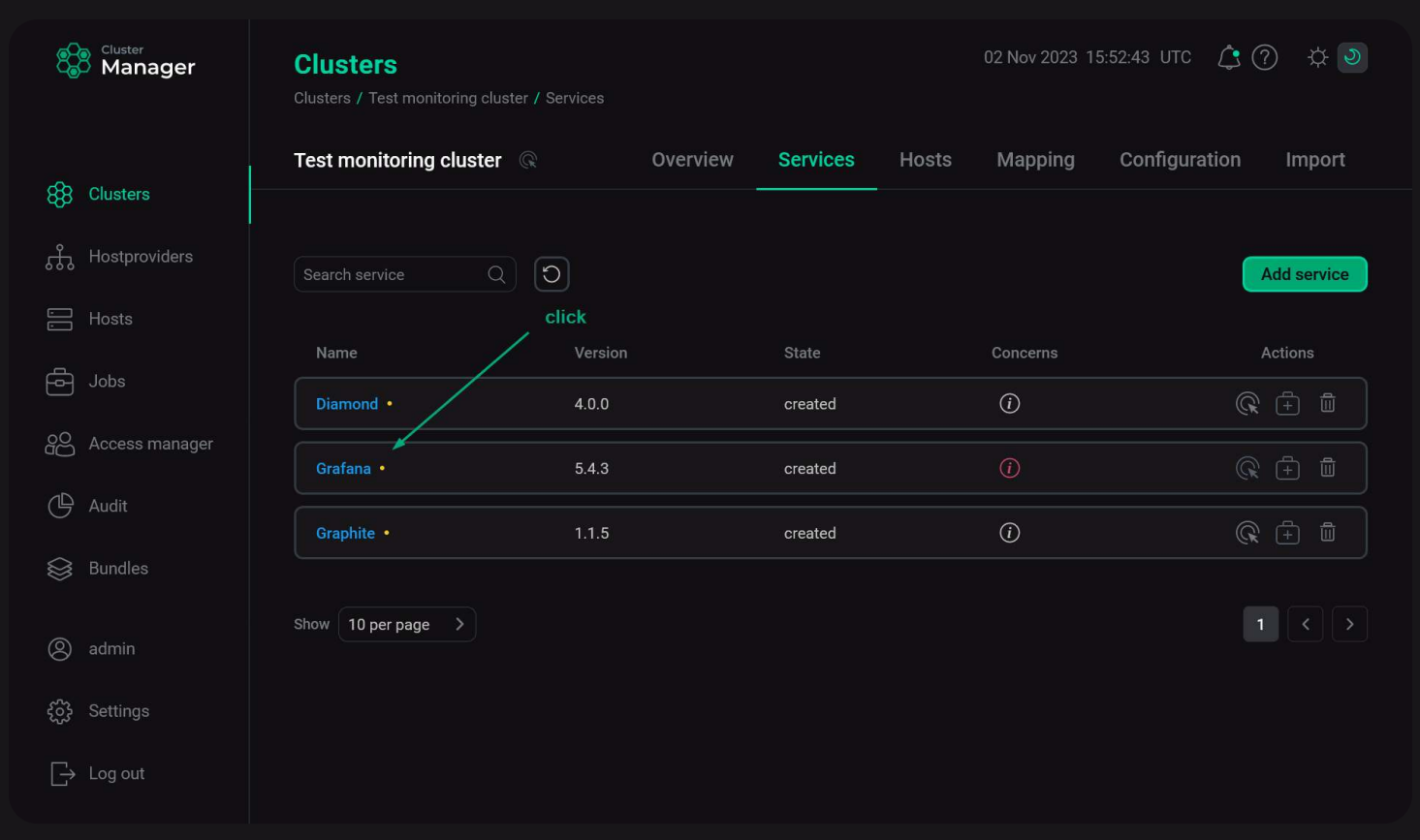
To Table of Contents

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



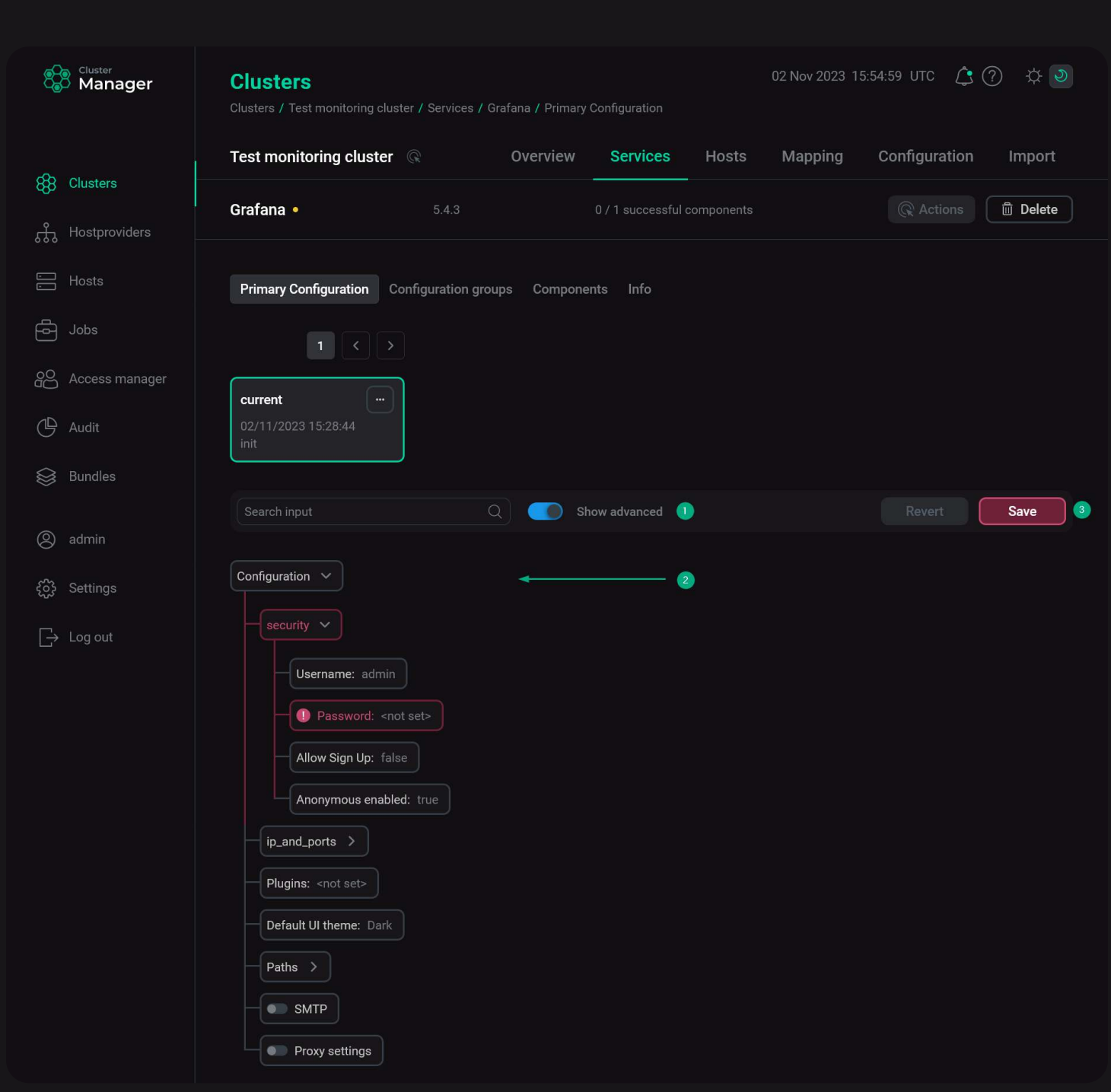
Select a cluster

2. Open the ***Services*** tab on the cluster page, find the row that contains the desired service, and click its name in the ***Name*** column.



Switch to configuring a service

3. In the window that opens:
   - Open the ***Primary Configuration*** tab.
   - Switch on the ***Show advanced*** toggle.
   - Fill in configuration parameters of the selected service. The fields highlighted in red are mandatory.
   - Click ***Save***.



Configure a service

Install a cluster

Daria Barysheva

Contents

To Table of Contents

Step 1. Run installation

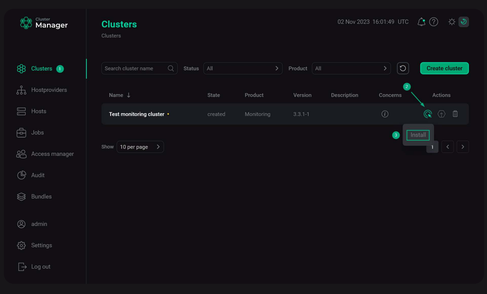
Step 2. View results

- Graphite
- Grafana

Step 1. Run installation

The last step of adding a new monitoring cluster is to install its services. To do this:

1. Apply the ***Install*** action to the cluster selected on the ***Clusters*** page by clicking the icon in the ***Actions*** column. Verify the requested action in the opened window. All services are installed consecutively.



Switch to monitoring cluster installation

2. Wait until the job is completed.

NOTE

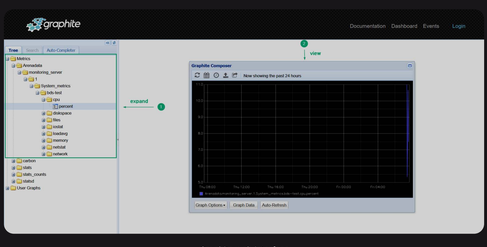
In case of errors, you can find logs on the ***Jobs*** page.

Step 2. View results

Graphite

To check availability of the Graphite web-interface, you can do:

1. In the browser bar, enter an address of the host where the monitoring cluster is deployed. To specify the port number, use the value of the ***Web-interface TCP port*** parameter from the ***ip_and_ports*** section defined when configuring the Graphite service during the monitoring cluster installation. The value `80` (used by default) can be omitted.
2. Expand a tree at the left side of the opened form. System metrics are displayed there, grouped by hosts. You can select some of these metrics to view their change dynamics in the form of a graph at the right side of the screen.

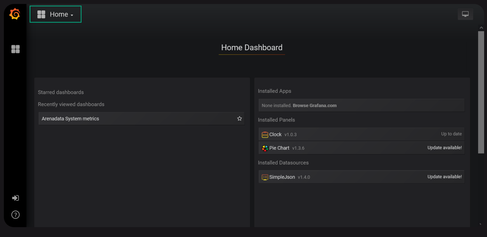


Graphite web-interface

Grafana

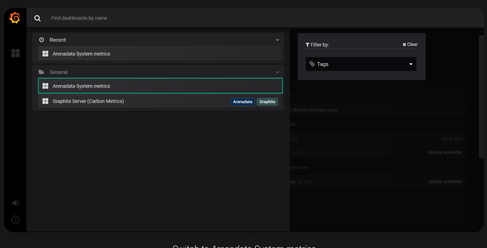
To check availability of the Grafana web-interface, you can do:

1. In the browser bar, enter an address of the host where the monitoring cluster is deployed. To specify the port number, use the value of the ***Port*** parameter from the ***ip_and_ports*** section defined when configuring the Grafana service during the monitoring cluster installation (by default, `3000`).
2. Select the ***Home*** menu item in the opened form.



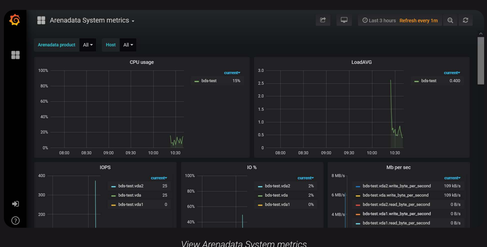
Grafana web-interface

3. Open one of available dashboards. For example, ***Arenadata System metrics***.



Switch to Arenadata System metrics

4. View available graphs.



View Arenadata System metrics

Integrate with ADPS cluster

S. Sergo, T.Nikonov, Dana Zaiyshina

Contents

To Table of Contents

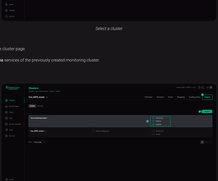
Step 1. Import monitoring settings
Step 2. Add a monitoring service
Step 3. Map monitoring components
Step 4. Install a monitoring service
Step 5. Verify results
- Graphite
- Grafana

In order to determine that a new cluster should be monitored, it is necessary to integrate it with the previously created monitoring cluster and to add a separate Monitoring service into it.

The steps for building integration between a monitoring cluster and ADPS cluster are described below. All actions should be performed on the base of ADPS cluster.

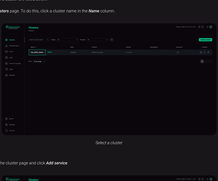
Step 1. Import monitoring settings

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



Select a cluster

2. Open the **Import** tab on the cluster page.
3. Select **Graphite** and **Grafana** services of the previously created monitoring cluster.
4. Click **Import**

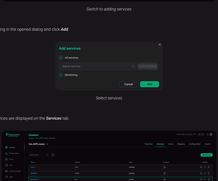


Integration with a monitoring cluster

Step 2. Add a monitoring service

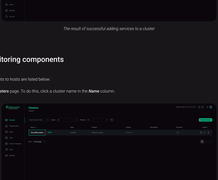
The steps for adding services to a cluster are listed below:

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



Select a cluster

2. Open the **Services** tab on the cluster page and click **Add service**

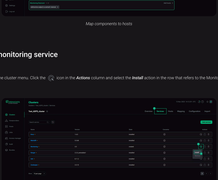


Switch to adding services

3. Select the service Monitoring in the opened dialog and click **Add**

Select services

4. As a result, the added services are displayed on the **Services** tab.

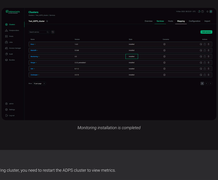


The result of successful adding services to a cluster

Step 3. Map monitoring components

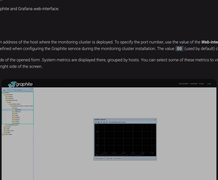
The steps for adding components to hosts are listed below:

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



Select a cluster

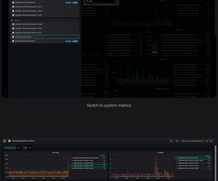
2. Open the **Mapping** tab on the cluster page, map monitoring components to all cluster hosts, and click **Save**



Map components to hosts

Step 4. Install a monitoring service

1. Open the **Services** tab in the cluster menu. Click the ⚙ icon in the **Actions** column and select the **Install** action in the row that refers to the Monitoring service. Verify the action in the opened window.



Switch to monitoring service installation

2. Wait until the job is completed. After successful installation, the monitoring service changes its state from `created` to `installed`

> **NOTE**
>
> In case of errors, you can find logs on the **Jobs** page.

Monitoring installation is completed

> **CAUTION**
>
> After you install the monitoring cluster, you need to restart the ADPS cluster to view metrics.

Step 5. Verify results

To ensure that installation process is completed successfully, check that the system metrics are collected from the hosts of your cluster. Not only from the host where the monitoring cluster has been deployed earlier.

You can check it both in the Graphite and Grafana web-interface.

Graphite

1. In the browser bar, enter an address of the host where the monitoring cluster is deployed. To specify the port number, use the value of the **Web-interface TCP port** parameter from the **ip_and_ports** section defined when configuring the Graphite service during the monitoring cluster installation. The value `80` (used by default) can be omitted.
2. Expand a tree at the left side of the opened form. System metrics are displayed there, grouped by hosts. You can select some of these metrics to view their change dynamics in the form of a graph at the right side of the screen.

New hosts are available in Graphite

Grafana

1. In the browser bar, enter an address of the host where the monitoring cluster is deployed. To specify the port number, use the value of the **Port** parameter from the **ip_and_ports** section defined when configuring the Grafana service during the monitoring cluster installation (by default, `3000`).
2. Select the **Home** menu item in the opened form.

Switch to system metrics

3. View available graphs.

View system metrics

Offline installation

To Table of Contents

This topic describes **offline** installation of ADPS. It can be applied to the hosts without access to the Internet.

For offline installation, you need to install and configure Arenadata Enterprise Tools (ET). ET is a solution that provides the necessary infrastructure for Arenadata product deployment in environments with limited internet access. The solution is a part of the Arenadata data platform. It provides the repositories necessary for installing and updating the platform components.

The process of offline installation via ADCM includes the following steps.

Step 1.

Install ADCM

Step 2.

Prepare hosts

Step 3.

Install ET cluster

Step 4.

Install ADPS cluster

Step 5.

Install monitoring

Install ADCM

Contents

Step 1. Check prerequisites
- Software and hardware requirements
- Install Docker
- Disable SELinux
- Open firewalld
- Set up a proxy for Docker (if it's offline and not behind one)

Step 2. Install ADCM

Step 3. Start ADCM

Step 4. Check installation
- Check the Docker container status
- Allow access to the Docker CLI without root privileges

Step 5. Update ADCM

ADCM is also supported on Astra Linux starting from its current update 1.7.

Step 1. Check prerequisites

Software and hardware requirements

Before you start, make sure your machine meets the minimum system requirements:

- One of the following operating systems:
  - Astra Linux
  - CentOS 7
  - RHEL 7
  - Oracle Linux 7
  - Ubuntu 20.04
  - Ubuntu 22.04
  - Alt Linux SP 10
  - Alt Linux SP 11
  - RED OS 7.3 "Murom"
  - RED OS 8 "Standard"
- The hardware meets the following conditions:

RECOMMENDED

Provides a quick experience, as **minimal**. The target setup should be able to accommodate the customer requirements.

- CPU: 2 CPU cores
- RAM: 4 GB
- Disk: 40 GB

MINIMUM

Starting with ADCM 1.7.1, you have a flexible choice with recommended images of CPU, RAM, and disk parameters compatible with the list of supported operating systems. Minimal configuration is not recommended for production environments. Use it for testing purposes only, for example, to check the ADCM installation process.

- CPU: 1 CPU core
- RAM: 2 GB
- Disk: 20 GB

To perform well, the filesystem needs to provide the following:


```
Sequential read speed: not less than 120 MB/s
Random read speed: not less than 4000 IOPS
Sequential write speed: not less than 40 MB/s
Random write speed: not less than 2000 IOPS
```


Ubuntu 22.04 Minimal Linux AMI ☁ Free

- 1 CPU core
- 1 GB RAM
- 8 GB disk space
- Ubuntu 22.04 Minimal Linux AMI

Ubuntu 22.04 Minimal Linux AMI ☁ Free

- 1 CPU core
- 2 GB RAM
- 8 GB disk space
- Ubuntu 22.04 Minimal Linux AMI

AltLinux SP 10 ☁ Free

- 1 CPU core
- 2 GB RAM
- 8 GB disk space
- AltLinux SP 10

AltLinux SP 11 ☁ Free

- 1 CPU core
- 2 GB RAM
- 8 GB disk space
- AltLinux SP 11

RED OS 7.3 ☁ Free

- 1 CPU core
- 2 GB RAM
- 8 GB disk space
- RED OS 7.3

RED OS 8 ☁ Free

- 1 CPU core
- 2 GB RAM
- 8 GB disk space
- RED OS 8

Install Docker

ALTLINUX

Install Docker from the official repository:

```
apt-get update
apt-get install docker-engine docker-compose
```

Refer to the Alt Linux documentation: Docker installation on Alt Linux and Docker Compose installation on Alt Linux.

CentOS 7/RHEL 7

1. Install **Docker**, which provides Docker from the currently installed Linux system:

```
yum update
yum install docker
```

2. Start Docker:

```
systemctl start docker
```

3. Add **DOCKERIZED** to the storage driver options in the **/etc/sysconfig/docker** file:

```
sed -i '/^DOCKERIZED/d' /etc/sysconfig/docker
echo "DOCKERIZED=yes -s devicemapper" >> /etc/sysconfig/docker
```

4. Restart Docker to confirm the new settings:

```
systemctl restart docker
```

5. Enable Docker to autostart on boot:

```
systemctl enable docker
```

Oracle Linux

1. Install **Docker**, which provides Docker from the currently installed Linux system:

```
yum update
yum install docker
```

2. Start Docker:

```
systemctl start docker
```

3. Add **DOCKERIZED** to the storage driver options in the **/etc/sysconfig/docker** file:

```
sed -i '/^DOCKERIZED/d' /etc/sysconfig/docker
echo "DOCKERIZED=yes -s devicemapper" >> /etc/sysconfig/docker
```

4. Restart Docker to confirm the new settings:

```
systemctl restart docker
```

5. Enable Docker to autostart on boot:

```
systemctl enable docker
```

Ubuntu 20.04

1. Install **Docker**, which provides Docker from the currently installed Linux system:

```
apt-get update
apt-get install docker.io
```

2. Start Docker:

```
systemctl start docker
```

3. Add **DOCKERIZED** to the storage driver options in the **/etc/default/docker** file:

```
sed -i '/^DOCKERIZED/d' /etc/default/docker
echo "DOCKERIZED=yes -s devicemapper" >> /etc/default/docker
```

4. Restart Docker to confirm the new settings:

```
systemctl restart docker
```

5. Enable Docker to autostart on boot:

```
systemctl enable docker
```

Ubuntu 22.04

1. Install **Docker**, which provides Docker from the currently installed Linux system:

```
apt-get update
apt-get install docker.io
```

2. Start Docker:

```
systemctl start docker
```

3. Add **DOCKERIZED** to the storage driver options in the **/etc/default/docker** file:

```
sed -i '/^DOCKERIZED/d' /etc/default/docker
echo "DOCKERIZED=yes -s devicemapper" >> /etc/default/docker
```

4. Restart Docker to confirm the new settings:

```
systemctl restart docker
```

Alt Linux SP 10

1. Install **Docker**, which provides Docker from the currently installed Linux system:

```
apt-get update
apt-get install docker-engine
```

2. Start Docker:

```
systemctl start docker
```

3. Add **DOCKERIZED** to the storage driver options in the **/etc/sysconfig/docker** file:

```
sed -i '/^DOCKERIZED/d' /etc/sysconfig/docker
echo "DOCKERIZED=yes -s devicemapper" >> /etc/sysconfig/docker
```

4. Restart Docker to confirm the new settings:

```
systemctl restart docker
```

Alt Linux SP 11

1. Install **Docker**, which provides Docker from the currently installed Linux system:

```
apt-get update
apt-get install docker-engine
```

2. Start Docker:

```
systemctl start docker
```

3. Add **DOCKERIZED** to the storage driver options in the **/etc/sysconfig/docker** file:

```
sed -i '/^DOCKERIZED/d' /etc/sysconfig/docker
echo "DOCKERIZED=yes -s devicemapper" >> /etc/sysconfig/docker
```

4. Restart Docker to confirm the new settings:

```
systemctl restart docker
```

Astra Linux SE 1.7.1

1. Install **Docker**, which provides Docker from the currently installed Linux system:

```
apt-get update
apt-get install docker.io
```

2. Start Docker:

```
systemctl start docker
```

3. Add **DOCKERIZED** to the storage driver options in the **/etc/default/docker** file:

```
sed -i '/^DOCKERIZED/d' /etc/default/docker
echo "DOCKERIZED=yes -s devicemapper" >> /etc/default/docker
```

4. Restart Docker to confirm the new settings:

```
systemctl restart docker
```

Astra Linux SE 1.7.2

1. Install **Docker**, which provides Docker from the currently installed Linux system:

```
apt-get update
apt-get install docker.io
```

2. Start Docker:

```
systemctl start docker
```

3. Add **DOCKERIZED** to the storage driver options in the **/etc/default/docker** file:

```
sed -i '/^DOCKERIZED/d' /etc/default/docker
echo "DOCKERIZED=yes -s devicemapper" >> /etc/default/docker
```

4. Restart Docker to confirm the new settings:

```
systemctl restart docker
```

Disable SELinux

CentOS

SELinux is enabled in Enforcing mode by default in CentOS 7 to 8. To switch SELinux to permissive mode, run the following command:

```
setenforce 0
```

To make this change permanent, edit the **/etc/selinux/config** file in the following way:

```
sed -i 's/^SELINUX=.*/SELINUX=permissive/g' /etc/selinux/config
```

RHEL

SELinux is enabled in Enforcing mode by default in RHEL 7 to 8. To switch SELinux to permissive mode, run the following command:

```
setenforce 0
```

To make this change permanent, edit the **/etc/selinux/config** file in the following way:

```
sed -i 's/^SELINUX=.*/SELINUX=permissive/g' /etc/selinux/config
```

Oracle Linux

SELinux is enabled in Enforcing mode by default in Oracle Linux 7 to 8. To switch SELinux to permissive mode, run the following command:

```
setenforce 0
```

To make this change permanent, edit the **/etc/selinux/config** file in the following way:

```
sed -i 's/^SELINUX=.*/SELINUX=permissive/g' /etc/selinux/config
```

Astra Linux SE

SELinux is not supported in Astra Linux SE. Skip this step.

Ubuntu

SELinux is not supported in Ubuntu 20.04 to 22.04. Skip this step.

ALTLINUX

SELinux is not supported in ALTLINUX. Skip this step.

Open firewalld

To open firewalld to enable ADCM to run on the default port 8000, run the following commands:

```
firewall-cmd --zone=public --add-port=8000/tcp --permanent
firewall-cmd --reload
```

Set up a proxy for Docker (if it's offline and not behind one)

If you're offline and not behind one, configure a proxy for Docker:

1. Create a directory for the Docker service configuration:

```
mkdir /etc/systemd/system/docker.service.d
```

2. Create the configuration file **/etc/systemd/system/docker.service.d/http-proxy.conf** with the following contents:

```
[Service]
Environment="HTTP_PROXY=http://proxy.example.com:80/"
Environment="HTTPS_PROXY=http://proxy.example.com:80/"
Environment="NO_PROXY=localhost,127.0.0.1,.example.com"
```

3. Reload the daemon for the new configuration and restart Docker:

```
systemctl daemon-reload
systemctl restart docker
```

Step 2. Install ADCM

Before you start, make sure you meet the following steps:

- Check that you have root privileges or a user account that is added to sudo and has the sudo privileges set.
- Check that you have the Docker CLI installed and configured to run without root privileges.
- Check the limits for the maximum number of processes of the available machine.

ALTLINUX

Starting from version 1.6, ADCM is also available in ALTLINUX.

1. Install Docker Image on the system using the command:

```
apt-get install adcm
```

2. Start ADCM:

```
systemctl start adcm
```

OTHER LINUX

1. Install Docker Image on the system using the command:

```
docker pull hub.arenadata.io/adcm/adcm:1.7.2
```

2. Create a new container based on the downloaded image:

```
docker run -d --name adcm -p 8000:8000 hub.arenadata.io/adcm/adcm:1.7.2
```

The command creates a container named **adcm** and starts it.

3. Start ADCM:

```
docker start adcm
```

Step 3. Start ADCM

ADCM runs in a container started by Docker. If you are new to it and have installed Docker, you can start ADCM with the following commands:

1. Load the image:

```
docker pull hub.arenadata.io/adcm/adcm:1.7.2
```

2. Create a container:

```
docker create --name adcm -p 8000:8000 hub.arenadata.io/adcm/adcm:1.7.2
```

3. Start the container:

```
docker start adcm
```

4. Check if ADCM is running:

```
docker ps
```

If the command shows a container with the name **adcm** and the status **Up**, ADCM is running.

Step 4. Check installation

Check the Docker container status

To make sure that the Docker container **adcm** is started and ready for the following commands:

1. Check the container status:

```
docker ps -a
```

The example of the command and the result is provided. Make sure that the **STATUS** column reports **Up**.


```
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS          PORTS                    NAMES
3ab1e0f1f1f0   hub.arenadata.io/adcm/adcm:1.7.2   "/usr/local/bin/sup'…"   About a minute ago   Up About a minute   0.0.0.0:8000->8000/tcp   adcm
```


Check the 8000 port availability

After ADCM is started and the container is up, check that the port **8000** is open and that the daemon is listening:

1. Check that the port **8000** is open:

```
sudo ss -tlpn | grep 8000
```

The example of the command and the result is provided.


```
LISTEN 0      4096         0.0.0.0:8000       0.0.0.0:*    users:(("docker-proxy",pid=3000,fd=4))
```


2. Check that the daemon is listening:

```
sudo netstat -tlpn | grep 8000
```

The example of the command and the result is provided.


```
tcp        0      0 0.0.0.0:8000            0.0.0.0:*               LISTEN      3000/docker-proxy
```


Check the ADCM web interface

To check that ADCM is up and running, open your browser and go to the following default:

- **IP**: your IP
- **Password**: admin

Log in to ADCM

Log in to your Arenadata Cluster Manager. To do this:

1. Enter the username in the **Login** field.
2. Enter the **Password**.
3. Click the **Log in** button.

Log in to ADCM

4. Confirm your password in the **Confirm password** field.
5. Click **Save**.

Change the password

6. In the window that opens, click the **ADCM** tab, then click the **Import** button.

The Import tab

Change the ADCM URL

7. In the window that opens, click **Save**.

Save the ADCM URL

Step 5. Update ADCM

NOTICE

- Before upgrading, it is strongly recommended to back up ADCM data. For more information, see Backup and restore.
- Downgrade of ADCM is not supported.
- Starting from version 1.6, ADCM supports upgrade from version 1.5 and 1.6.
- Starting from version 1.7, ADCM supports upgrade from version 1.6 and 1.7.

To update ADCM, you need to install ADCM of the version you want to upgrade to.

1. Stop the **adcm** container:

```
docker stop adcm
```

2. Remove the **adcm** container:

```
docker rm adcm
```

NOTE

If you want to preserve the old container, rename it instead of removing:

```
docker rename adcm adcm_backup
```

3. Load the image of the new version using one of the following commands:

- **ALTLINUX**: `apt-get install adcm`
- **OTHER LINUX**: `docker pull hub.arenadata.io/adcm/adcm:1.7.2`

4. Create a new container based on the downloaded image:

- **ALTLINUX**:

  DEFAULT: `apt-get install adcm`

  CUSTOM: `apt-get install adcm-<version>`, where **<version>** is the version you want to install, for example, `adcm-1.7.2`.

- **OTHER LINUX**: `docker create --name adcm -p 8000:8000 hub.arenadata.io/adcm/adcm:1.7.2`

5. Start the container:

```
docker start adcm
```

6. Check that ADCM is running:

```
docker ps
```

If the command shows a container with the name **adcm** and the status **Up**, ADCM is running.

7. Update the hosts' configuration to the current version (see Update Hostprovider).

8. Confirm that you have the new version:

```
docker exec adcm cat /adcm/data/version
```

The command returns the current version of ADCM.


```
{"adcm_version": "1.7.2", "adcm_edition": "community", "db_version": "0.0.0", "db_edition": "community"}
```


9. Check that the **adcm** container is running:

```
docker ps -a
```

If the command shows a container with the name **adcm** and the status **Up**, ADCM is running.

10. Delete the **adcm_backup** container:

```
docker rm adcm_backup
```

Prepare hosts

To Table of Contents

To manage a cluster and its nodes via ADCM, it is necessary to prepare hosts for a cluster and link these hosts to ADCM. This is the function of ADCM **hostproviders** — special plugins for ADCM, which encapsulate the complex mechanism of interaction between ADCM and the hosts controlled by it.

ADCM supports several types of hostproviders. For more information on how to configure hostproviders and use them to create hosts, refer to the following documentation:

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

For offline installation, the SSH hostprovider is recommended.

CAUTION

When creating ADPS hosts via ADCM, you should specify their FQDNs. The maximum FQDN length is 49 symbols.

Create a cluster

Daria Barysheva

Contents

To Table of Contents

Step 1. Get the Enterprise version of a cluster bundle

Step 2. Upload a cluster bundle to ADCM

Step 3. Create a cluster based on the uploaded bundle

In ADCM a **cluster** means a set of services running on a set of hosts.

Clusters distributions for ADCM are made in the form of so-called bundles. Generally a bundle is a regular archive that includes a description and software logic of cluster deployment and exploitation.

The steps for adding a cluster to ADCM are listed below.

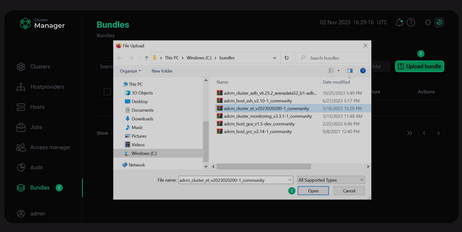
Step 1. Get the Enterprise version of a cluster bundle

Please get a cluster bundle from the Arenadata support team.

Step 2. Upload a cluster bundle to ADCM

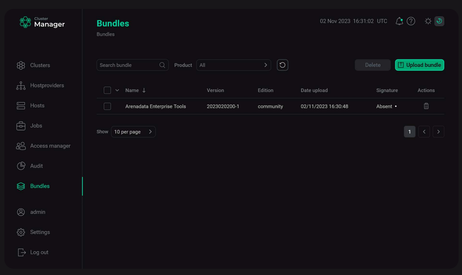
To upload a bundle to ADCM, follow the steps:

1. Select the ***Bundles*** item in the left navigation menu and click ***Upload bundle***.
2. Select a bundle in the Open File Dialog.



Upload a bundle

3. As a result of the performed actions, a bundle is displayed on the ***Bundles*** page.



The result of successful uploading a bundle

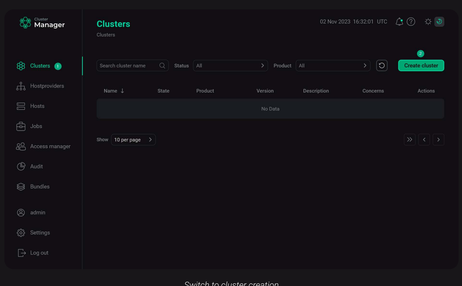
Step 3. Create a cluster based on the uploaded bundle

IMPORTANT

When creating a cluster via the ADCM web interface, you generate a new ADCM cluster instance. It means only adding the cluster to ADCM — at this stage no services are deployed.

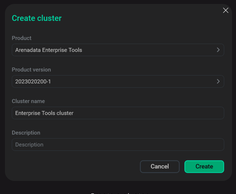
The steps for cluster creation are given below:

1. Select the ***Clusters*** item in the left navigation menu and click ***Create cluster***.



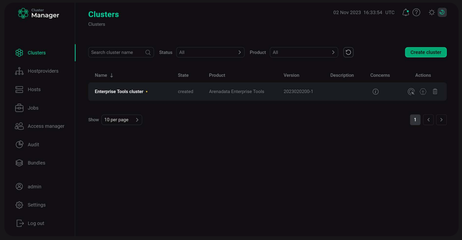
Switch to cluster creation

2. In the opened dialog, describe a new cluster instance:
   - Select an uploaded cluster bundle in the ***Product*** field.
   - Select a bundle version in the ***Product version*** field. Several versions become available in the case of different versions of the same bundle being uploaded.
   - Enter a cluster name in the ***Cluster name*** field.
   - Enter a cluster description in the ***Description*** field if necessary.
   - Set the ***I accept Terms of Agreement*** flag to sign the license agreement. This flag is displayed only for Enterprise bundles. To read the agreement text, refer to the ***Terms of Agreement*** link. You can also sign the license agreement on the ***Bundles*** page by clicking the desired product bundle.
   - Click ***Create***.



Create a cluster

3. As a result, the created cluster instance is displayed on the ***Clusters*** page.



The result of successful cluster creation

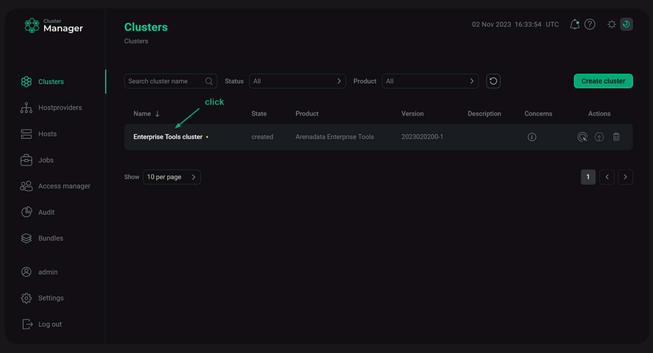
Add services

Daria Barysheva

To Table of Contents

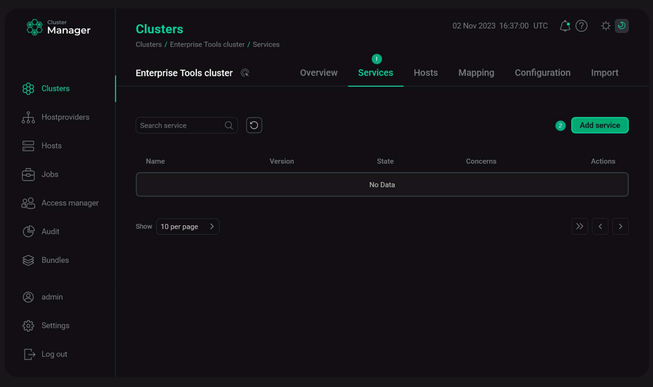
In ADCM a **service** means a software that performs some function. The steps for adding services to a cluster are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



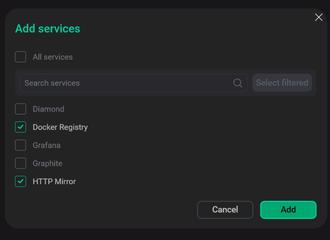
Select a cluster

2. Open the ***Services*** tab on the cluster page and click ***Add service***.



Switch to adding services

3. In the opened dialog, select services that should be added to the cluster and click ***Add***.



Select services

The brief description of available services is listed below.

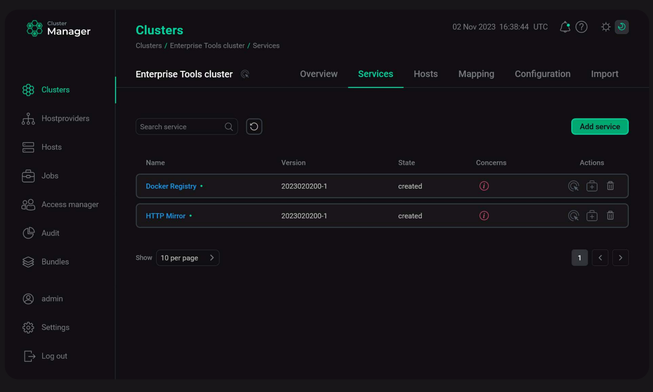
Services that should be added to the Enterprise Tools cluster

Service	Purpose
HTTP Mirror	A service used for hosting packages of the repositories that are needed for offline installation of products
Docker Registry	A service used to store Docker Images that are needed for offline installation of products

NOTE

The CoreDNS service is deprecated — do not select it. The Diamond, Graphite, and Grafana services are used for building monitoring — they will be described separately.

4. As a result, the added services are displayed on the ***Services*** tab.



The result of successful adding services to a cluster

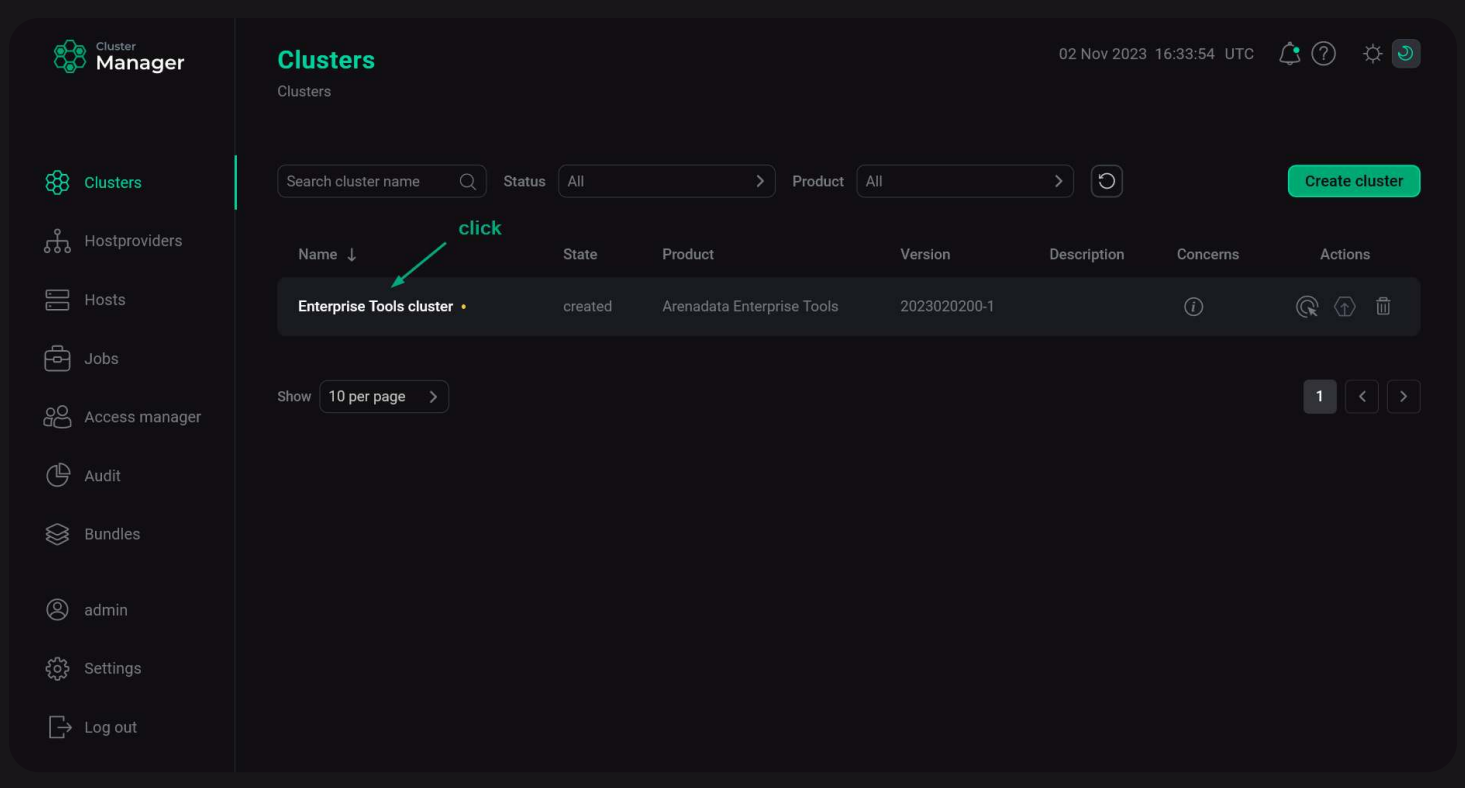
Add hosts to a cluster

Daria Barysheva

To Table of Contents

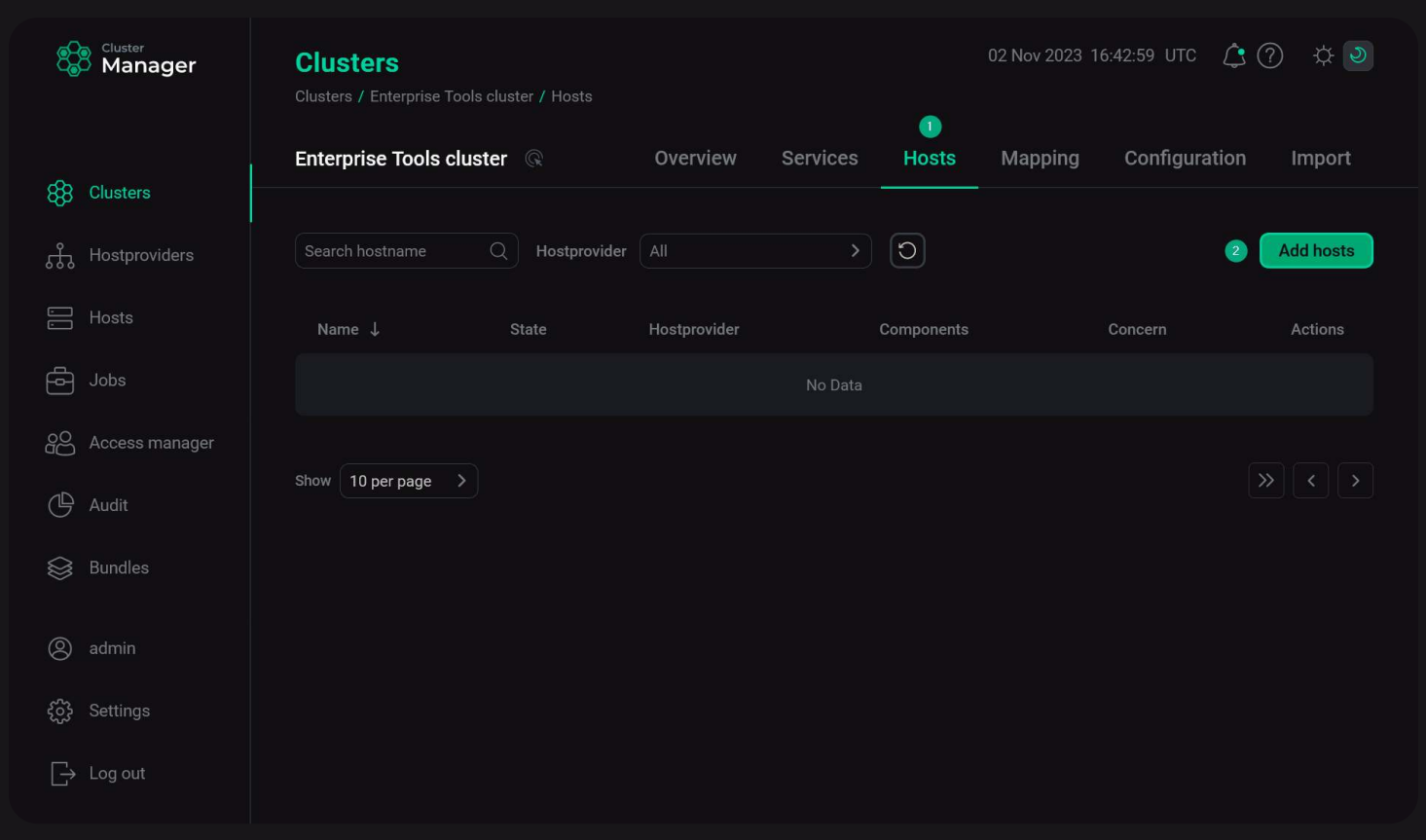
The steps for adding hosts to a cluster are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



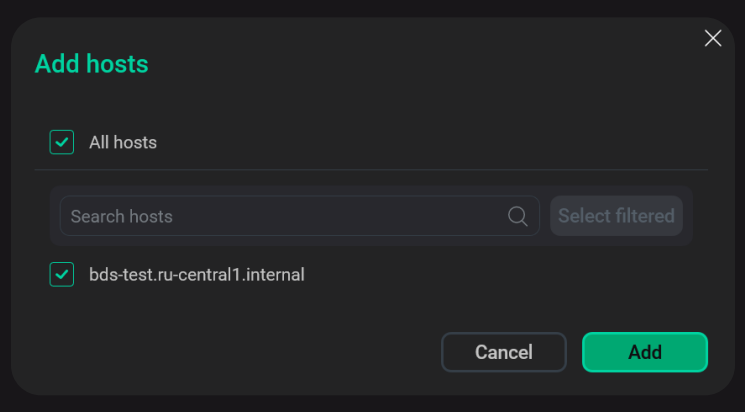
Select a cluster

2. Open the ***Hosts*** tab on the cluster page and click ***Add hosts***.



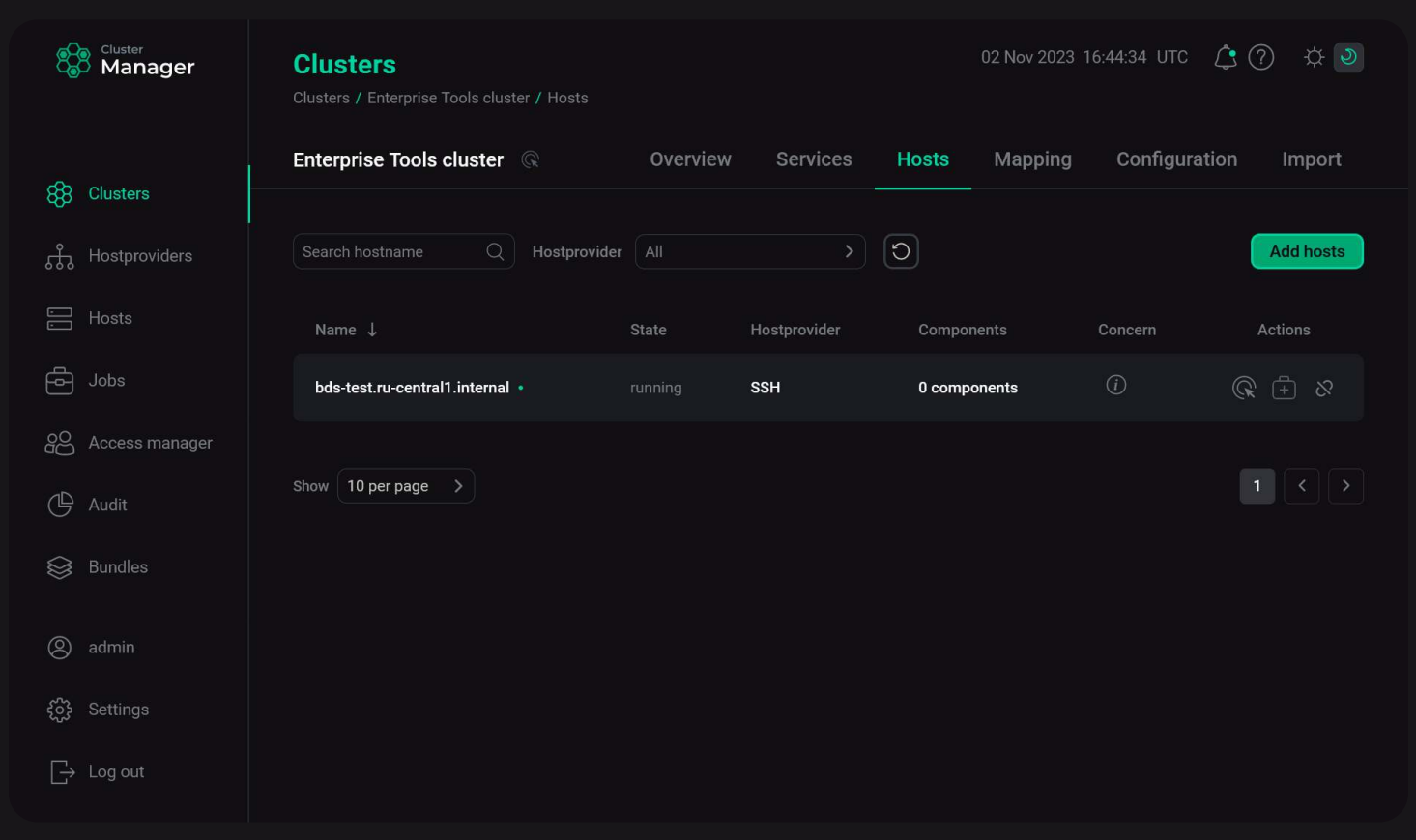
Switch to adding hosts

3. In the opened dialog, select hosts that should be added to the cluster and click ***Add***.



Select hosts

4. As a result, the added hosts are displayed on the ***Hosts*** tab.



The result of successful adding hosts to a cluster

Add components

Daria Barysheva

To Table of Contents

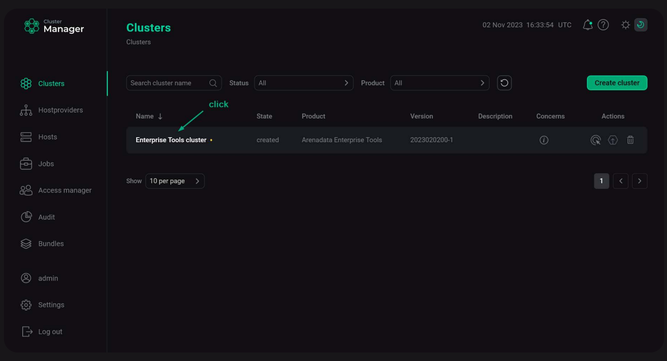
In ADCM a **component** means a part of a service that should be deployed on one or several cluster hosts.

IMPORTANT

Initially, there are no components on any of cluster hosts. You should map components to hosts manually.

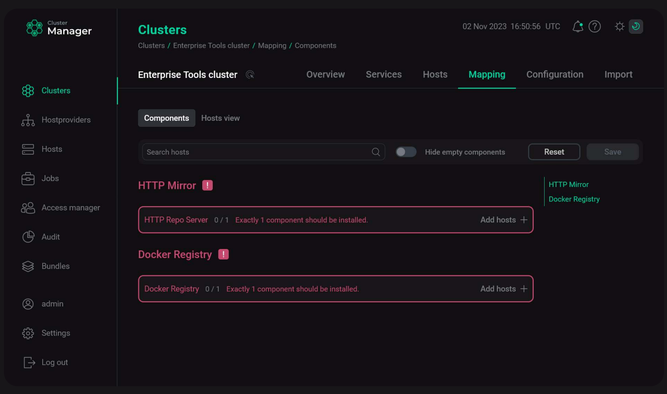
The steps for adding components to hosts are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



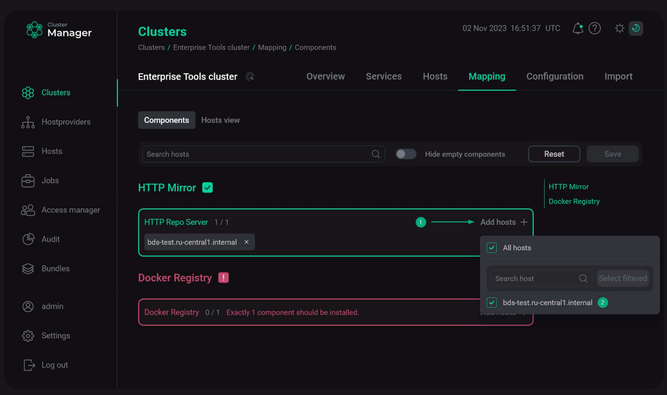
Select a cluster

2. Open the ***Mapping*** tab on the cluster page.



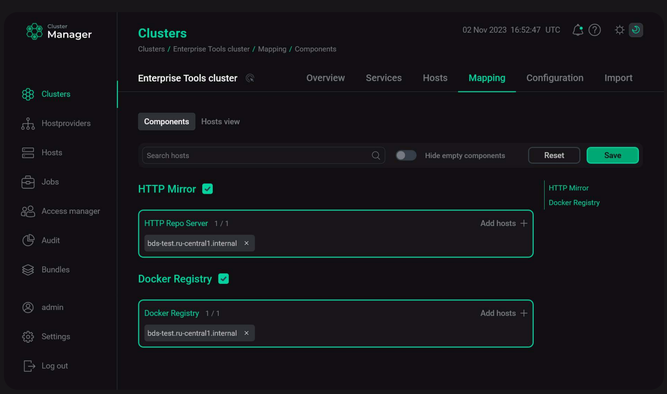
Switch to mapping service components

3. In the window that opens, map components to hosts. To assign hosts for each component, click ***Add hosts*** and select hosts in the pop-up window.



Select hosts for a component

4. After component distribution is completed, click ***Save***.



Save mapping of components

Install a cluster

Sergei Tikhomirov, Daria Barysheva

Contents

To Table of Contents

Step 1. Run installation

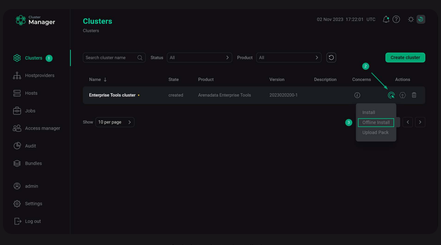
Step 2. Upload packages for offline installation

Step 3. Verify results

Step 1. Run installation

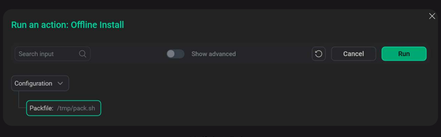
The last step of adding a new Enterprise Tools cluster is to install its services. To do this:

1. Apply the ***Offline Install*** action to the cluster selected on the ***Clusters*** page by clicking the icon in the ***Actions*** column.



Switch to cluster installation

2. In the opened window, enter a full path to the shell script that is received from the Arenadata support team and click ***Run***.

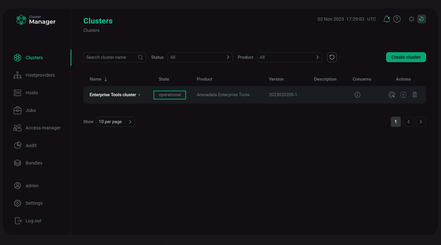


Verify installation

3. Wait until the installation process is completed.

NOTE

In case of errors, you can find logs on the ***Jobs*** page.

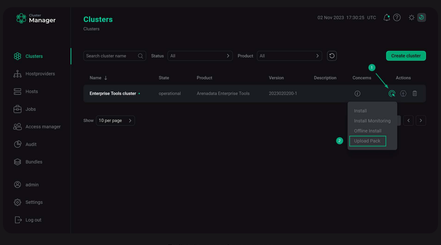


Cluster installation is completed

Step 2. Upload packages for offline installation

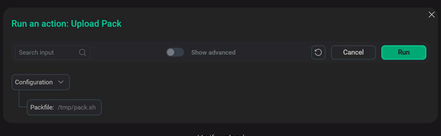
In addition to installing the cluster, you should process the shell script that is received from the Arenadata support team. It is necessary to enable offline installation of products in the future. To do this:

1. Apply the ***Upload Pack*** action to the cluster selected on the ***Clusters*** page by clicking the icon in the ***Actions*** column.



Switch to uploading packages

2. In the opened window, enter a full path to the shell script received from the Arenadata support team and click ***Run***.



Verify upload

3. Wait until the upload process is completed.

NOTE

In case of errors, you can find logs on the ***Jobs*** page.

Step 3. Verify results

The steps for checking the installation results are listed below:

1. Check that ports `443` and `81` are listened at the host where cluster is installed. The first port is used by the Docker Registry service, the second one — by the HTTP Mirror service.

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

The result should look like this:

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

2. Check that the */opt/arenadata/etools/httprepo/arenadata-repo* folder exists at the host where the cluster is installed. This folder should contain the data necessary for offline installation of the products that you have chosen before getting a shell script from the Arenadata support team.

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

The following example displays the data that can be placed in this folder for ADPS installation.

```
total 0
drwxr-xr-x. 5 root root 46 Aug 28 10:33 .
drwxr-xr-x. 3 root root 28 Aug 28 10:33 ..
drwxr-xr-x. 3 root root 17 Aug 28 10:33 ADH
drwxr-xr-x. 3 root root 19 Aug 28 10:33 ADPS
drwxr-xr-x. 3 root root 20 Aug 28 10:33 zookeeper
```

Create a cluster

Sergei Tikhomirov, Daria Barysheva

Contents

To Table of Contents

Step 1. Get the Enterprise version of a cluster bundle

Step 2. Upload a cluster bundle to ADCM

Step 3. Create a cluster based on the uploaded bundle

In ADCM a **cluster** means a set of services running on a set of hosts.

Clusters distributions for ADCM are made in the form of so-called bundles. Generally a bundle is a regular archive that includes a description and software logic of cluster deployment and exploitation.

The steps for adding a cluster to ADCM are listed below.

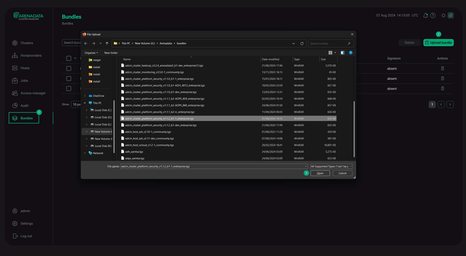
Step 1. Get the Enterprise version of a cluster bundle

Please get a cluster bundle from the Arenadata support team.

Step 2. Upload a cluster bundle to ADCM

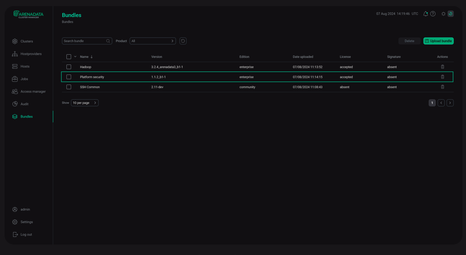
To upload a bundle to ADCM, follow the steps:

1. Select the ***Bundles*** item in the left navigation menu and click ***Upload bundle***.
2. Select a bundle in the Open File Dialog.



Upload a bundle

3. As a result of the performed actions, a bundle is displayed on the ***Bundles*** page.



The result of successful uploading a bundle

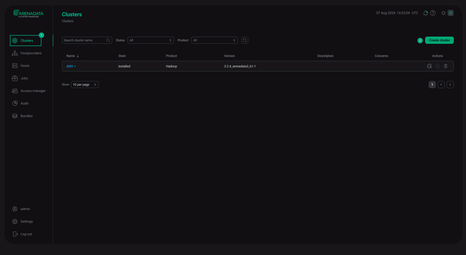
Step 3. Create a cluster based on the uploaded bundle

IMPORTANT

When creating a cluster via the ADCM web interface, you generate a new ADCM cluster instance. It means only adding the cluster to ADCM — at this stage no services are deployed.

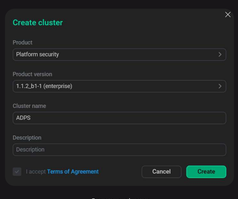
The steps for cluster creation are given below:

1. Select the ***Clusters*** item in the left navigation menu and click ***Create cluster***:



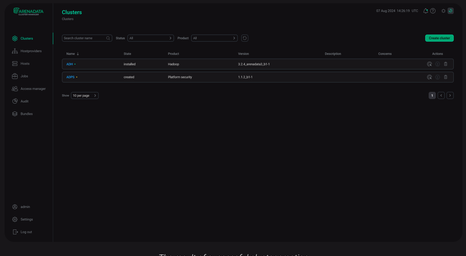
Switch to cluster creation

2. In the opened dialog, describe a new cluster instance:
   - Select an uploaded cluster bundle in the ***Product*** field.
   - Select a bundle version in the ***Product version*** field. Several versions become available in the case of different versions of the same bundle being uploaded.
   - Enter a cluster name in the ***Cluster name*** field.
   - Enter a cluster description in the ***Description*** field if necessary.
   - Set the ***I accept Terms of Agreement*** flag to sign the license agreement. This flag is displayed only for Enterprise bundles. To read the agreement text, refer to the ***Terms of Agreement*** link. You can also sign the license agreement on the ***Bundles*** page by clicking the desired product bundle.
   - Click ***Create***.



Create a cluster

3. As a result, the created cluster instance is displayed on the ***Clusters*** page.



The result of successful cluster creation

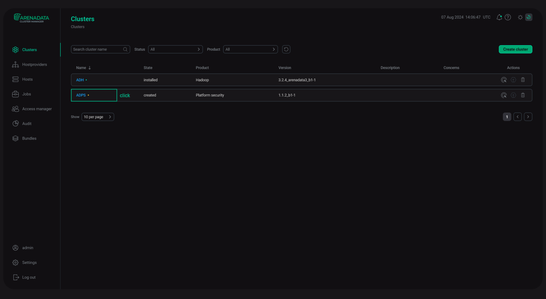
Add services

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

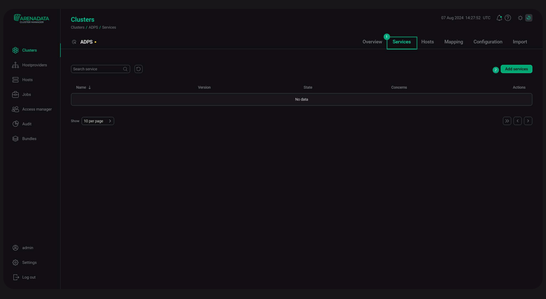
In ADCM a **service** means a software that performs some function. Examples of services for ADPS clusters: Ranger, Knox, MariaDB, etc. The steps for adding services to a cluster are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



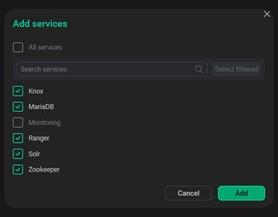
Select a cluster

2. Open the ***Services*** tab on the cluster page and click ***Add service***.



Switch to adding services

3. In the opened dialog, select services that should be added to the cluster and click ***Add***.



Select services

The brief description of available services is listed below.

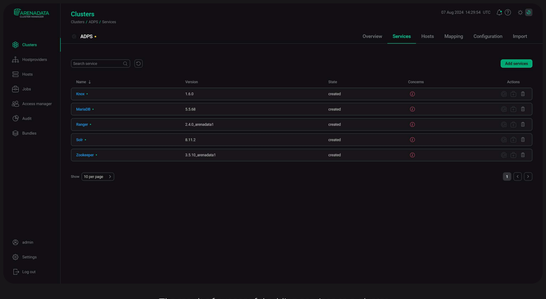
Services that can be added to the ADPS cluster

Service	Purpose
Knox	Reverse proxy service that acts a single access point for HTTP interactions with resources
Maria DB	Maria DB — a relational database created on the base of MySQL and compatible with it. Some MariaDB commands and interfaces are closer to NoSQL, than to SQL. For example, it provides such data storage types as: ColumnStore — for column data storage and distributed architecture support, OQGRAPH — for storing tree and graph structures, etc. Used as a database for Ranger
Monitoring	A service that should be added if monitoring of the ADPS cluster is planned
Ranger	A data security framework used for authorization and policy management
Solr	A search platform based on the Apache Lucene project. Its main features include full-text search, faceted search, highlighting search results, distributed indexing, integration with databases, processing documents with a complex format (Word, PDF, etc.), load-balanced querying, centralized configuration, and others. Used as an audit storage for Ranger
Zookeeper	A centralized coordination service for distributed applications. It is used for failure detection, health monitoring, session management, consensus, etc.

The minimal set of services recommended for ADPS clusters is described below:

- Knox;
- MariaDB;
- Ranger;
- Solr;
- Zookeeper.

4. As a result, the added services are displayed on the ***Services*** tab.



The result of successful adding services to a cluster

NOTE

You can also add services later. The process of adding new services to already running cluster does not differ from installing a service from scratch.

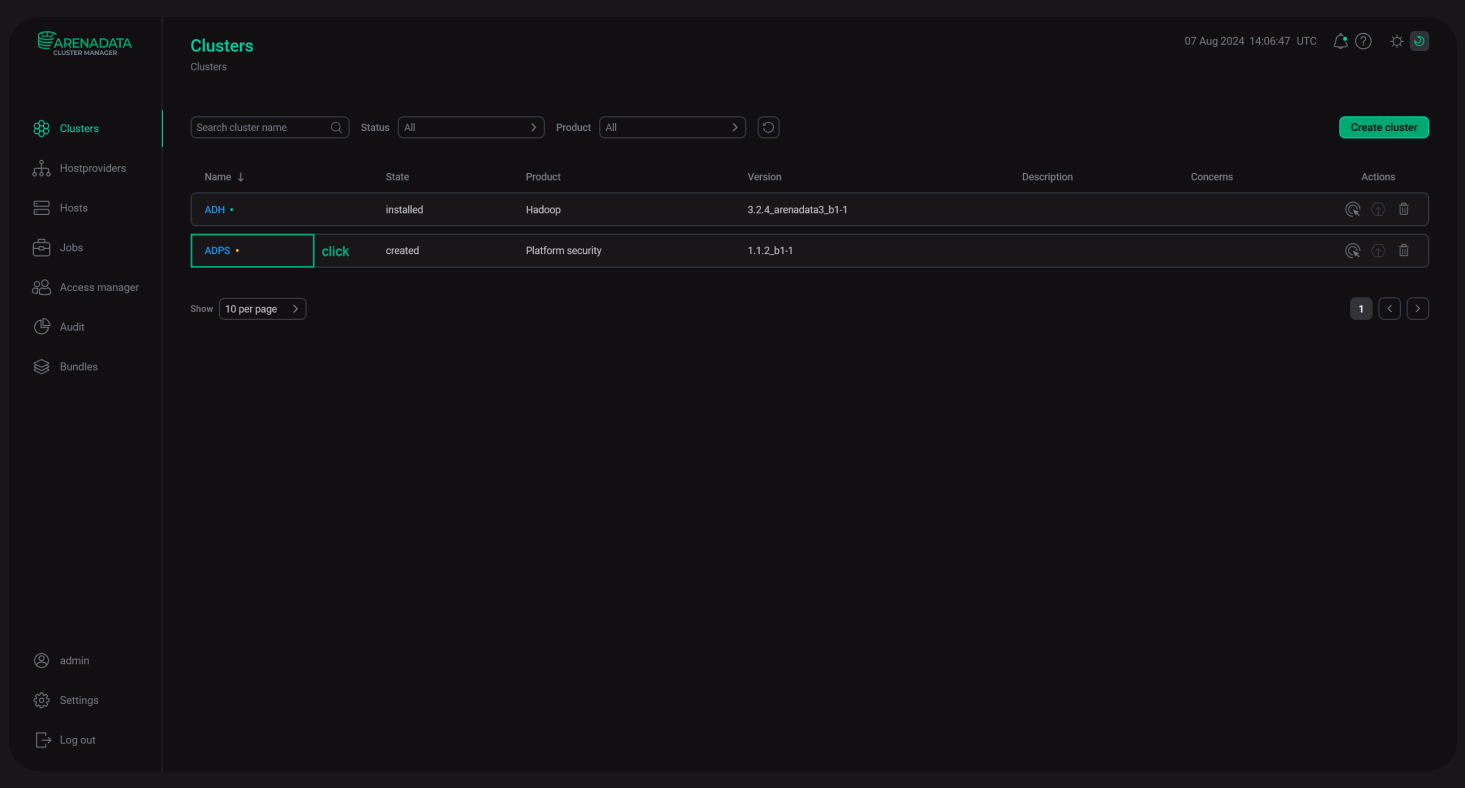
Add hosts to a cluster

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

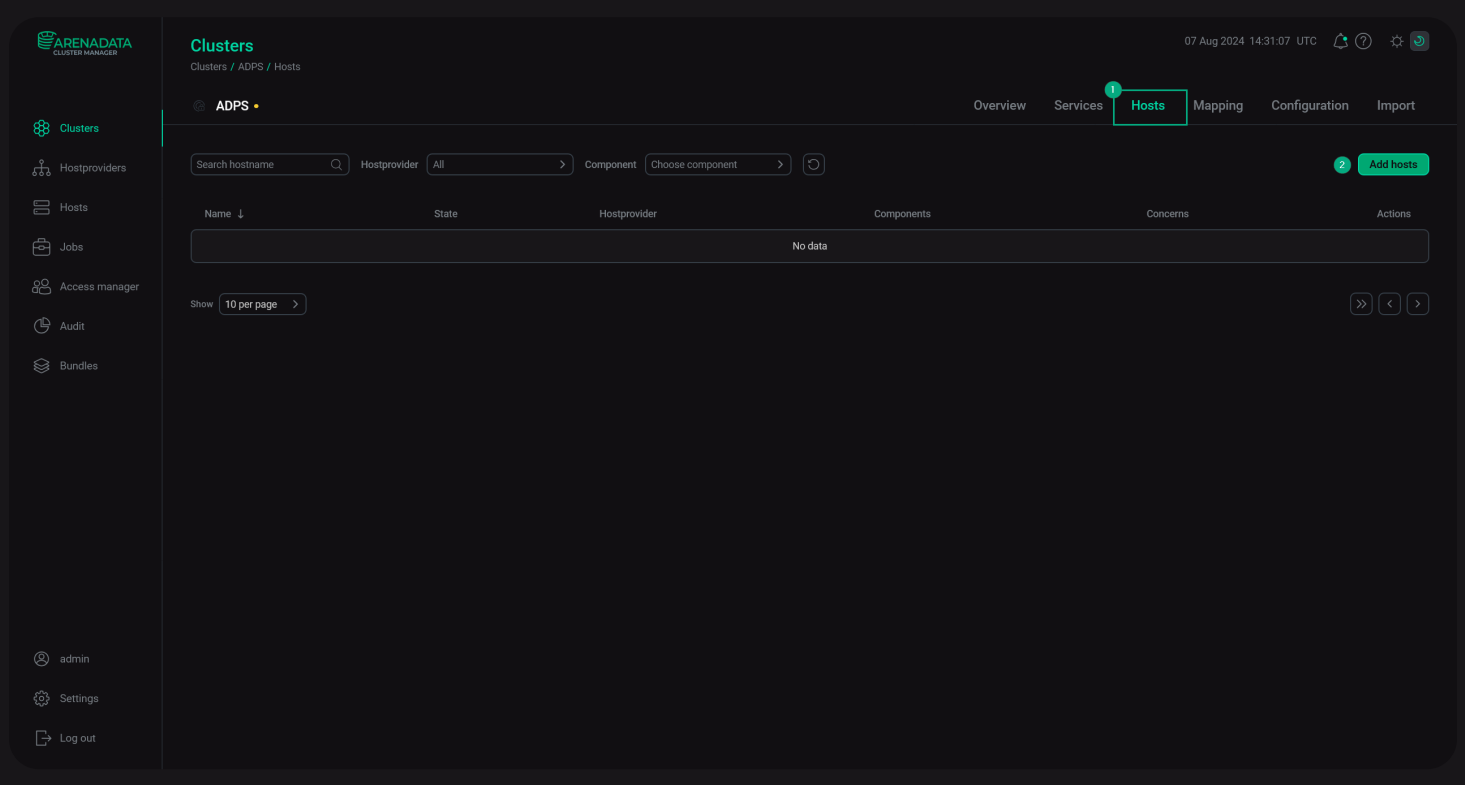
The steps for adding hosts to a cluster are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



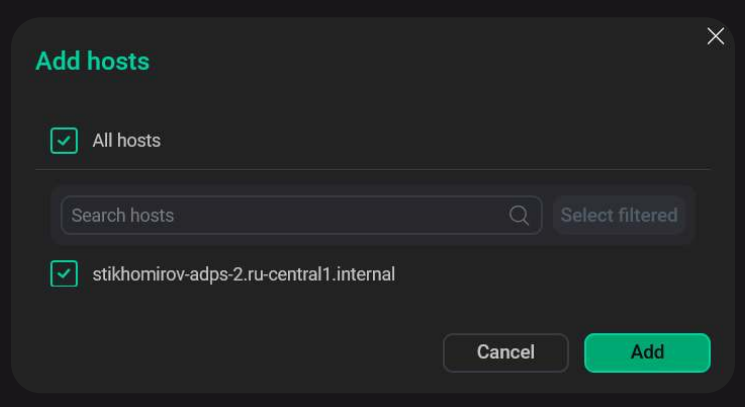
Select a cluster

2. Open the ***Hosts*** tab on the cluster page and click ***Add hosts***.



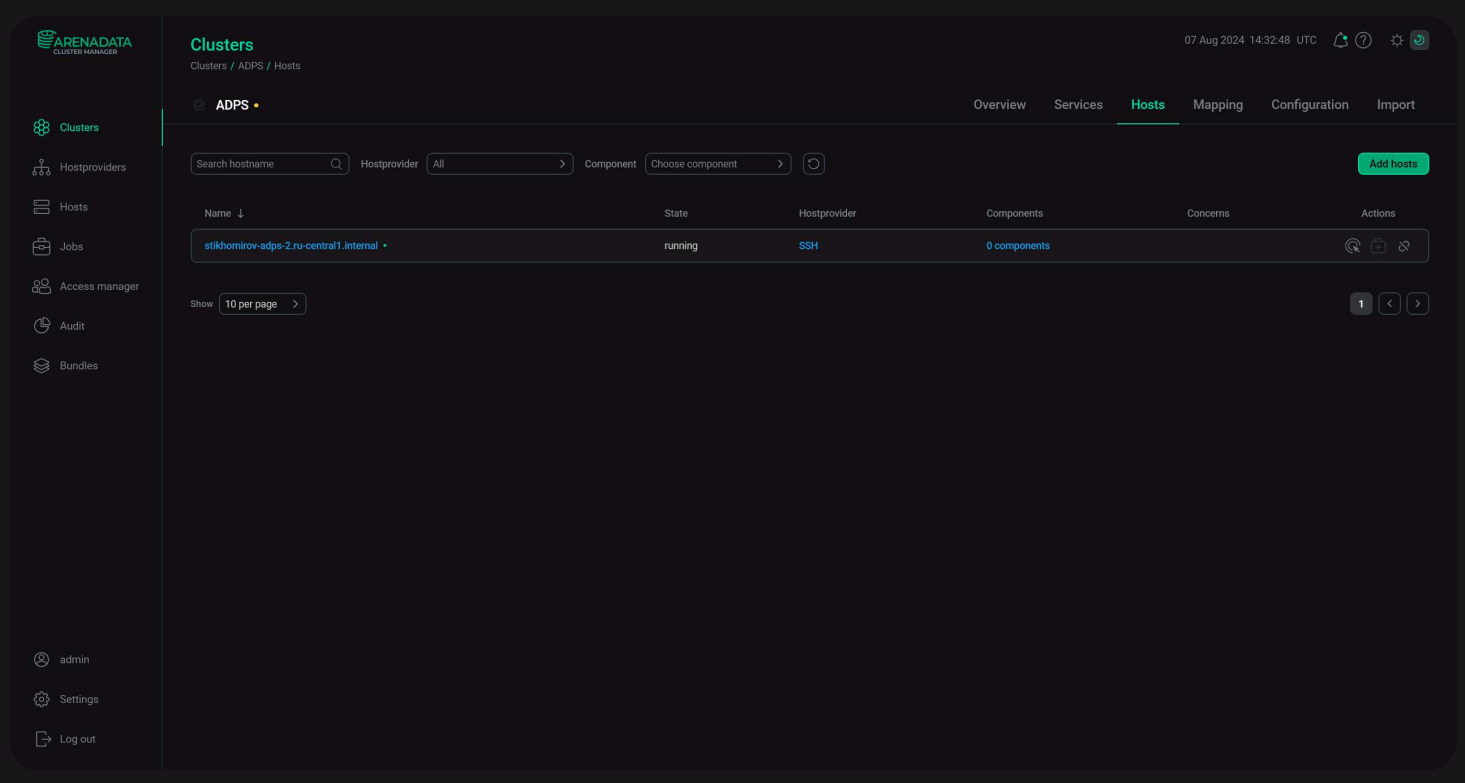
Switch to adding hosts

3. In the opened dialog, select hosts that should be added to the cluster and click ***Add***.



Select hosts

4. As a result, the added hosts are displayed on the ***Hosts*** tab.



The result of successful adding hosts to a cluster

Add components

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

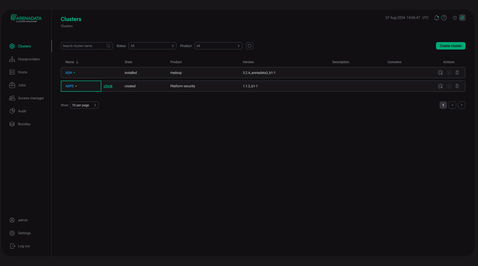
In ADCM a **component** means a part of a service that should be deployed on one or several cluster hosts.

IMPORTANT

Initially, there are no components on any of cluster hosts. You should map components to hosts manually.

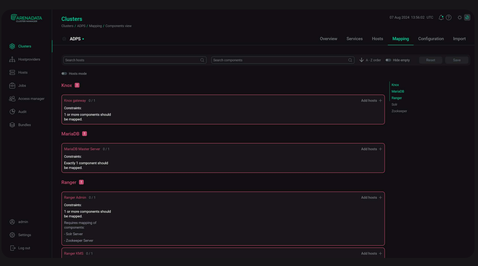
The steps for adding components to hosts are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



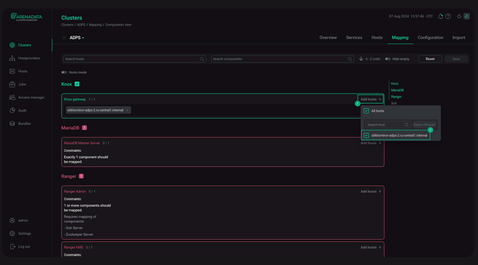
Select a cluster

2. Open the ***Mapping*** tab on the cluster page.



Switch to mapping service components

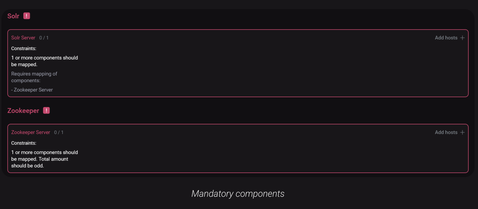
3. In the window that opens, map components to hosts. To assign hosts for each component, click ***Add hosts*** and select hosts in the pop-up window.



Select hosts for a component

Mandatory components

Each service can have mandatory and optional components. Mandatory components are highlighted in red. The ADCM UI also displays how many mandatory components should be installed.



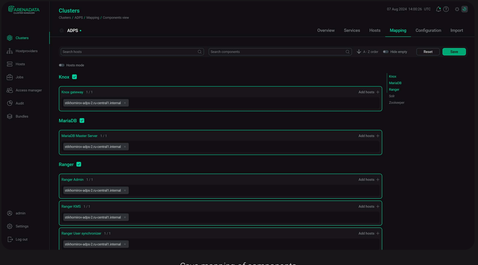
Mandatory components

CAUTION

Without assigning a necessary number of hosts to mandatory components, the component mapping cannot be saved.

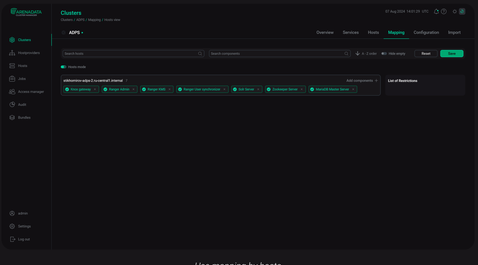
For safety reasons it's recommended to distribute the services among several hosts to ensure fault tolerance, but it is acceptable to configure everything to just one host.

4. After component distribution is completed, click ***Save***.



Save mapping of components

Starting with ADCM 2.2.0, you can use another mapping mode – by hosts. To do this, switch on the ***Hosts mode*** toggle and map components to hosts by clicking ***Add components*** for each host sequentially.



Use mapping by hosts

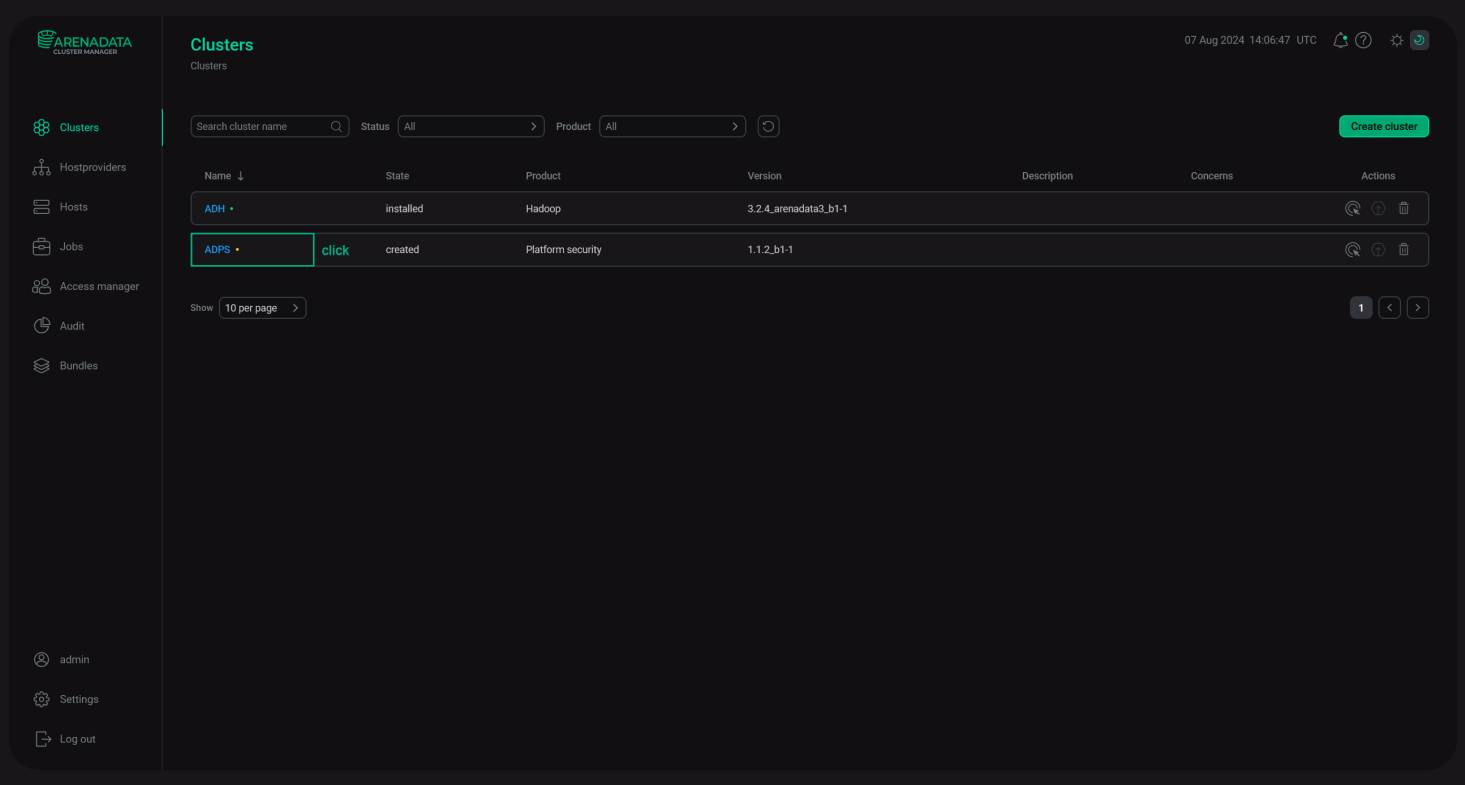
Configure services

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

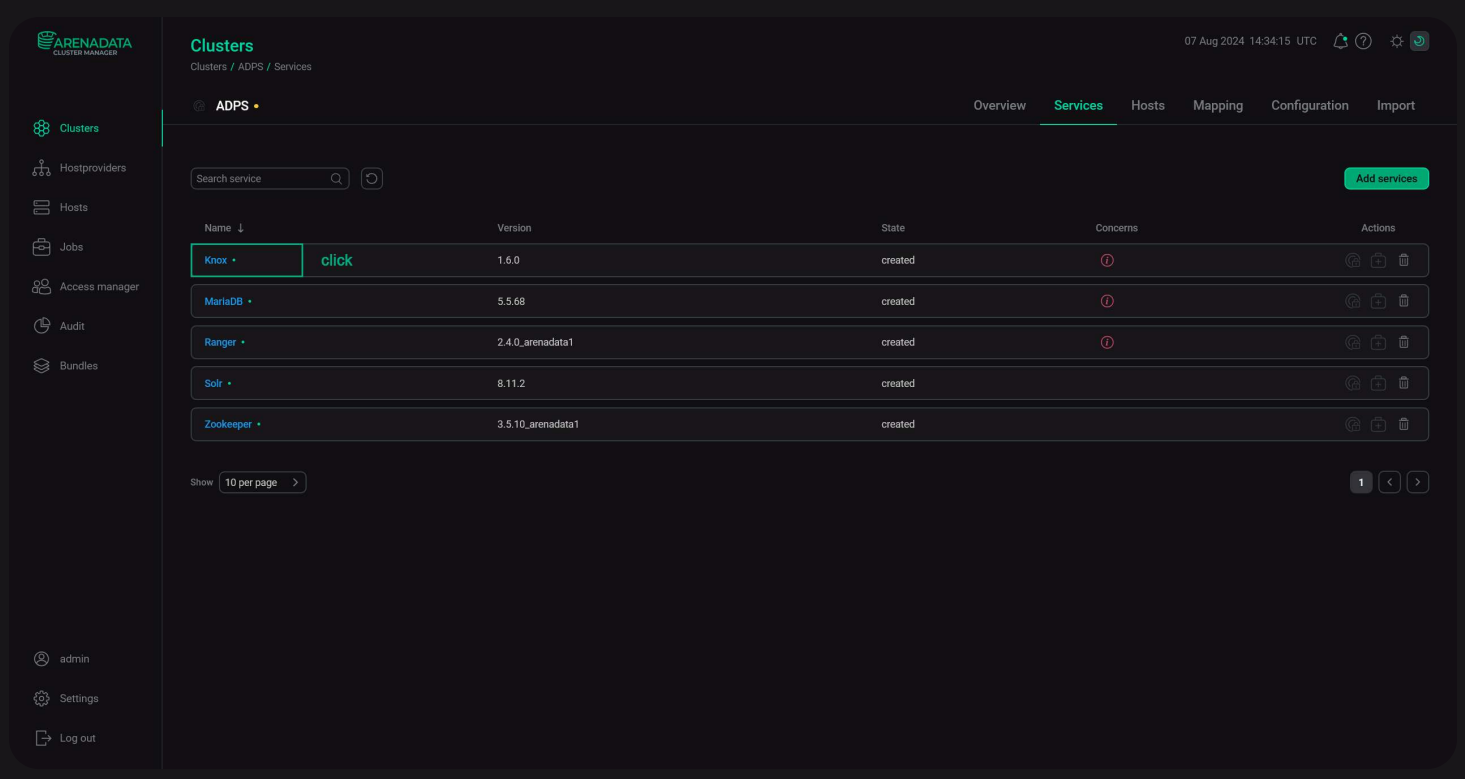
After adding all necessary services, you can configure its parameters. The steps for configuring cluster services are listed below:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



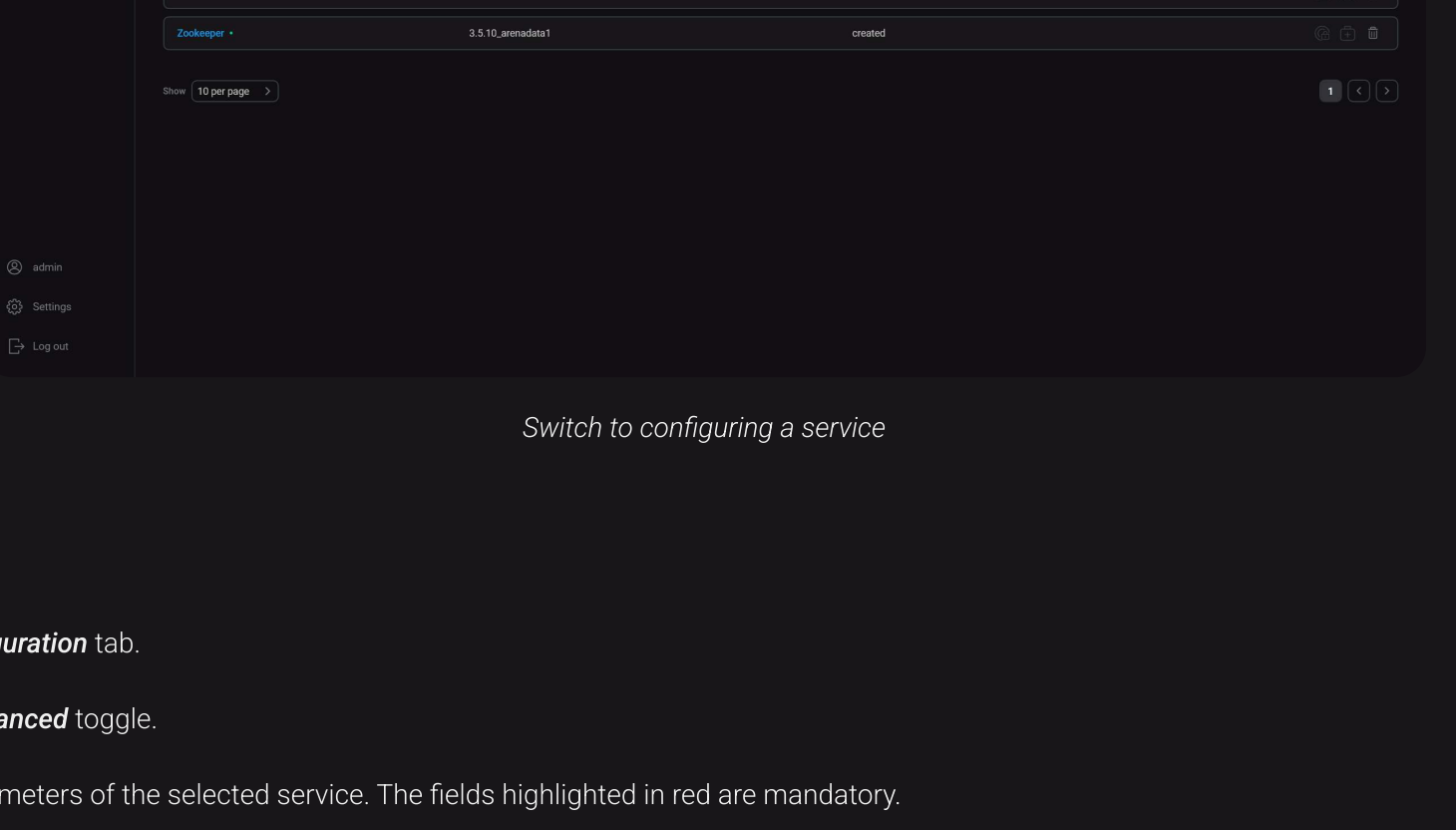
Select a cluster

2. Open the ***Services*** tab on the cluster page, find the row that contains the desired service, and click its name in the ***Name*** column.



Switch to configuring a service

3. In the window that opens:
   - Open the ***Primary Configuration*** tab.
   - Switch on the ***Show advanced*** toggle.
   - Fill in configuration parameters of the selected service. The fields highlighted in red are mandatory.
   - Click ***Save***.



Configure a service

NOTE

For more information about services configuration parameters, see Configuration parameters.

Configure a cluster

Sergei Tikhonov, Daria Barysheva

AD.PS

Contents

To Table of Contents

Primary configuration

Configuration groups

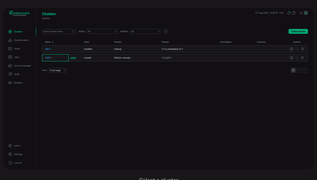
Ansible settings

IMPORTANT

There is usually no need to change cluster configuration parameters. You can leave all parameters at the default values.

After creation a new cluster, you can configure it by performing the following steps:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.



Select a cluster

2. Open the ***Configuration*** tab on the cluster page. The ***Configuration*** tab includes the following sections: ***Primary configuration***, ***Configuration groups***, ***Ansible settings***.
3. Fill in all necessary parameters on the selected tab and click ***Save***.

Primary configuration

The ***Configuration → Primary configuration*** tab allows you to configure the general cluster settings.

Available configuration parameters are listed below:

- Paths to the repositories for CentOS, Red Hat, ALT Linux 8, and Astra Linux that are used in the process of installing — depending on the cluster operation system:
  - ***monitoring*** — monitoring components repository;
  - ***zookeeper_repo*** — Arenadata Zookeeper YUM repository;
  - ***arenadata*** — Arenadata YUM repository.
- The ***Kerberos*** section that includes settings listed in the table below.

Parameter	Description	Default value
Enable Kerberos	Enables the Kerberos authentication	False
Authentication on WEB UIs	Enables the Kerberos authentication on WEB UIs	False
SPNEGO Signature Value	A secret key that is used to generate and verify the signature in the HTTP authentication protocol SPNEGO (in the Base64 format)	—
Kerberos KDC type	A KDC type. For more information, see Kerberos	MIT
KDC hosts	One or more KDC hosts with running FreeIPA server(s)	—
Realm	A Kerberos realm to connect to the FreeIPA server	—
Domains	One or more domains associated with FreeIPA	—
Kadmin server	A host where `kadmin` is running	—
Kadmin principal	A principal name used to connect via `kadmin`, for example `admin@RU-CENTRAL1.INTERNAL`	—
Kadmin password	An IPA Admin password	—
Keytabs directory	A directory with keytab files	/etc/security/keytabs
Additional realms	Additional Kerberos realms	—
Delay between kinit invocation attempts	Time to wait before attempting another kinit operation if the previous one has failed	5
Number of retries for kinit invocation attempts	Number of attempts to launch the kinit command if it fails	10
Trusted Active Directory server	An Active Directory server for one-way cross-realm trust from the MIT Kerberos KDC	—
Trusted Active Directory realm	An Active Directory realm for one-way cross-realm trust from the MIT Kerberos KDC	—
Custom krb5.conf	Enables adding custom parameters to the *krb5.conf* file	False
krb5.conf	Additional parameters to write to the *krb5.conf* file	—
Admin DN	A full DistinguishedName of the admin user with rights to `create/modify/delete/pwdchange` user accounts in the target Organizational Unit	—
LDAP URL	LDAP URL that consists of the `ldap://` or `ldaps://` protocol prefix, hostname or IP address and port of an AD server. For example, `ldaps://192.168.4.2:636`	—
Container DN	A container distinguished name	—
TLS CA certificate Path	A CA certificate path on the host filesystem that is pre-located or should be written from the ***TLS CA certificate*** field	—
TLS CA certificate (optional)	A CA certificate that is written on a host by the path from ***TLS CA certificate Path*** during the execution of the ***Enable Kerberos*** action. The certificate is not saved in ADCM	—
Custom ldap.conf	Enables adding custom parameters to the *ldap.conf* file	False
ldap.conf	Additional parameters that are written to the *ldap.conf* file	—
IpaClient No NTP Autoconf	Disables the NTP configuration during the IPA client installation	False
IpaClient No DNS Lookup	Disables the DNS lookup for the FreeIPA server during the IPA client installation	True
Set up Kerberos utils	Defines whether to install or remove the Kerberos client and utils	True
Configure Kerberos on hosts	Cluster configuration including *krb5.conf* and *ldap.conf*	True
Set up principals and keytabs	Defines whether to create, re-create, or remove principals and keytabs	True
Configure services and clients	Defines whether to update the services and clients configuration	True
Run service checks	Defines whether to run the service checks	True

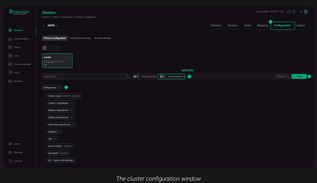
- The ***SSL*** section with the settings listed in the table below.

Parameter	Description	Default value
Enable SSL	Indicates whether SSL is enabled or not	False
Keystore path	A path to the keystore file	/tmp/keystore.jks
Keystore password	A password for the keystore file	—
Truststore path	A path to the truststore file	/tmp/truststore.jks
Truststore password	A password for the truststore file	—
TLS Version	Version of the TLS protocol	TLSv1.2
Run service checks	Defines whether to run the service checks	True

- The ***Advanced SSL*** section with the settings listed in the table below.

Parameter	Description	Default value
Java cacerts file location	Path to the *cacerts* file location of your Java distribution	/etc/pki/ca-trust/extracted/java/cacerts
Java cacerts file password	Password for the *cacerts* file of your Java distribution	—

- The ***java.io.tmpdir*** parameter that specifies an environment variable to control `java.io.tmpdir` for Solr.
- The ***jna.tmpdir*** parameter that specifies an environment variable to control `jna.tmpdir` for Solr.
- The ***Custom JDK settings*** group of parameters specifies the custom Java path.



The cluster configuration window

Configuration groups

The ***Configuration → Configuration groups*** tab is designed to set cluster configuration groups.

Ansible settings

The ***Configuration → Ansible settings*** tab allows you to set Ansible configuration options at the cluster level. The tab is available starting with ADCM 2.2.0.

Ansible settings

Name	Description	Default
forks	The number of parallel processes to spawn when communicating with remote hosts	5

Import ET settings

Sergei Tikhomirov, Daria Barysheva

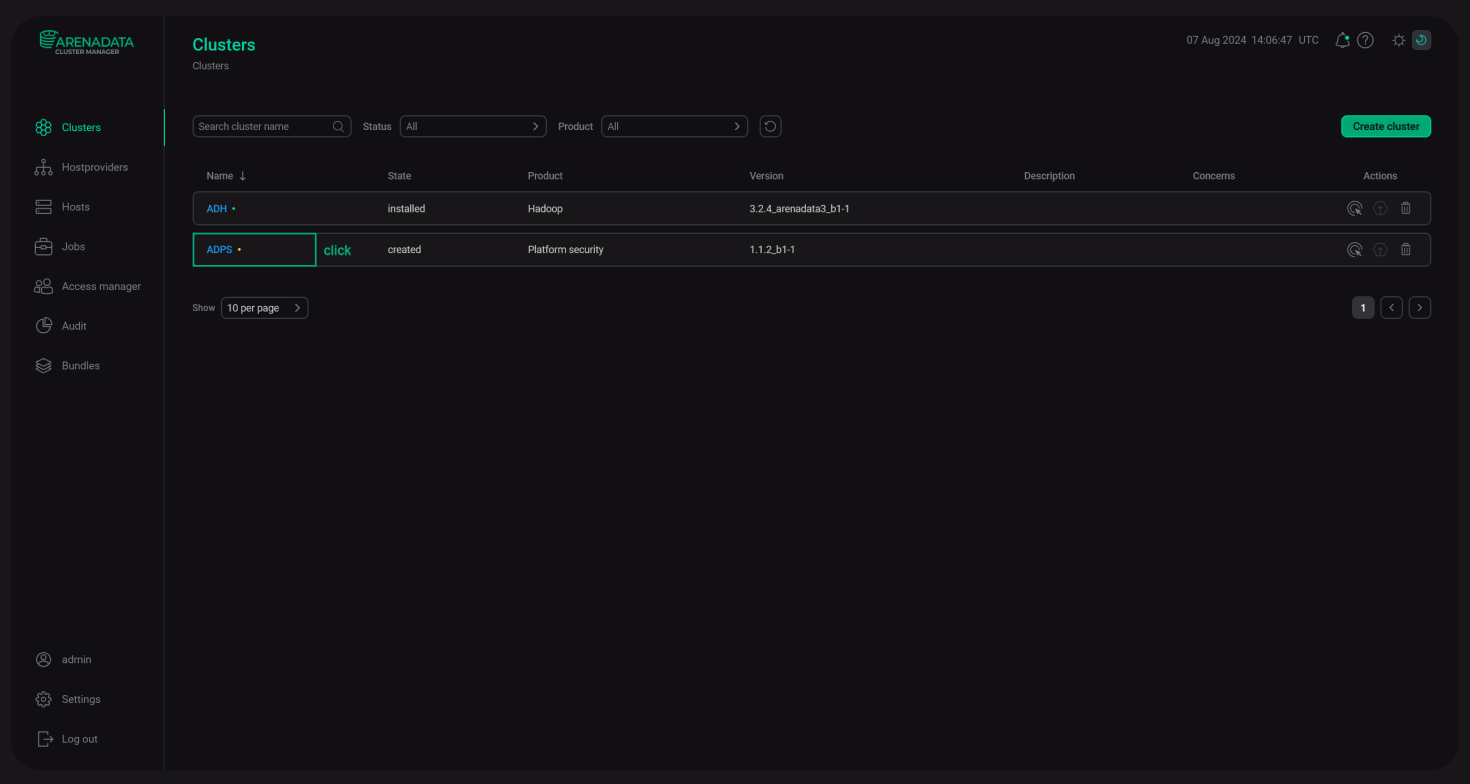
To Table of Contents

IMPORTANT

This step should be performed only in the case of offline installation.

In order to determine that a new cluster should be installed offline, it is necessary to integrate it with the previously created Enterprise Tools (ET) cluster. Follow the steps:

1. Select a cluster on the ***Clusters*** page. To do this, click a cluster name in the ***Name*** column.

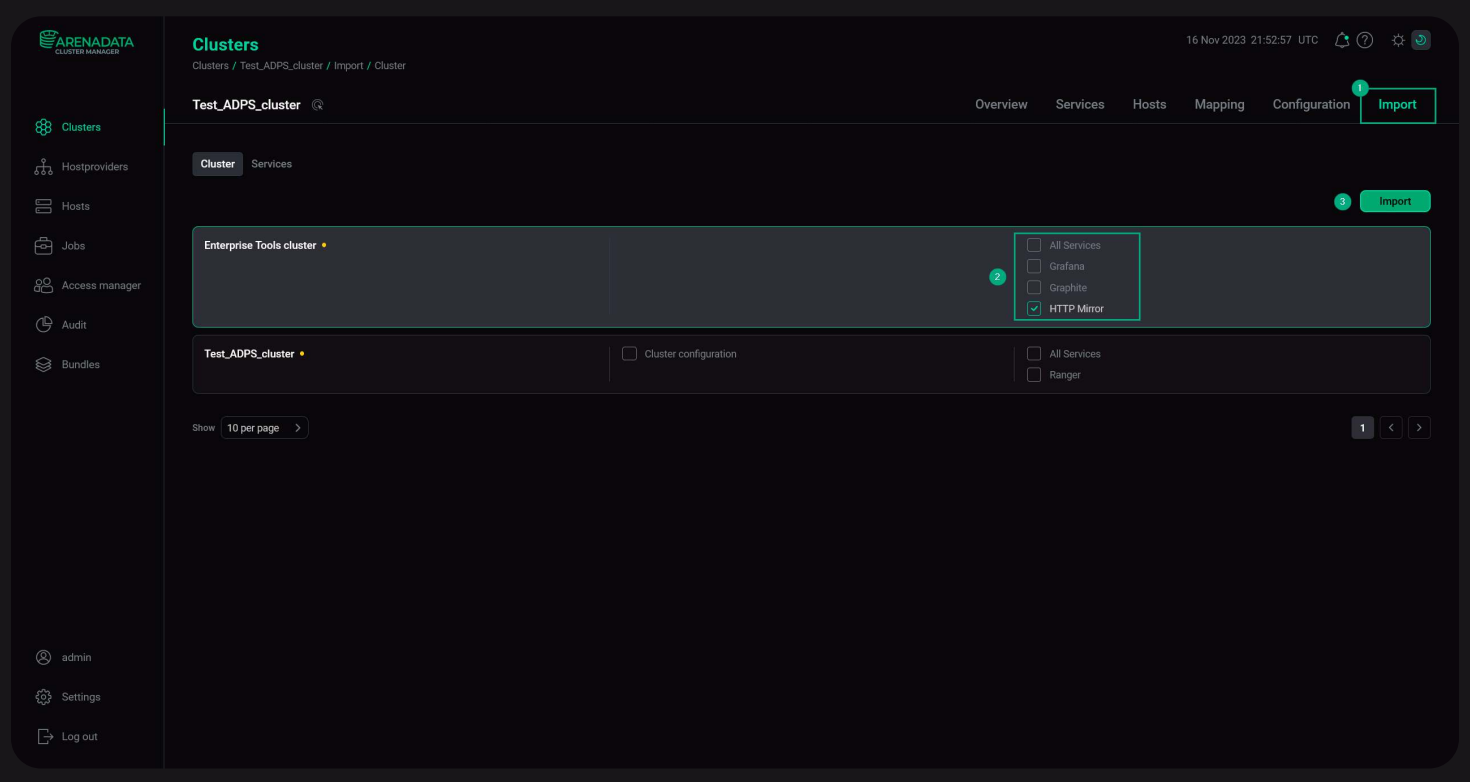


Select a cluster

2. Open the ***Import*** tab in the cluster menu.

3. Select the HTTP Mirror service of the previously created ET cluster.

4. Click ***Save***.



Integrate with the ET cluster

NOTE

After successful import, the data stored locally in the Enterprise Tools cluster can be used at the cluster installation stage. So there is no need to extract this data via Internet.

Install a cluster

Sergei Tikhonkov, Daria Barysheva

AD.PS

Contents

To Table of Contents

Step 1. Installation

- Whole cluster
- Single services

Step 2. View installation results

Step 3. Check the cluster state after the installation

Step 1. Installation

There are two ways to install cluster services:

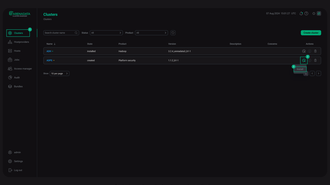
1. **Whole cluster**. In this method all services are installed automatically one by one after applying the ***Install*** action to a cluster.
2. **Single services**. In this method each service is installed manually after applying one or more actions to it. This way is also suitable for installing new services in the already deployed cluster.

In both cases each service starts automatically after installation.

Whole cluster

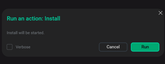
To install all cluster services within a single action, follow the steps:

1. Apply the ***Install*** action to the cluster selected on the **Clusters** page by clicking the icon in the ***Actions*** column.



Switch to the whole cluster installation

2. Verify the requested action in the opened window.



Verify the action

3. Wait until the job is completed.

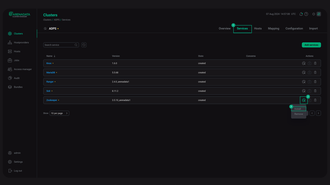
Single services

To install one or several cluster services manually, add these services to the cluster and then perform the actions described below.

CAUTION

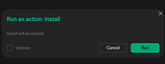
Notice that services are installed one by one. Do not install another service until the installation of the previous one is completed.

1. Open the **Services** tab on the cluster page. For each service that should be installed, click the icon in the **Actions** column and select the ***Install*** action.



Switch to installation of single services

2. Verify the action in the opened window.



Verify the action

3. Wait until the job is completed. Then check that the service state has changed from `created` to `installed`.
4. Repeat the previous actions for other services that should be installed.

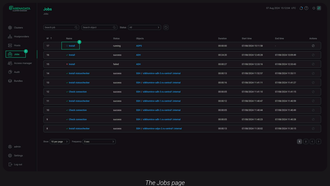
IMPORTANT

When you install services manually, one by one, use the following order:

1. Zookeeper
2. Solr
3. MariaDB
4. Ranger
5. Knox
6. Monitoring

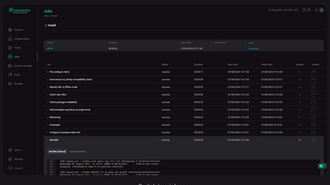
Step 2. View installation results

ADCM starts a single job for installation process, as well as for any other task. You can find out about the status of jobs more specifically on the **Jobs** page.



The Jobs page

The successful completion of the service installation is determined by the transition of the job from the `running` status to the `success` status on the **Jobs** page. If the job fails, it switches to the `failed` status. In this case it is possible to see more detailed information about the errors that occurred by clicking the failed job on the **Jobs** page.



Switch to a job page

A job page contains two sections displayed at the left side of the screen: ***ansible [stdout]*** and ***ansible [stderr]***. These are technical logs of the job that include information from standard I/O streams stdout/stderr. These logs can help to understand what caused the problems.

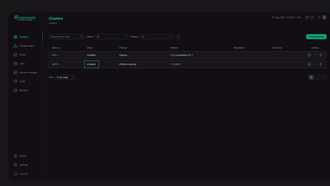
There can also be the optional third section ***ansible [check]***. These are the logs of the most frequent errors checks. The description of these errors is more simple and specific, than in two previous technical logs.

The contents of all three sections should be studied, if errors occur.

Step 3. Check the cluster state after the installation

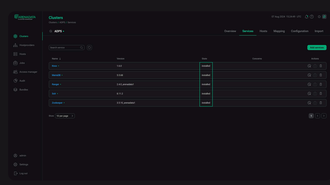
As a result of successful installation, the cluster and its services change their state according to the following rules:

- The cluster state, displayed in the ***State*** field on the ***Clusters*** page, is being changed from `created` to `installed`.



The cluster state after successful installation

- The state of cluster services, displayed on the **Services** tab of the cluster page, is being changed from `created` to `installed`.



The state of cluster services after successful installation

Install monitoring

Sergei Tikhomirov, Daria Barysheva

To Table of Contents

Monitoring is installed optionally (yet strongly recommended). ADCM monitoring is based on three services: Diamond, Graphite, and Grafana.

For **offline** installation, there is no need for a separate monitoring cluster, as the Enterprise Tools (ET) cluster plays its role. The steps for installing monitoring in this case are listed below. You can perform them both before and after the ADPS cluster installation, except the last step Integrate with ADPS cluster, that can be executed only when the ADPS cluster already exists.

- Configure Enterprise Tools cluster
- Integrate with ADPS cluster

Configure Enterprise Tools cluster

A Data Mechanics

Contents

Step 1. Add monitoring services to the ET cluster
Step 2. Map monitoring components
Step 3. Configure monitoring services
Step 4. Install monitoring services
Step 5. Verify results
- Graphite
- Grafana

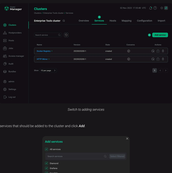
IMPORTANT

Before the subsequent monitoring configuration, make sure that the Enterprise Tools (ET) cluster is installed via ADCM. All the following actions are performed on its base.

Step 1. Add monitoring services to the ET cluster

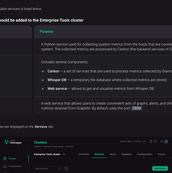
The steps for adding services to a cluster are listed below:

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



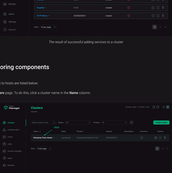
Select a cluster

2. Open the **Services** tab on the cluster page and click **Add service**.



Switch to adding services

3. In the opened dialog, select services that should be added to the cluster and click **Add**.



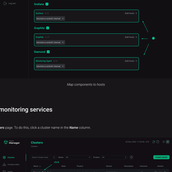
Select services

The brief description of available services is listed below.

Monitoring services that should be added to the Enterprise Tools cluster

Service	Purpose
Diamond	A Python service used for collecting system metrics from the hosts that are connected to the monitoring system. The collected metrics are processed by Carbon (the backend services of Graphite).
Graphite	Includes several components: • **Carbon** – a part of services that are used to process metrics collected by Diamond. • **Whisper DB** – a temporary file database where collected metrics are stored. • **Web service** – allows to get and visualize metrics from Whisper DB.
Grafana	A web service that allows users to create convenient sets of charts, alerts, and other visualization tools for metrics received from Graphite. By default, uses the port **3000**.

4. As a result, the added services are displayed on the **Services** tab.

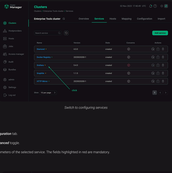


The result of successful adding services to a cluster

Step 2. Map monitoring components

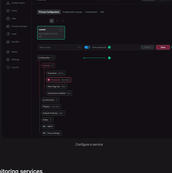
The steps for adding components to hosts are listed below:

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



Select a cluster

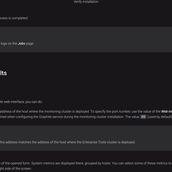
2. Open the **Mapping** tab on the cluster page, map monitoring components to all cluster hosts, and click **Save**.



Map components to hosts

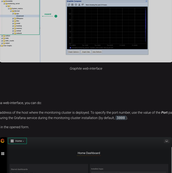
Step 3. Configure monitoring services

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



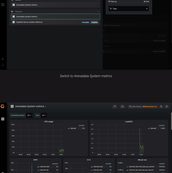
Select a cluster

2. Open the **Services** tab on the cluster page, find the row that contains the desired service, and click its name in the **Name** column.



Switch to configuring service

3. In the window that opens:
   - Open the **Primary Configuration** tab.
   - Switch on the **Show advanced** toggle.
   - Fill in configuration parameters of the selected service. The fields highlighted in red are mandatory.
   - Click **Save**.



Configure a service

Step 4. Install monitoring services

1. Apply the **Install Monitoring** action to the cluster selected on the **Clusters** page by clicking the ⋮ icon in the **Actions** column.

Switch to monitoring installation

2. In the opened window, click **Run**.

Verify installation

3. Wait until the installation process is completed.

NOTE

In case of errors, you can find logs on the **Jobs** page.

Step 5. Verify results

Graphite

To check availability of the Graphite web interface, you can do:

1. In the browser bar, enter an address of the host where the monitoring cluster is deployed. To specify the port number, use the value of the **Web interface TCP port** parameter from the **ip_and_ports** section defined when configuring the Graphite service during the monitoring cluster installation. The value **80** is used by default can be omitted.

IMPORTANT

In case of offline installation, this address matches the address of the host where the Enterprise Tools cluster is deployed.

2. Expand a tree at the left side of the opened form. System metrics are displayed there grouped by hosts. You can select some of these metrics to view their change dynamics in the form of a graph at the right side of the screen.

Graphite web interface

Grafana

To check availability of the Grafana web interface, you can do:

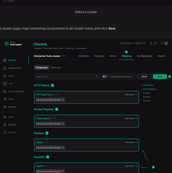
1. In the browser bar, enter an address of the host where the monitoring cluster is deployed. To specify the port number, use the value of the **Port** parameter from the **ip_and_ports** section defined when configuring the Grafana service during the monitoring cluster installation. By default, **3000**.
2. Select the **Home** menu item in the opened form.

Grafana web interface

3. Open one of available dashboards. For example, **Anomalies System metrics**.

Switch to Anomalies System metrics

4. View available graphs.



View Anomalies System metrics

Integrate with ADPS cluster

S. Serga (Tekhnonic Zavod Dzerzhinsk)

Contents

To Table of Contents

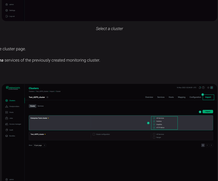
Step 1. Import monitoring settings
Step 2. Add a monitoring service
Step 3. Map monitoring components
Step 4. Install a monitoring service
Step 5. Verify results
- Graphite
- Grafana

In order to determine that a new cluster should be monitored, it is necessary to integrate it with the previously created monitoring cluster and to add a separate Monitoring service into it.

The steps for building integration between a monitoring cluster and ADPS cluster are described below. All actions should be performed on the base of ADPS cluster.

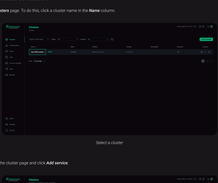
Step 1. Import monitoring settings

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



Select a cluster

2. Open the **Import** tab on the cluster page.
3. Select **Graphite** and **Grafana** services of the previously created monitoring cluster.
4. Click **Import**

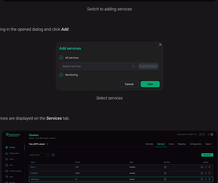


Integration with a monitoring cluster

Step 2. Add a monitoring service

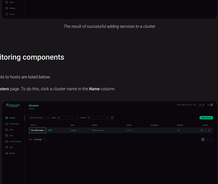
The steps for adding services to a cluster are listed below:

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



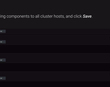
Select a cluster

2. Open the **Services** tab on the cluster page and click **Add service**



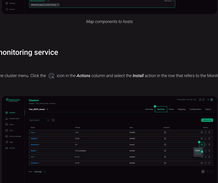
Switch to adding services

3. Select the service Monitoring in the opened dialog and click **Add**



Select services

4. As a result, the added services are displayed on the **Services** tab.

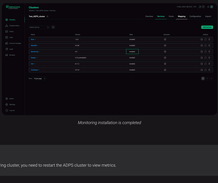


The result of successful adding services to a cluster

Step 3. Map monitoring components

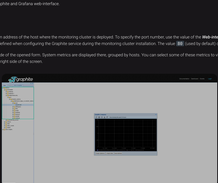
The steps for adding components to hosts are listed below:

1. Select a cluster on the **Clusters** page. To do this, click a cluster name in the **Name** column.



Select a cluster

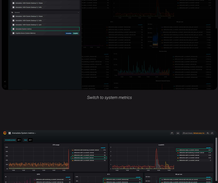
2. Open the **Mapping** tab on the cluster page, map monitoring components to all cluster hosts, and click **Save**



Map components to hosts

Step 4. Install a monitoring service

1. Open the **Services** tab in the cluster menu. Click the ⚙ icon in the **Actions** column and select the **Install** action in the row that refers to the Monitoring service. Verify the action in the opened window.



Switch to monitoring service installation

2. Wait until the job is completed. After successful installation, the monitoring service changes its state from `created` to `installed`

> **NOTE**
> In case of errors, you can find logs on the **Jobs** page.

Monitoring installation is completed

> **CAUTION**
> After you install the monitoring cluster, you need to restart the ADPS cluster to view metrics.

Step 5. Verify results

To ensure that installation process is completed successfully, check that the system metrics are collected from the hosts of your cluster, not only from the host where the monitoring cluster has been deployed earlier.

You can check it both in the Graphite and Grafana web-interface.

Graphite

1. In the browser bar, enter an address of the host where the monitoring cluster is deployed. To specify the port number, use the value of the **Web-interface TCP port** parameter from the **ip_and_ports** section defined when configuring the Graphite service during the monitoring cluster installation. The value `80` (used by default) can be omitted.
2. Expand a tree at the left side of the opened form. System metrics are displayed there, grouped by hosts. You can select some of these metrics to view their change dynamics in the form of a graph at the right side of the screen.

New hosts are available in Graphite

Grafana

1. In the browser bar, enter an address of the host where the monitoring cluster is deployed. To specify the port number, use the value of the **Port** parameter from the **ip_and_ports** section defined when configuring the Grafana service during the monitoring cluster installation (by default, `3000`).
2. Select the **Home** menu item in the opened form.

Switch to system metrics

3. View available graphs.

View system metrics

Connect to Knox via Admin API

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Overview

Get a list of topologies

Add a topology

Overview

Knox provides an Admin REST API to manage the administrator functions of the gateway: topology management, working with descriptors, provider configs, and aliases. This article shows several examples of using the API. To get a full list of available endpoints and methods, refer to the Knox Admin API documentation.

A simple way to create a query is to use curl and it's going to be demonstrated below, but you can construct requests in any convenient way.

A template request looks like this:

```
$ curl -iu <login>:<password> -H <header> -d '<data>' -X <request_type> https://<gateway-host>/gateway/admin/api/v1/<endpoint>
```

where:

- `<login>` — admin username.
- `<password>` — password for `<login>`.
- `<header>` — header to send with a request. It should mark the type of content you send or expect to receive (JSON or XML).
- `<data>` — data to be sent with the request (JSON or XML). Omit if the request type is `DELETE` or `GET`.
- `<request_type>` — type of the HTTP request (`GET`, `POST`, `PUT`, `DELETE`). If the request type is `DELETE`, remove the `-H` parameter from the command.
- `<gateway-host>` — a Knox Gateway host with a port (e.g. *https://stikhomirov-adps.ru-central1.internal:8443*).
- `<endpoint>` — resource to which the request will be sent.

> **NOTE**
>
> If the authentication is not enabled, omit the `<login>` and `<password>` parameters, and use the `-k` flag instead of `-u`.

The `-i` option allows you to see the HTTP headers and check the status of the request, feel free to remove it if deemed unnecessary.

Get a list of topologies

To get a JSON with the Knox topologies, use the following command:

```
$ curl -iu <login>:<password> -H "Accept: application/json" -X GET https://<gateway-host>/gateway/admin/api/v1/topologies
```

The output should be similar to:

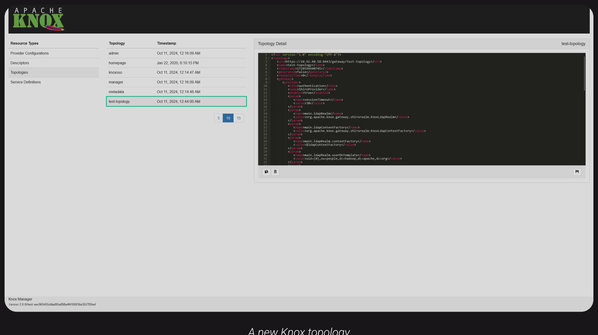
```
HTTP/1.1 200 OK
Date: Thu, 10 Oct 2024 21:28:25 GMT
Content-Type: application/json
Content-Length: 1190

{
   "topologies" : {
      "topology" : [ {
         "name" : "admin",
         "timestamp" : "1728594886988",
         "uri" : "https://10.92.40.58:8443/gateway/admin",
         "href" : "https://10.92.40.58:8443/gateway/manager/api/v1/topologies/admin"
      }, {
         "name" : "homepage",
         "timestamp" : "1579705815000",
         "uri" : "https://10.92.40.58:8443/gateway/homepage",
         "href" : "https://10.92.40.58:8443/gateway/manager/api/v1/topologies/homepage"
      }, {
         "name" : "knoxsso",
         "timestamp" : "1728594887604",
         "uri" : "https://10.92.40.58:8443/gateway/knoxsso",
         "href" : "https://10.92.40.58:8443/gateway/manager/api/v1/topologies/knoxsso"
      }, {
         "name" : "manager",
         "timestamp" : "1728594888074",
         "uri" : "https://10.92.40.58:8443/gateway/manager",
         "href" : "https://10.92.40.58:8443/gateway/manager/api/v1/topologies/manager"
      }, {
         "name" : "metadata",
         "timestamp" : "1728594888537",
         "uri" : "https://10.92.40.58:8443/gateway/metadata",
         "href" : "https://10.92.40.58:8443/gateway/manager/api/v1/topologies/metadata"
      } ]
   }
}
```

Add a topology

To manually add a topology, use the corresponding `PUT` request:

```
$ curl -iu <login>:<password> -H "Content-Type: application/xml" -d '<xml>' -X PUT https://<gateway-
host>/gateway/admin/api/v1/topologies/test-topology
```

You will receive the contents of the XML file as an output of this command. To check that the topology was created, you can refer to the Knox Admin UI.



A new Knox topology

Connect to Knox via CLI

Sergei Tikhomirov

Contents

To Table of Contents

Overview

Work with topologies

Overview

You can connect to Knox via a command line interface (CLI) to manage various sides of the Knox deployment. Its focus is to configure the security artifacts for the gateway instance and topologies (e.g. setting the master secret).

To use the Knox CLI, invoke the *knoxcli.sh* script located in the */usr/lib/knox/bin/* directory on a Knox host:

```
$ sudo bash knoxcli.sh [options]
```

For example, to check the Knox version, execute the command below:

```
$ sudo bash knoxcli.sh version
```

The expected output is:

```
Apache Knox: 2.0.0 (eec965455cfdad95a898a4f4100836e3527f26e4)
```

You can find the full list of available commands in the Knox documentation.

Work with topologies

To get a list of topologies, use the command below:

```
$ sudo bash knoxcli.sh list-topologies
```

The output should be similar to:

```
List of files available in the topologies directory
/usr/lib/knox/conf/topologies
admin
homepage
knoxsso
manager
metadata
test-topology
```

The result of this command can help you provide the correct `cluster` argument for other commands.

For example, to redeploy the `admin` topology, you can run:

```
$ sudo bash knoxcli.sh redeploy admin
```

Knox UI overview

Sergei Tikhomirov

Contents

To Table of Contents

- Overview
- Admin
  - Provider Configurations
  - Descriptors
  - Topologies
  - Service Definitions
- Homepage
  - General Proxy Information
  - Topologies

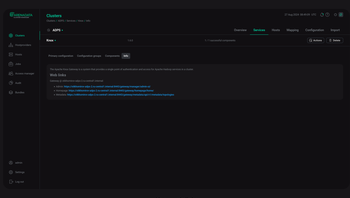
Overview

Knox has two accessible web interfaces: Admin and Homepage. The URLs of these interfaces are presented below, where `<knox_host>` is a URL of an ADPS host, on which Knox is present:

- Admin — `https://<knox_host>:8443/gateway/manager/admin-ui/`
- Homepage — `https://<knox_host>:8443/gateway/homepage/home/`

You can also get direct links to them from ADCM by following the steps below:

1. On the ***Clusters*** page in ADCM, select your ADPS cluster and navigate to the ***Services*** tab.
2. In the list of services, select ***Knox***.
3. Head to the ***Info*** tab. The links are available in the ***Web links*** section.



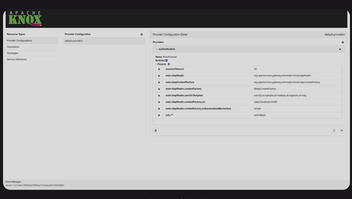
Knox web links

Admin

The Admin web interface provides access to the resource types described below.

Provider Configurations

In this section, you can see and edit the available provider configurations and create a new one by clicking ⊕ next to ***Provider Configuration***. There is a default provider configuration named ***default-providers***.

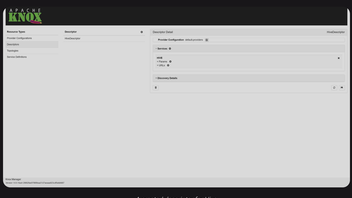


Provider configurations

Here, you can enable/disable a provider. All the parameters in a provider configuration are editable: click on the parameter value, and a text box will appear. You can add a parameter to the configuration by clicking ⊕ next to ***Params***. To remove it, click ✕ on the left from it. To save the edited configuration, click 💾. To revert the changes, click ↻.

Descriptors

This section shows the existing descriptors. A descriptor is a set of service roles that should be proxied with a provider configuration reference. By default, there are no descriptors, but you can create one by clicking ⊕ next to ***Descriptors***.

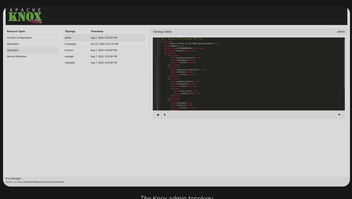


A created descriptor for Hive

The created descriptor has information about its provider configuration, services for which it was created, and discovery details for the target cluster endpoints.

Topologies

This section shows the existing topologies.



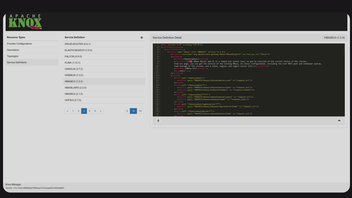
The Knox admin topology

To see the topology details, click the desired topology, and an XML editor will open on the right. Some topologies may be read-only.

> **NOTE**
>
> Manually added topologies disappear after the ***Reload topologies*** ADCM service action.

Service Definitions

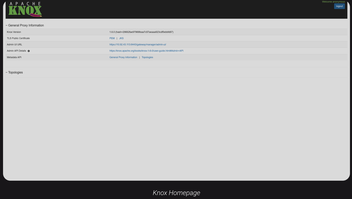
This section shows the existing service definitions. Here you can see and edit service configurations (e.g. change timeout times).



An HBase UI service definition

Homepage

This interface contains general information about proxy and topologies.



Knox Homepage

General Proxy Information

This section provides the following information about the proxy:

- ***Knox Version*** — the current version of Knox.
- ***TLS Public Certificate*** — links to download the public certificate (available in PEM and JKS).
- ***Admin UI URL*** — a link to the Knox Admin interface.
- ***Admin API Details*** — a link to the Knox Admin API documentation.
- ***Metadata API*** — contains two links that allow you to receive the XML file containing metadata (Knox version, Admin UI URL, and Admin API docs link) and the user-created topology information.

Topologies

This section shows a list of user-created topologies.

Enable SSL on ADPS cluster with Knox

Sergei Tikhonov

Contents

To Table of Contents

Prerequisites

Enable SSL

Prerequisites

Before you move on to enable SSL for your ADPS cluster, follow the steps below:

1. Make sure that your hosts have the *truststore.jks* and *keystore.jks* files.
2. Create a certificate for each node using the RSA algorithm. The key length should be 2048 bit, CN should be the FQDN of the corresponding node. It is not recommended to use the DSA algorithm. It is also not recommended to use the ECDSA algorithm due to this algorithm not being widely supported by certification centers.

You can use the script below to generate and place self-signed certificates where they need to be.

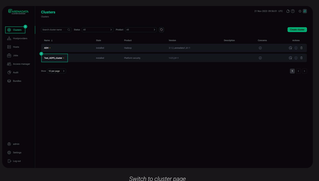
Certificate generation script

```
#!/bin/bash

# Script for generating and import self-signed certificates to java keystore and openssl ca-bundle
# 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 "adps-$i.ru-central1.internal"; done)

echo Generate keystore.jks on each host
for HOST in $HOSTS; do
  echo "Generating keypair"
        ssh $SSH_OPTS $HOST "keytool -genkeypair -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 $HOST "keytool -exportcert -file /tmp/$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 -importcert -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

echo
echo Create and import OpenSSL cert for Nginx
for HOST in $HOSTS; do
        ssh $SSH_OPTS $HOST "sudo openssl req -new -newkey rsa:4096 -days 365 -nodes -x509 \
              -subj \"/C=RU/ST=Denial/L=MSK/O=AD/CN=$HOST\" \
              -keyout /etc/ssl/host_cert.key  -out /etc/ssl/certs/host_cert.cert"
        ssh $SSH_OPTS $HOST "sudo bash -c \"cat /etc/ssl/certs/host_cert.cert >> /etc/pki/tls/c
erts/ca-bundle.crt\""
done
```

Enable SSL

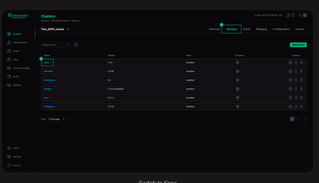
To enable SSL on ADPS cluster, follow these steps:

1. On the ***Clusters*** page, click the name of the installed ADPS cluster.



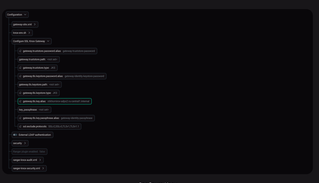
Switch to cluster page

2. Open the the ***Services*** tab and choose ***Knox***.



Switch to Knox

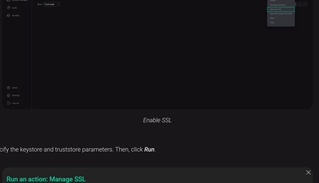
3. In the service menu, choose ***Configuration*** and scroll down to the ***Configure SSL Knox Gateway*** section. Find the ***gateway.tls.key.alias*** parameter and change its default value `gateway-identity` to the actual alias for Knox host (e.g. FQDN of Knox). You can also clear the field (click the icon on the left so that the parameter value becomes `<not set>`) to configure the value automatically. After that click ***Save***.



Configure Knox

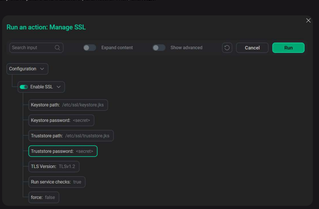
> **TIP**
>
> You can read more about these parameters in the Configuration parameters article in the ***Knox → Configure SSL Knox Gateway*** section.

4. Go to the ***Clusters*** page and choose the ***Manage SSL*** action for your ADPS cluster.



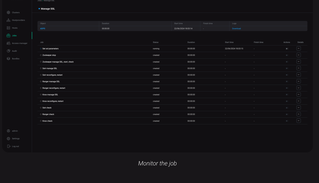
Enable SSL

5. In the pop-up window, specify the keystore and truststore parameters. Then, click ***Run***.



Specify the keystore and truststore parameters

6. You can monitor the process on the **Jobs** page and once it's finished successfully — SSL will be enabled.



Monitor the job

> **NOTE**
>
> If the process fails, try restarting Knox and retrying the failed action.

Ranger Knox plugin

Sergei Tikhomirov

Contents

To Table of Contents

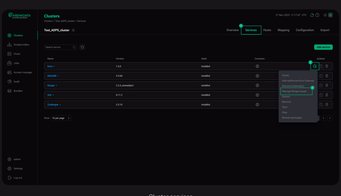
Enable Knox plugin

Add a new policy in Ranger

Enable Knox plugin

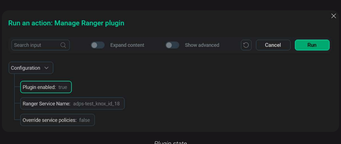
To enable the Ranger Knox plugin, follow the steps below:

1. Go to the ***Clusters*** → ***<ADPS_cluster>*** → ***Services*** page.
2. Find Knox and click the icon in the ***Actions*** column. In the drop-down menu select the ***Manage Ranger plugin*** action.



Cluster services

3. Select the required state of the ***Plugin enabled*** flag. Also, here you can set the name of the Ranger service that will be added. If a service with such name already exists, you can override it by enabling the ***Override service policies*** parameter — in that case, the old service will be deleted and new policies will be generated for this service.



Plugin state

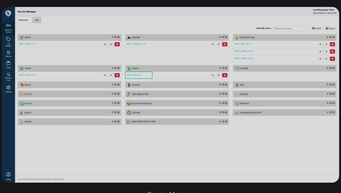
> **NOTE**
>
> On the first plugin enabling, a service with policies will be created (if it doesn't exist yet) regardless of the ***Override service policies*** parameter.

4. Click ***Run*** and confirm the action in the pop-up window.

Add a new policy in Ranger

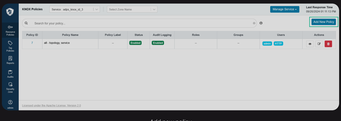
To add a new policy to an existing Knox service, you should perform the following actions:

1. On the ***Service Manager*** page, click an existing service in the Knox pane.



Service Manager

2. On the Knox policy page, click ***Add New Policy***



Add new policy

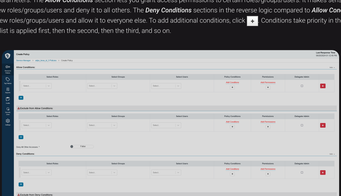
3. On the opened ***Create Policy*** page, fill in the required policy details.

Knox policy details

Policy details parameters

Parameter	Description
Policy Name	The policy name. Must be unique across the system
Enabled	Indicates whether to enable the policy after creation
Normal/Override	Allows you to specify an override policy. When ***override*** state is selected, the access permissions of the new policy override the access permissions in existing policies
Policy Label	Allows grouping of sets of policies with one or more labels and searching for policies by label names. You can use search on the ***Tag Policies***, ***Resource Policies***, and ***Reports*** pages. Also helps to export/import policies. If a user has to export some specific set of policies, then they can search for a policy label and export the specific set of policies
Knox Topology	Knox topology filename
Knox Service	Name of the Knox service
Description	Describes the purpose of the policy
Audit Logging	Enables audit for the policy
Add Validity Period	Allows you to set the lifetime for the policy

4. Set allow/deny condition parameters. The ***Allow Conditions*** section lets you grant access permissions to certain roles/groups/users. It makes sense to use this section if you want to allow access to a few roles/groups/users and deny it to all others. The ***Deny Conditions*** sections in the reverse logic compared to ***Allow Conditions*** — it's useful when you want to deny access to a few roles/groups/users and allow it to everyone else. To add additional conditions, click + . Conditions take priority in the order listed in the policy. The condition at the top of the list is applied first, then the second, then the third, and so on.



Knox allow conditions parameters

Allow Conditions parameters

Parameter	Description
Select Role	Specifies the roles to which this policy applies
Select Group	Specifies the groups to which this policy applies. The public group contains all users, so granting access to the public group grants access to all users
Select User	Specifies a user to which this policy applies (outside an already-specified group) or makes the user an Administrator for this policy
Permissions	Allows you to add or edit permissions
Delegate Admin	Grants administrator privileges to the users or groups specified in the policy. Administrators can edit or delete policies and create child policies based on the existing ones

Deny Conditions parameters

Parameter	Description
Select Role	Specifies the roles to which this policy does not apply
Select Group	Specifies the groups to which this policy does not apply. The public group contains all users, so denying access to the public group denies access to all users
Select User	Specifies a user to which this policy does not apply (outside an already-specified group) or makes the user an Administrator for this policy
Permissions	Allows you to add or edit permissions
Delegate Admin	Grants administrator privileges to the users or groups specified in the policy. Administrators can edit or delete policies and create child policies based on the existing ones

5. Click ***Add*** at the bottom of the page.

ACL in Knox

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Configuration

Example

You can restrict cluster service access with an included service-level authorization mechanism. The implementation features a common access control list (ACL) pattern which implies creating whitelists of users, groups, and IPs that are allowed to access certain resources.

Configuration

To configure access for a service, you need to append the Knox topology (*knox/conf/topologies/admin.xml*) with an authorization provider:

```
<provider>
    <role>authorization</role>
    <name>AclsAuthz</name>
    <enabled>true</enabled>
</provider>
```

You can edit a topology or create a new one via Knox's REST API. This enables the service-level authorization in general, but lacks specific services and the corresponding whitelists.

For example, consider the following topology with a service definition (*knox/conf/topologies/test_topology.xml*):

```
<topology>
   <name>test-topology</name>
   <service>
      <role>WEBHDFS</role>
      <url>http://localhost:50070/webhdfs</url>
   </service>
</topology>
```

To add a whitelist for it, edit the provider section described above the following way:

```
<provider>
    <role>authorization</role>
    <name>AclsAuthz</name>
    <enabled>true</enabled>
    <param>
        <name>webhdfs.acl</name>
        <value>[user list];[group list];[IP list]</value>
    </param>
</provider>
```

where:

- `[user list]` is a comma-separated list of users that are allowed to access the service;
- `[group list]` is a comma-separated list of groups, whose users are allowed to access the service;
- `[IP list]` is a comma-separated list of IP addresses that are allowed to access the service.

IMPORTANT

There are two modes for the ACL. In the default `AND` mode, to get access, a user has to meet all the described conditions (be in `[user list]`, `[group list]`, and have a whitelisted IP address). In this case, if a user is present in the ACL but is not a member of a whitelisted group, they will be denied access.

The alternative is the `OR` mode, which requires a user to satisfy only one of the conditions instead of all of them. To change the operating mode, use the following parameter:

```
<param>
    <name>webhdfs.acl.mode</name>
    <value>OR</value>
</param>
```

In the `AND` mode, you can use the `*` wildcard to allow anyone access, while in the `OR` mode, this wildcard denies access to anyone.

Example

The following example demonstrates how to restrict access to the Solr service in ADPS. To restrict access to any other Hadoop service, follow the same steps.

1. Consider the following topology for the Solr service:

```
<?xml version="1.0" encoding="UTF-8"?>
<topology>
   <uri>https://stikhomirov-adps.ru-central1.internal:8443/gateway/solr</uri>
   <name>solr</name>
   <timestamp>1733257387073</timestamp>
   <generated>true</generated>
   <redeployTime>0</redeployTime>
   <gateway>
      <provider>
         <role>authentication</role>
         <name>HadoopAuth</name>
         <enabled>true</enabled>
         <param>
            <name>config.prefix</name>
            <value>hadoop.auth.config</value>
         </param>
         <param>
            <name>hadoop.auth.config.cookie.domain</name>
            <value>ru-central1.internal</value>
         </param>
         <param>
            <name>hadoop.auth.config.cookie.path</name>
            <value>/</value>
         </param>
         <param>
            <name>hadoop.auth.config.kerberos.keytab</name>
            <value>/etc/security/keytabs/HTTP.service.keytab</value>
         </param>
         <param>
            <name>hadoop.auth.config.kerberos.name.rules</name>
            <value>DEFAULT</value>
         </param>
         <param>
            <name>hadoop.auth.config.kerberos.principal</name>
            <value>HTTP/stikhomirov-adps.ru-central1.internal@AD.RANGER-TEST</value>
         </param>
         <param>
            <name>hadoop.auth.config.simple.anonymous.allowed</name>
            <value>false</value>
         </param>
         <param>
            <name>hadoop.auth.config.token.validity</name>
            <value>1800</value>
         </param>
         <param>
            <name>hadoop.auth.config.type</name>
            <value>kerberos</value>
         </param>
      </provider>
      <provider>
         <role>identity-assertion</role>
         <name>Default</name>
         <enabled>true</enabled>
      </provider>
   </gateway>
   <service>
      <role>SOLR</role>
      <url>http://stikhomirov-adps.ru-central1.internal:8983/solr/</url>
   </service>
</topology>
```

2. Append it with an ACL authorization provider as discussed in the configuration:

```
<provider>
    <role>authorization</role>
    <name>AclsAuthz</name>
    <enabled>true</enabled>
    <param>
        <name>solr.acl</name>
        <value>s_tikhomirov_krb1;*;*</value>
    </param>
</provider>
```

   This is done on a kerberized cluster, so a Kerberos principal name works.

3. Try accessing the Solr service via the Knox proxy:

```
$ curl -ik --negotiate -u : -X GET https://stikhomirov-adps.ru-central1.internal:8443/gateway/solr/solr
```

4. With a correct configuration, you will get an HTML page returned. Otherwise, there will be the `403 Forbidden` error.

Logging in Knox

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Logging settings

Available logs

- Audit logs
- Gateway logs

Logging settings

The Knox logging settings are defined in the log4j 2 format. The Knox Gateway component has two parameters that manage the logging process:

- ***gateway-log4j2.xml template*** — it contains the logging configuration for the gateway and the audit process.
- ***knoxshell-log4j2.xml*** — it contains the logging configuration for the HTTP client.

You can find the default contents of these files in the Knox configuration parameters.

To alter the logging behavior, edit the file parameters according to the log4j 2 logging rules.

Available logs

The Gateway component generates several files with various logging information. All of the mentioned files are stored in the *var/log/knox* directory:

- *gateway-audit.log* — logs from the Audit facility that tracks Knox actions.
- *gateway.log* — logs for the internal gateway operations.
- *gateway.err* — logs that contain error descriptions (e.g. `SLF4J: Class path contains multiple SLF4J bindings`).

Some files may have a date in their name according to the time-based rolling policy that's defined in the logging settings. Viewing logs is available only via CLI, use the following command:

```
$ cat var/log/knox/<log_file>
```

Audit logs

The audit logs are saved in the following format:

```
EVENT_PUBLISHING_TIME
ROOT_REQUEST_ID|PARENT_REQUEST_ID|REQUEST_ID|LOGGER_NAME|TARGET_SERVICE_NAME|USER_NAME|PROXY_USER_NAME|SYSTEM_USER_NAME|ACTION|RE
SOURCE_TYPE|RESOURCE_NAME|OUTCOME|LOGGING_MESSAGE
```

where:

- `EVENT_PUBLISHING_TIME` — the time when a record was written.
- `ROOT_REQUEST_ID` — currently empty.
- `PARENT_REQUEST_ID` — currently empty.
- `REQUEST_ID` — a unique value representing the current request.
- `LOGGER_NAME` — the name of a logger.
- `TARGET_SERVICE_NAME` — the name of a Hadoop service. Can be empty if an audit record is not linked to any Hadoop service, for example, an audit record for topology deployment.
- `USER_NAME` — the name of a user that initiated a session with Knox.
- `PROXY_USER_NAME` — mapped user name.
- `SYSTEM_USER_NAME` — currently empty.
- `ACTION` — the type of action that was executed. The following actions are defined: `authentication`, `authorization`, `redeploy`, `deploy`, `undeploy`, `identity-mapping`, `dispatch`, `access`.
- `RESOURCE_TYPE` — the type of resource for which an action was executed. The following resource types are defined: `uri`, `topology`, `principal`.
- `RESOURCE_NAME` — the name of a resource. For the `topology` resource type, it is the name of a topology. For `uri` — it is inbound or dispatch request path. For `principal` — the name of a mapped user.
- `OUTCOME` — the action result type. Following outcomes are defined: `success`, `failure`, `unavailable`.
- `LOGGING_MESSAGE` — a logging message. Contains tracking information.

For example, here's a log entry with fields mapped:

```
EVENT_PUBLISHING_TIME   25/01/23 12:25:31
ROOT_REQUEST_ID
PARENT_REQUEST_ID       |
REQUEST_ID              |27742083-7b22-4a87-8aa8-9befa06efcdc
LOGGER_NAME             |audit
TARGET_SERVICE_NAME     |
USER_NAME               |
PROXY_USER_NAME         |
SYSTEM_USER_NAME        |
ACTION                  |access
RESOURCE_TYPE           |uri
RESOURCE_NAME           |/gateway/homepage/favicon.ico
OUTCOME                 |success
LOGGING_MESSAGE         |Response status: 404
```

Gateway logs

The gateway logs are saved in the following format:

```
EVENT_PUBLISHING_TIME REQUEST_ID LOG_LEVEL TARGET_SERVICE_NAME LOGGING_MESSAGE
```

where:

- `EVENT_PUBLISHING_TIME` — the time when an event occurred.
- `REQUEST_ID` — a unique value representing the current request.
- `LOG_LEVEL` — a level at which an event occurred. See possible logging levels.
- `TARGET_SERVICE_NAME` — the name of a Hadoop service.
- `LOGGING_MESSAGE` — a logging message. Contains tracking information.

For example, a *gateway.log* record may look like so:

```
2025-01-23 12:25:31,908 27742083-7b22-4a87-8aa8-9befa06efcdc WARN  knox.gateway (GatewayFilter.java:doFilter(202)) - Failed to
match path /favicon.ico
```

Configure topologies in Knox

Sergey Tikhomirov

Contents

To Table of Contents

Load autocreated topologies

Create custom topologies

- Authentication provider
- Descriptor

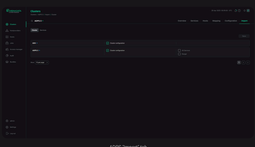
Knox allows you to manually create topologies for your clusters, while another option is to bind clusters in ADCM and load automatically created (**not** generated) topologies. This article describes how to configure topologies both ways.

Load autocreated topologies

NOTE

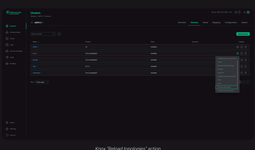
Currently, only ADPS and ADH clusters can be imported into an ADPS cluster. Thus, this topology loading method doesn't suit the ADS users.

1. In ADCM, find your ADPS cluster, select it, and head to the **Import** configuration tab. There, select the ***Cluster configuration*** option for each cluster, topologies for which you want to autoload. Click ***Save***.



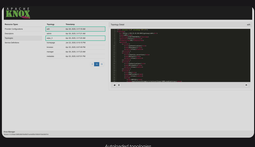
ADPS "Import" tab

2. On the **Services** tab of your ADPS cluster, run the ***Reload topologies*** action for Knox.



Knox "Reload topologies" action

3. Wait for the action to finish and check the Knox Admin UI to see that the necessary topologies have appeared.



Autoloaded topologies

Create custom topologies

In Knox, a topology is defined by it's provider and descriptor. The example below demonstrates how to setup access for some ADH services in the Knox Admin UI. This guide only features the cURL access. For Groovy and DSL access, refer to the Knox documentation.

Authentication provider

NOTE

If you already have a sufficient authentication provider, you may skip this section.

First, set up an authentication provider:

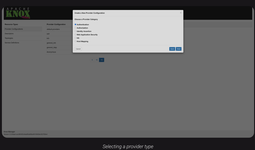
1. On the ***Provider configurations*** page, click ⊕ to the right from the ***Provider Configurations*** list title.

Provider configurations

2. Choose a name for a provider (e.g. `my-provider`) and click ***Add Provider***.

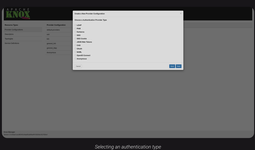
Adding a provider

3. On the opened screen, select the **Authentication** option and click **Next**



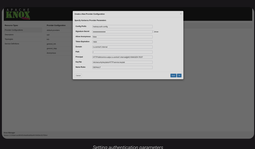
Selecting a provider type

4. Select an authentication type and click **Next**. In this example, it's Kerberos.



Selecting an authentication type

5. Fill in the required information and click **Ok**



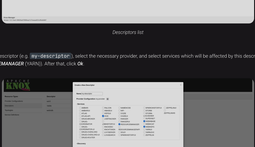
Setting authentication parameters

6. On the final screen, you can configure additional settings for a provider like ACLs.

Descriptor

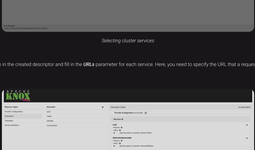
The next step is to prepare a descriptor. If you want to create a topology for several services within a cluster, follow the steps below. If you want to have separate topologies, you'll have to repeat the same steps for each service.

1. On the ***Descriptors*** page, click ⊕ to the right from the ***Descriptors*** list title.



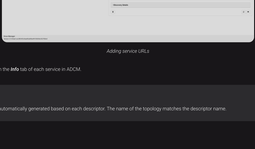
Descriptors list

2. Choose a name for your descriptor (e.g. `my-descriptor`), select the necessary provider, and select services which will be affected by this descriptor (e.g. ***WEBHBASE***, ***HIVE***, ***WEBHDFS***, and ***RESOURCEMANAGER*** (YARN)). After that, click **Ok**



Selecting cluster services

3. Expand the **Services** group in the created descriptor and fill in the **URLs** parameter for each service. Here, you need to specify the URL that a request will be redirected to. Once you're done, click .



Adding service URLs

You can find such URLs on the **Info** tab of each service in ADCM.

NOTE

A read-only topology file is automatically generated based on each descriptor. The name of the topology matches the descriptor name.

Access ADH services via the Knox Gateway

Sergei Tikhomirov

Contents

To Table of Contents

Overview

Service access

- WEBHDFS
- HIVE
- RESOURCEMANGER
- WEBHBASE

Overview

In the Hadoop ecosystem, Knox Gateway, among other things, is used as a reverse proxy — a single point of access to communicate with Hadoop services while keeping the internal topology hidden from potential attackers. Knox accepts requests and redirects them to the appropriate backend services.

To read about the topology creation, see Configure topologies in Knox.

Service access

To test if a service can be access through Knox Gateway, you can run a command of the following format:

```
$ curl -ik --negotiate -u <principal>:<password> -X GET https://<gateway_host>:8443/gateway/my-descriptor/<service>/[<parameters>]
```

where:

- `<principal>` is a Kerberos principal that has access to `<gateway_host>`.
- `<password>` is a password for `<principal>`.
- `<gateway_host>` is a Knox gateway host.
- `<service>` is a service that is defined within the `my-descriptor` topology.
- `<parameters>` are optional parameters that specify the requested resource.

You can find examples for each service below.

WEBHDFS

```
$ curl -ik --negotiate -u <principal>:<password> -X GET https://<gateway_host>:8443/gateway/my-descriptor/webhdfs/v1/tmp?op=LISTSTATUS
```

As a result, you should receive a JSON with the contents of the *tmp* directory.

HIVE

```
$ curl -ik --negotiate -u <principal>:<password> -X GET https://<gateway_host>:8443/gateway/my-descriptor/hive
```

As a result, you should receive HTML code of the main Hive Server UI page.

RESOURCEMANGER

```
$ curl -ik --negotiate -u <principal>:<password> -X GET https://<gateway_host>:8443/gateway/my-descriptor/resourcemanager/v1/cluster
```

As a result, you should receive a JSON with cluster information.

WEBHBASE

```
$ curl -ik --negotiate -u <principal>:<password> -X GET https://<gateway_host>:8443/gateway/my-descriptor/hbase/version/cluster
```

As a result, you should receive a currently installed HBase version number:

```
2.5.10
```

Access ADS services via the Knox Gateway

Sergei Tikhomirov

ADPS

Contents

To Table of Contents

- Overview
- Authentication provider
- Descriptor
- Service access
  - KAFKA
  - NIFI

Overview

In ADPS, Knox Gateway, among other things, is used as a reverse proxy – a single point of access to communicate with target services while keeping the internal topology hidden from potential attackers. Knox accepts requests and redirects them to the appropriate backend services.

In Knox, a topology is defined by its *provider* and *descriptor*. The example below demonstrates how to setup access for the ADS services in the Knox Admin UI. This guide only features the cURL access and UI proxying. For Groovy and DSL access, refer to the Knox documentation.

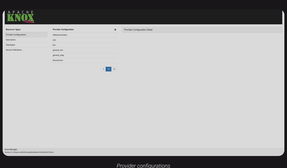
Authentication provider

NOTE

If you already have a sufficient authentication provider, you may skip this section.

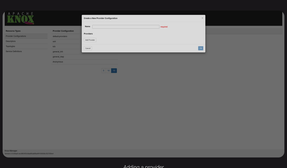
First, set up an authentication provider:

1. On the ***Provider configurations*** page, click ⊕ to the right from the ***Provider Configurations*** list title.



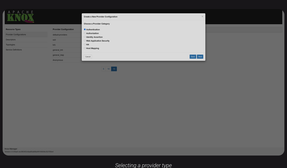
Provider configurations

2. Choose a name for a provider (e.g. `my-provider`) and click ***Add Provider***



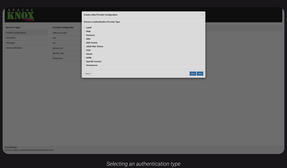
Adding a provider

3. On the opened screen, select the **Authentication** option and click ***Next***.



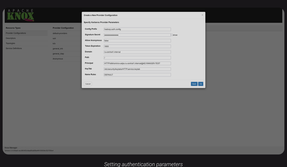
Selecting a provider type

4. Select an authentication type and click ***Next***. In this example, it's Kerberos.



Selecting an authentication type

5. Fill in the required information and click **Ok**.



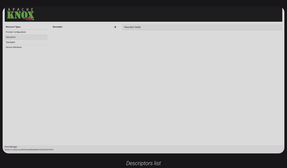
Setting authentication parameters

6. On the final screen, you can configure additional settings for a provider like ACLs.

Descriptor

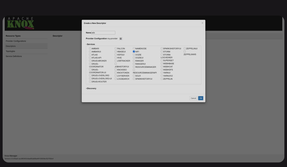
The next step is to prepare a descriptor. If you want to create a topology for several services within a cluster, follow the steps below. If you want to have separate topologies, you'll have to repeat the same steps for each service.

1. On the ***Descriptors*** page, click ⊕ to the right from the ***Descriptors*** list title.



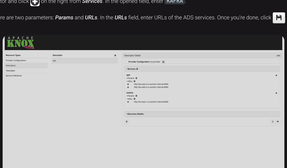
Descriptors list

2. Choose a name for your descriptor (e.g. `ads`) and select the necessary provider. Among the presented services, select ***NIFI*** – it's a service for proxying the UI of NiFi Flow. After that, click **Ok**.



Selecting cluster services

3. Select the created descriptor and click ⊕ on the right from ***Services***. In the opened field, enter `KAFKA`.
4. Below a service name, there are two parameters: ***Params*** and ***URLs***. In the **URLs** field, enter URLs of the ADS services. Once you're done, click ✔



Adding service URLs

By default, the URLs are:

- **KAFKA** – `http://<kafka_rest_host>:8082` (make sure that the ***Kafka REST Proxy*** service is installed).
- **NIFI** – `http://<nifi_server_host_1>:9090` `http://<nifi_server_host_2>:9090`

NOTE

A read-only topology file is automatically generated based on each descriptor. The name of the topology matches the descriptor name.

Service access

To test if a service can be access through Knox Gateway, you can run a command of the following format:

```
$ curl -ik --negotiate -u <principal>:<password> -X GET https://<gateway_host>:8443/gateway/ads/<service>/[<parameters>]
```

where:

- `<principal>` is a Kerberos principal that has access to `<gateway_host>`
- `<password>` is a password for `<principal>`
- `<gateway_host>` is a Knox gateway host.
- `<service>` is a service that is defined within the `my-descriptor` topology.
- `<parameters>` are optional parameters that specify the requested resource.

KAFKA

```
$ curl -ik --negotiate -u <principal>:<password> -X GET https://<gateway_host>:8443/gateway/ads/kafka/topics
```

As a result, you will get a list of topics in Kafka:

```
["mm-connect-offsets","mm-connect-status","mm-connect-configs","__schemas"]
```

NIFI

Open your web browser and enter `https://<knox_host>:8443/gateway/ads/nifi-app/nifi` in the search bar. This will lead you to the NiFi Flow interface.

Knox service management via ADCM

Sergei Tikhomirov

To Table of Contents

The ADCM UI provides actions to manage the Knox service and its components. For information on how to run service actions, refer to ADPS service actions.

The actions available for the Knox service are listed in the table below.

Action	Description
Check	Runs service-specific tests to check the health of the service and its components
Start	Starts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied on the service startup. Otherwise, the service starts without applying configurations from ADCM
Stop	Stops the service
Restart	Restarts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied during the service restart. Otherwise, the service restarts without applying configurations from ADCM
Remove	Removes the service from the cluster. This action should be used to remove already installed services. Whereas the control can be used to remove a non-mapped service (a service which components have not been distributed among cluster hosts)
Manage Ranger plugin	Enables or disables the Ranger Knox plugin
Add/Remove components	Reconfigures the service topology. With it, you can change the amount of the Knox Gateway components. Running this action opens the host-component mapping interface where you can add new service components
Reload topologies	Reloads Knox topologies
Update Knox passwords	Updates passwords in the secured JCEKS files for Knox. Run this action if the Knox master password or some passwords in the Knox configuration were changed in ADCM. All passwords will be applied from the ADCM configuration, no custom passwords (e.g. from custom topologies) will be written to the credential store
Validate configuration	Checks ports for conflicts

CAUTION

In order to protect custom topology settings from overwriting, Knox topologies don't get automatically reloaded after a service is imported or during ADPS reconfiguration actions. In some cases, this may lead to unavailability of target services via Knox. You have to run the ***Reload topologies*** action manually in the following cases:

- A new service was imported into Knox.
- A new connection setting (SSL or Kerberos) was configured for one of the imported services.

MariaDB service management via ADCM

Sergei Tikhomirov

To Table of Contents

The ADCM UI provides actions to manage the MariaDB service and its components. For information on how to run service actions, refer to ADPS service actions.

The actions available for the MariaDB service are listed in the table below.

Action	Description
Check	Runs service-specific tests to check the health of the service and its components
Start	Starts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied on the service startup. Otherwise, the service starts without applying configurations from ADCM
Stop	Stops the service
Restart	Restarts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied during the service restart. Otherwise, the service restarts without applying configurations from ADCM
Remove	Removes the service from the cluster. This action should be used to remove already installed services. Whereas the control can be used to remove a non-mapped service (a service which components have not been distributed among cluster hosts)
Move	Moves a MariaDB Master Server from one host to another. Running this action opens the host-component mapping interface where you can move MariaDB components

OpenBao UI overview

Angel Feliciano

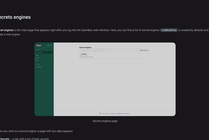
Contents

- Overview
- Secrets engines
- Access
- Policies
  - ACL policies
- Tools
- Seal OpenBao

Overview

Each OpenBao Server instance has an available web interface. You can find the links to them in SCOM:

1. On the **Clusters** page, select your AZPS cluster and head to the **Services** tab.
2. At the list of services, select **OpenBao**.
3. Head to the **URL** tab. The links are available in the **Web links** section.



OpenBao web links in case of two OpenBao Server instances

Secrets engines

Secret engines is the main page that appears right after you log into the OpenBao web interface. Here, you see the list of all secret engines (**cubbyhole**) is created by default) and create a new one.

Secrets engines page

When you click on a secret engine, a page with two tabs appears:

- **Secrets** — a page with a list of kept secrets
- **Configuration** — a tab with the engine configuration, where you can find such information as the engine's path, description, accessor, and default lease TTLs

Secret engine secrets

Also, each page has a sidebar which has two buttons on top:

- **CLI** — opens a OpenBao Browser CLI, which is essentially a shell where you can execute OpenBao's commands
- **Help** — if enabled your username, allows you to view your access tokens, and to log out of the system

Access

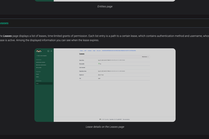
Authentication methods

The **Authentication methods** page displays a list of the available authentication methods. By default, it's only **Token**, which allows users to log in using a secret token. Other possible methods are **Username**, **LDAP**, **OIDC** and **RADIUS**. Clicking on an authentication method opens a page where you can view or change the method's configuration.

Authentication methods page

Multi-factor authentication

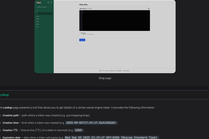
The **Multi-factor authentication** page displays configured MFA methods and allows you to configure a new one. The available methods are **TOTP** (time-based one-time password), **Duo**, **Okta**, and **PingID**. If you have MFA methods configured, a list of methods will be displayed.



Multi-factor authentication page with no configured methods

If you have at least one method configured, you will see a page with two tabs:

- **Methods** — list of configured authentication methods. If a method (e.g., **TOTP**) is configured with different settings, it will be displayed multiple times.
- **Enforcements** — a list of enforcements that define which authentication types, authentication accounts, groups, and/or entities will require an MFA method.



Multi-factor authentication page with two configured TOTP methods

OIDC provider

The **OIDC provider** page displays which applications can OpenBao act as an OpenID Connect (OIDC) provider and allows you to configure an application to do so. If you have no apps configured, a short instruction will be displayed.

Empty OIDC provider page

If you have at least one application configured, you will see a page with five tabs:

- **Applications** — a list of applications that have OpenBao as an OIDC provider
- **Keys** — a list of keys to sign and verify the JSON Web Tokens
- **Assignments** — a list of assignments, parameters that limit which OpenBao users are allowed to authenticate.
- **Providers** — a list of OIDC providers, mostly, there is only the **default** provider, which can be altered but not deleted.
- **Scopes** — a list of scopes. Provides map reference a set of scopes to make specific identity information available as claims.

OIDC provider page

Groups

The **Groups** page displays two tabs:

- **Groups** — a list of available groups (both internal and external)
- **Aliases** — a list of aliases for external groups

On this page, you can also create a new group as well as existing one. If you click on an external group, you will be able to create an alias for it.

Groups page

Entities

The **Entities** page displays two tabs:

- **Entities** — a list of available entities
- **Aliases** — a list of aliases for entities

On this page, you can create a new entity with an existing one, or merge one entity into another. If you click on an entity, you will be able to create an alias for it.

Entities page

Leases

The **Leases** page displays a list of leases, time-limited grants of permission. Each list entry is a path to a certain lease, which contains authentication method and expiration, whose more or active. Among the necessary information you can see when they'll be lose expiry.

Lease details on the Leases page

Policies

ACL policies

The **ACL policies** page displays a list of ACL policies. Initially, there's an editable **default** policy and a **root** policy that does not contain any valid but can edit anything within OpenBao.

ACL policies page

Tools

Wrap

The **Wrap** page presents a tool that allows you to wrap your JSON data into a secret engine token.

Wrap page

Lookup

The **Lookup** page presents a tool that allows you to get details of a certain secret engine token. It provides the following information:

- **Creation path** — path where a token was created (e.g. auth/approle/login)
- **Creation time** — time when a token was created (e.g. **2025-09-26T13:00:47.000-04:00[America/New_York]**)
- **Creation TTL** — time-to-live (TTL) of a token in seconds (e.g. **2764800**)
- **Expire time** — date-time in a format with zone (e.g. **Wed Sep 26 2035 13:00:47 GMT-0400 (Eastern Daylight Time)**)
- **Explain** — approximate countdown to the token expiration time (e.g. **307 milliseconds**)

Lookup page

Unwrap

The **Unwrap** page presents a tool that allows you to unwrap your secret engine token into JSON data.

Unwrap page

Rewrap

The **Rewrap** page presents a tool that allows you to rewrap the data of your secret engine token into a new token.

Rewrap page

Random

The **Random** page presents a tool that generates a sequence of bytes of custom length in the **Bytes** or **Hex** formats.

Random page

Hash

The **Hash** page presents a tool that allows you to generate a cryptographic hash of any input data. It can generate the output in **hex/hex** or **Hex** while using one of the available algorithms: **sha2-224**, **sha2-256**, **sha2-384**, **sha2-512**, and **sha3-224**.

Hash page

Seal OpenBao

The **Seal OpenBao** page allows you to seal OpenBao. When OpenBao is sealed, it hastes its root data storage (about all your secrets) is encrypted and inaccessible. The server cannot respond to almost anything: authentication, operation, any cryptographic operations until it's unsealed again with the **UNSEAL** CLI command or via the API.

Seal page

Use OpenBao command line interface

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Overview

Workflow

Usage examples

- Write and read secrets
- Create a user

Overview

OpenBao provides two types of command line interface (CLI): OpenBao Web CLI which is available in the web UI and a local shell tool `bao` — a wrapper around the HTTP API which is identical to the HashiCorp Vault API. The web version is very limited in terms of the operations it can perform.

For a local shell tool, a template command looks like this:

```
$ bao <command> [<options>] [<path>] [<args>]
```

where:

- `<command>` – a command from the OpenBao command list.
- `<options>` – flags to specify additional parameters.
- `<path>` – a secret path that is required for certain operations.
- `<args>` – API arguments specific to the operation.

To see the help manual for a command, call it with the `-h` flag as follows:

```
$ bao <command> -h
```

OpenBao CLI tool can use environment variables to obtain certain information so that you don't have to specify it every time you execute a command. If there's an environment variable set for something and you specify the same parameter as a flag, the flag value will be used. You can see which variable OpenBao can read in the OpenBao documentation.

Workflow

1. As an optional first step, you need to set the `VAULT_ADDR` environment variable to a host where OpenBao is running. By default, OpenBao tries to access `https://127.0.0.1:8200`, which is not always correct. If you omit this step, you can add the `-address` parameter to a command call in order to specify the host.
2. To perform an operation on a server, you need to authenticate. The default authentication method is `token`, which OpenBao will prompt for if it's not provided initially. You can choose the authentication method using the `-method` flag. For example, authentication with a `userpass` method looks as follows:

```
$ bao login -method=userpass username=test_user
```

   Next, OpenBao will prompt you to enter the user's password. If successful, your token will be cached to be used in further operations, and you will receive the following response:

```
Success! You are now authenticated. The token information displayed below is
already stored in the token helper. You do NOT need to run "bao login" again.
Future OpenBao requests will automatically use this token.

Key                    Value
---                    -----
token                  <your_token>
token_accessor         o5mN3ud1fz1Zmb4EIamcVhve
token_duration         1h
token_renewable        true
token_policies         ["default"]
identity_policies      []
policies               ["default"]
token_meta_username    test_user
```

3. Perform an operation. For example, to write some data into the `cubbyhole` secret engine, you can use a command as follows:

```
$ bao write cubbyhole/git-credentials username="student01" password="password"
```

   If successful, you will receive the following response:

```
Success! Data written to: cubbyhole/git-credentials
```

Usage examples

Write and read secrets

1. Enable the `kv` secrets engine:

```
$ bao secrets enable kv
```

2. Write a secret into the `kv` engine with a `kv put` command:

```
$ bao kv put kv/ssh-creds username="student02" password="password"
```

3. List all the keys in the `kv` engine:

```
$ bao list kv
```

   The command yields the following result:

```
Keys
----
ssh-creds
```

4. Read the `ssh-creds` secret in the `kv` engine:

```
$ bao read kv/ssh-creds
```

   The command yields the following result:

```
Key         Value
---         -----
password    password
username    student02
```

Create a user

1. Create an ACL policy that controls what a user will be able to do:

```
$ sudo vi admin.hcl
```

   You can paste the following policy that has no limitations or come up with your own:

```
path "*" {
  capabilities = ["create", "read", "update", "delete", "list", "sudo"]
}
```

2. Apply the created policy:

```
$ bao policy write admin admin.hcl
```

3. Create a user:

```
$ bao write auth/userpass/users/<username> password=<password> policies=admin
```

   The command should yield the following output:

```
Success! Data written to: auth/userpass/users/<username>
```

OpenBao service management via ADCM

Sergei Tikhomirov

To Table of Contents

The ADCM UI provides actions to manage the OpenBao service and its components. For information on how to run service actions, refer to ADPS service actions.

The actions available for the OpenBao service are listed in the table below.

Action	Description
Check	Runs service-specific tests to check the health of the service and its components
Start	Starts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied on the service startup. Otherwise, the service starts without applying configurations from ADCM
Stop	Stops the service
Restart	Restarts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied during the service restart. Otherwise, the service restarts without applying configurations from ADCM
Remove	Removes the service from the cluster. This action should be used to remove already installed services. Whereas the control can be used to remove a non-mapped service (a service which components have not been distributed among cluster hosts)
Validate configuration	Checks ports for conflicts
Add/Remove components	Reconfigures the service topology. With it, you can change the amount of the OpenBao Server components. Running this action opens the host-component mapping interface where you can add new service components

Connect to Ranger via REST API

AD.PS

Sergey Talkanitsa

Contents

To Table of Contents

Overview

Create a policy

Update a policy

Get a policy

Delete a policy

Overview

Ranger provides a web interface – Ranger Admin UI, but another way to manage Ranger is via its REST API. This article provides a few examples of how to carry out HDFS policy-related operations using the API. To get a full list of endpoints and available methods, see the Ranger REST API documentation.

A simple way to create a query is to use curl and it's going to be demonstrated below, but you can construct requests in any convenient way and send them from any host within the cluster.

A template request looks like this:

```
$ curl -i -u <login>:<password> -H "Content-Type: application/json" -d '<json>' -X <request_type> <ranger_host>/service/<endpoint>
```

where:

- `<login>` – admin username.
- `<password>` – password for `<login>`.
- `<json>` – JSON string to be sent with the request. Omit if the request type is `DELETE`.
- `<request_type>` – type of the HTTP request (`GET` `POST` `PUT` `DELETE`). If the request type is `DELETE`, remove the `-H` parameter from the command.
- `<ranger_host>` – Ranger host URL (e.g. *http://rokhomov-wbp2.vn.verint1.internal:6080*).
- `<endpoint>` – resource to which the request will be sent.

The `-i` option allows to see the HTTP headers and check the status of the request, feel free to remove it if deemed unnecessary.

Create a policy

To create a resource-based policy, you need to construct one in JSON. An example of a policy for an HDFS service is presented below:

```
{
    "isEnabled": true,
    "service": "adh-2_hdfs_id_1",
    "name": "test_policy",
    "description": "Test policy for HDFS service",
    "isAuditEnabled": true,
    "resources":{
        "path":{
            "values":["/resource"],
            "isExcludes":false,
            "isRecursive":true
        }
    },
    "policyItems": [{
        "accesses": [
        {
            "type":"read",
            "isAllowed":true
        },
        {
            "type":"write",
            "isAllowed":true
        }],
        "users": ["test_user"],
        "groups": [],
        "conditions": [],
        "delegateAdmin": false
    }],
    "denyPolicyItems": [],
    "allowExceptions": [],
    "denyExceptions": [],
    "dataMaskPolicyItems": [],
    "rowFilterPolicyItems": []
}
```

1. Name of the resource-based service for which the policy is created.
2. Name of the policy.
3. Resource for which the policy is created. Cannot create multiple policies for one resource.
4. Access permissions.
5. Users to which the access permissions apply.

IMPORTANT

Make sure to turn JSON into a string. To do that, you can use a service like JSON Minifier.

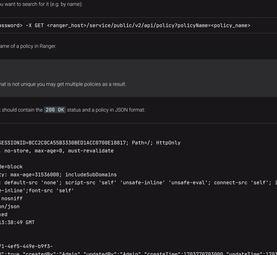
The corresponding curl command is:

```
$ curl -i -u <login>:<password> -H "Content-Type: application/json" -d '<json>' -X POST <ranger_host>/service/public/v2/api/policy
```

The expected output should contain the `200 OK` status and the newly created policy in JSON format:

```
HTTP/1.1 200 OK
Set-Cookie: RANGERADMINSESSIONID=AE33E1171A5ADF9950CC859F8908B2F; Path=/; HttpOnly
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
Strict-Transport-Security: max-age=31536000; includeSubDomains
Content-Security-Policy: default-src 'none'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; connect-src 'self'; img-src 'self';
style-src 'self' 'unsafe-inline';font-src 'self'
X-Content-Type-Options: nosniff
Content-Type: application/json
Transfer-Encoding: chunked
Date: Thu, 20 Dec 2023 13:38:23 GMT
Server: Apache Ranger

{"id":43,"guid":"2c2f8cf1-4ef5-449e-b9f3-
978f73d1ad0f","isEnabled":true,"createdBy":"Admin","updatedBy":"Admin","createTime":1703770703330,"updateTime":1703770703330,"ver
sion":1,"service":"adh-2_hdfs_id_1","name":"test_policy","policyType":0,"policyPriority":0,"description":"Test policy for HDFS
service","resourceSignature":"391f2d50f4f4737d077e9fe73af7bc7d918e0905025176820a2e8b5642575f2","isAuditEnabled":true,"resources
":{"path":{"values":["/resource"],"isExcludes":false,"isRecursive":true}},"policyItems":[{"accesses":
[{"type":"read","isAllowed":true},{"type":"write","isAllowed":true}],"users":["test_user"],"groups":[],"roles":[],"conditions":
[],"delegateAdmin":false}],"denyPolicyItems":[],"allowExceptions":[],"denyExceptions":[],"dataMaskPolicyItems":
[],"rowFilterPolicyItems":[],"serviceType":"hdfs","options":{},"validitySchedules":[],"policyLabels":
[],"zoneName":"","isDenyAllElse":false}
```

You can check that the policy was created in Ranger Admin UI:



Newly created policy

Update a policy

You may need to update a policy to add a user to the list of users with the same permissions to some resource. To do that, you need to use the `PUT` request and the following curl command:

```
$ curl -i -u <login>:<password> -H "Content-Type: application/json" -d '<json>' -X PUT
<ranger_host>/service/public/v2/api/policy/<id>
```

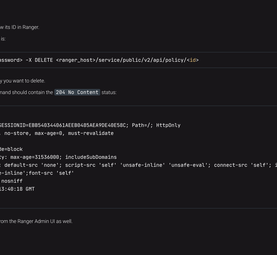
where:

- `<json>` is the updated policy.
- `<id>` is the ID of the policy you want to update.

The expected output should contain the `200 OK` status and the updated policy in JSON:

```
HTTP/1.1 200 OK
Set-Cookie: RANGERADMINSESSIONID=B5FA38544B01101230E81E32492B59E; Path=/; HttpOnly
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
Strict-Transport-Security: max-age=31536000; includeSubDomains
Content-Security-Policy: default-src 'none'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; connect-src 'self'; img-src 'self';
style-src 'self' 'unsafe-inline';font-src 'self'
X-Content-Type-Options: nosniff
Content-Type: application/json
Transfer-Encoding: chunked
Date: Thu, 20 Dec 2023 15:15:21 GMT
Server: Apache Ranger

{"id":43,"guid":"2c2f8cf1-4ef5-449e-b9f3-
978f73d1ad0f","isEnabled":true,"createdBy":"Admin","updatedBy":"Admin","createTime":1703770703000,"updateTime":1703774521670,"ver
sion":2,"service":"adh-2_hdfs_id_1","name":"test_policy","policyType":0,"policyPriority":0,"description":"Test policy for HDFS
service","resourceSignature":"391f2d50f4f4737d077e9fe73af7bc7d918e0905025176820a2e8b5642575f2","isAuditEnabled":true,"resources
":{"path":{"values":["/resource"],"isExcludes":false,"isRecursive":true}},"policyItems":[{"accesses":
[{"type":"read","isAllowed":true},{"type":"write","isAllowed":true}],"users":["test_user","new_test_user"],"groups":[],"roles":
[],"conditions":[],"delegateAdmin":false}],"denyPolicyItems":[],"allowExceptions":[],"denyExceptions":[],"dataMaskPolicyItems":
[],"rowFilterPolicyItems":[],"serviceType":"hdfs","options":{},"validitySchedules":[],"policyLabels":
[],"zoneName":"","isDenyAllElse":false}
```

In the Ranger Admin UI, you can see that a new user has appeared.



Added user to a policy

Get a policy

You can get a Ranger policy if you know its ID or you can search for it by its parameters.

The corresponding command if you know the ID:

```
$ curl -i -u <login>:<password> -X GET <ranger_host>/service/public/v2/api/policy/<id>
```

where `<id>` is the ID of a policy in Ranger.

The corresponding command if you want to search for it (e.g. by name):

```
$ curl -i -u <login>:<password> -X GET <ranger_host>/service/public/v2/api/policy?policyName=<policy_name>
```

where `<policy_name>` is the name of a policy in Ranger.

NOTE

If you search by a parameter that is not unique you may get multiple policies as a result.

In both cases, the expected output should contain the `200 OK` status and a policy in JSON format:

```
HTTP/1.1 200 OK
Set-Cookie: RANGERADMINSESSIONID=BCC2C8CA56B3330BED1AC0D70BE18017; Path=/; HttpOnly
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
Strict-Transport-Security: max-age=31536000; includeSubDomains
Content-Security-Policy: default-src 'none'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; connect-src 'self'; img-src 'self';
style-src 'self' 'unsafe-inline';font-src 'self'
X-Content-Type-Options: nosniff
Content-Type: application/json
Transfer-Encoding: chunked
Date: Thu, 20 Dec 2023 13:38:44 GMT
Server: Apache Ranger

{"id":43,"guid":"2c2f8cf1-4ef5-449e-b9f3-
978f73d1ad0f","isEnabled":true,"createdBy":"Admin","updatedBy":"Admin","createTime":1703770703000,"updateTime":1703770703000,"ver
sion":1,"service":"adh_hdfs_id_0","name":"test_policy","policyType":0,"policyPriority":0,"description":"Test policy for HDFS
service","resourceSignature":"391f2d50f4f4737d077e9fe73af7bc7d918e0905025176820a2e8b5642575f2","isAuditEnabled":true,"resources
":{"path":{"values":["/resource"],"isExcludes":false,"isRecursive":true}},"policyItems":[{"accesses":
[{"type":"read","isAllowed":true},{"type":"write","isAllowed":true}],"users":["test_user"],"groups":[],"roles":[],"conditions":
[],"delegateAdmin":false}],"denyPolicyItems":[],"allowExceptions":[],"denyExceptions":[],"dataMaskPolicyItems":
[],"rowFilterPolicyItems":[],"serviceType":"hdfs","options":{},"validitySchedules":[],"policyLabels":
[],"zoneName":"","isDenyAllElse":false}
```

Delete a policy

To delete a policy, you need to know its ID in Ranger.

The corresponding curl command is:

```
$ curl -i -u <login>:<password> -X DELETE <ranger_host>/service/public/v2/api/policy/<id>
```

where `<id>` is the ID of the policy you want to delete.

The expected output for this command should contain the `204 No Content` status:

```
HTTP/1.1 204 No Content
Set-Cookie: RANGERADMINSESSIONID=E8B54B3144061AEE8D485AE49DE40E58C; Path=/; HttpOnly
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
Strict-Transport-Security: max-age=31536000; includeSubDomains
Content-Security-Policy: default-src 'none'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; connect-src 'self'; img-src 'self';
style-src 'self' 'unsafe-inline';font-src 'self'
X-Content-Type-Options: nosniff
Date: Thu, 20 Dec 2023 13:40:18 GMT
Server: Apache Ranger
```

The deleted policy will disappear from the Ranger Admin UI as well.

Connect to Ranger via KMS REST API

Sergei Tikhomirov

Contents

To Table of Contents

Overview

Example

- Preparation
- Process

Overview

Ranger KMS REST API is identical to Hadoop KMS REST API. Hence, you need to have a valid Kerberos ticket on the host from which you are trying to access Ranger KMS. To get a full list of available endpoints, head over to Ranger KMS API documentation.

A simple way to create a query is to use curl and it's going to be demonstrated below, but you can construct requests in any convenient way and send them from any host within the cluster, provided you have a valid ticket. A template curl command looks like the one below:

```
$ curl -i --negotiate -u : -X <request_type> <ranger_kms_host>/kms/<endpoint>
```

where:

- `<request_type>` — type of the HTTP request (`GET`, `POST`, `DELETE`). If the request is supposed to carry some data, specify its content type using `-H` and attach the data using `-d`.
- `<ranger_kms_host>` — Ranger KMS host URL with port (e.g. *http://stikhomirov-adps.ru-central1.internal:9292*).
- `<endpoint>` — resource to which the request will be sent.

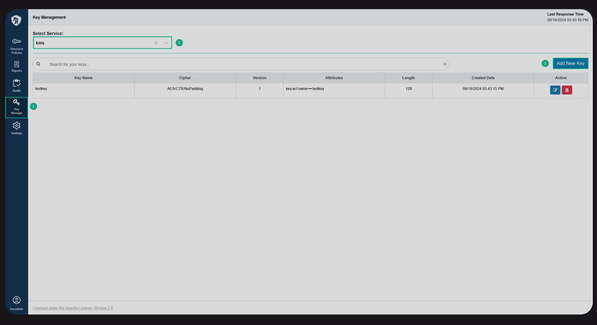
The `-i` option allows you to see the HTTP headers and check the status of the request, feel free to remove it if deemed unnecessary. The `-u` option is required but it's ignored and a ticket is used for authentication.

Example

This example demonstrates how to get key names using Ranger KMS REST API. Since, by default, there are no keys, you need to create one using the REST API or in Ranger Admin Web UI. The second option is demonstrated below.

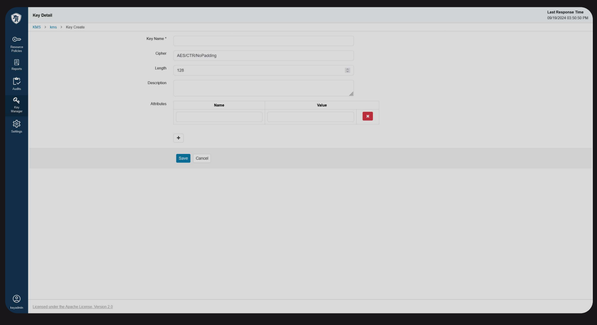
Preparation

1. Log into Ranger Admin Web UI as `keyadmin`, open the ***Key Manager*** page from a menu on the left, select the ***kms*** service option, and click ***Add New Key***.



The Encryption page

2. Fill in the necessary details and click ***Save*** at the bottom of the page. The newly created key will appear on the ***Key Manager*** page.



The Create Key page

Process

1. Log onto a host within a cluster that has a valid Kerberos ticket.
2. Make sure that your curl version has `GSS-Negotiate` among its features by running the command below:

```
$ curl -V
```

A possible output is:

```
curl 7.29.0 (x86_64-redhat-linux-gnu) libcurl/7.29.0 NSS/3.90 zlib/1.2.7 libidn/1.28 libssh2/1.8.0
Protocols: dict file ftp ftps gopher http https imap imaps ldap ldaps pop3 pop3s rtsp scp sftp smtp smtps telnet tftp
Features: AsynchDNS GSS-Negotiate IDN IPv6 Largefile NTLM NTLM_WB SSL libz unix-sockets
```

3. Make sure that you have a valid ticket by running `klist`.
4. Run the command below:

```
$ curl -i --negotiate -u : -X GET <ranger_kms_host>/kms/v1/keys/names
```

5. You can find an array of key names in the response body. The key name from the example is returned as follows:

```
[ "testkey" ]
```

Connect to Ranger via web interface

Sergei Tikhomirov

Contents

To Table of Contents

Overview

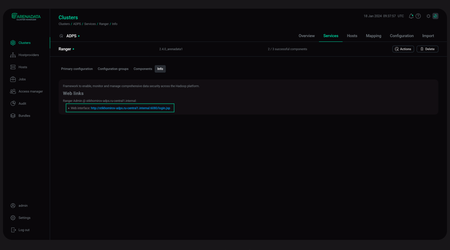
Header menu

Main page

Overview

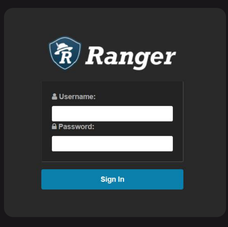
While you can manage Ranger via REST API, there's also a convenient web interface — Ranger Admin UI. You can connect to it directly on the 6080 port (or 6182 if SSL is enabled) of your Ranger host, or you can find a link in ADCM following the steps below:

1. On the ***Clusters*** page, find your ADPS cluster and click its name.
2. Open the ***Services*** tab and click ***Ranger***.
3. In the service menu, click ***Info*** and find the link leading to the web interface in the ***Web links*** pane.



Link to web interface

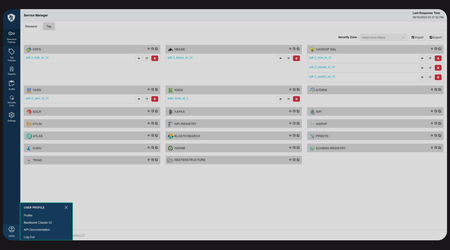
Once you are on a login page, fill in your credentials and click ***Sign In***.



Log in to Ranger

Header menu

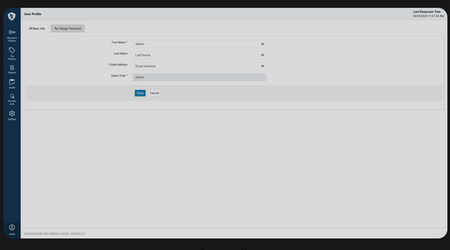
In the bottom left corner, there's a drop-down menu with several actions available.



Header menu

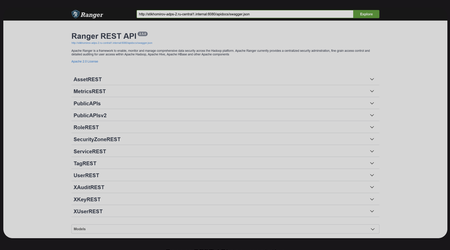
The available actions are:

- ***Profile***. This option takes you to the ***User Profile*** page on which you can edit your name, email address, and change password.



Ranger profile

- ***Backbone Classic UI***. This action changes Ranger UI from latest to classic and vice versa.
- ***API Documentation***. This option contains a link to the Ranger REST API Swagger documentation.

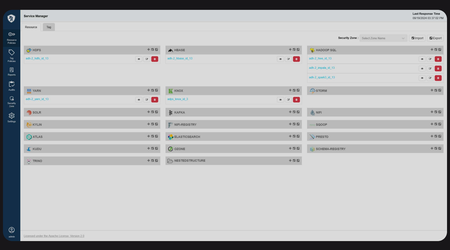


Ranger REST API documentation

- ***Log Out***. This action logs you out of Ranger and redirects you to the login page.

Main page

The default page is ***Service Manager***, which displays resource-based services.



Service Manager screen

The content of this page depends on the permissions the user has. Such permissions also control the tabs that are visible in the header. For example, by default only `keyadmin` will have the ***Key Manager*** tab, which allows them to manage keys, while `admin` has access to (apart from ***Service Manager***) Security zones and ***Settings***, where they can manage permissions.

Manage services in Ranger

Sergei Tikhomirov

Contents

To Table of Contents

Overview

Add a service

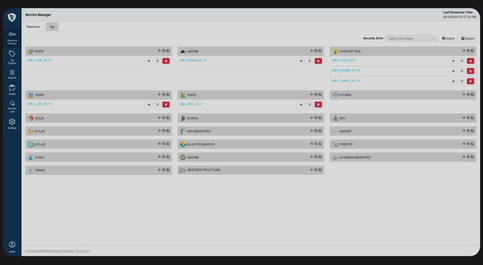
View service details

Edit a service

Remove a service

Overview

Ranger Admin UI provides a ***Service Manager*** screen on which you can view, add, edit, or remove a service. The screen appears once you open the ***Resource Policies*** or ***Tag Policies*** page. It is also the default screen you see after logging into Ranger Admin.

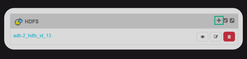


Service Manager screen

Add a service

To manually add a service in Ranger Admin UI, follow the steps below:

1. Click ➕ next to a type of service you want to add.



Adding a service

2. On the opened ***Create Service*** page, fill in the service details and config parameters.
3. Click ***Add*** at the bottom of the page. The newly added service will appear in the ***Service Manager*** pane.

Another way to add a service is by enabling its Ranger plugin in your ADH/ADS/ADPS cluster. Currently supported Ranger plugins for each product are presented below.

ADH

- HDFS
- HBase
- Hive
- Impala
- Ozone
- Solr
- Spark3
- Trino
- YARN

ADS

- Kafka
- NiFi

ADPS

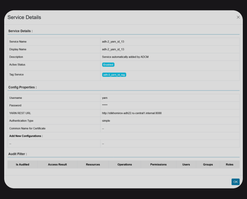
- Knox
- Solr

View service details

To view information about an existing service, click 👁 next to it.

Viewing service details

Service details, config properties, and audit filter information will appear in a pop-up window.

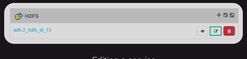


Service details

Edit a service

To edit an existing service, follow the steps below:

1. Click ✎ next to a service.



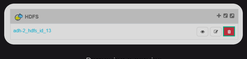
Editing a service

2. On the opened ***Edit Service*** page, edit the necessary information.
3. Click ***Save*** at the bottom of the page.

Remove a service

To remove an existing service from Ranger, follow the steps below:

1. Click 🗑 next to a service.



Removing a service

2. Confirm the action by clicking ***OK*** in the pop-up window. The removed service will disappear from the ***Service Manager*** screen.

Manage policies in Ranger

Version: 1.0.0 (Current)

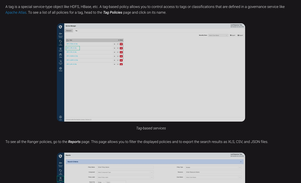
Contents

To Table of Contents

Overview
Add a policy
View policy details
Edit a policy
Remove a policy
Export policies
Import a policy
Label a policy

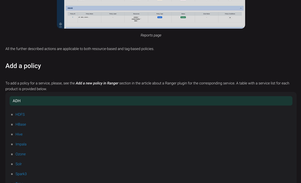
Overview

There are two types of policies in Ranger – the **resource** based and the **tag** based ones. Resources are parts of services which you can manage access for (e.g. Hive column, Solr collection, etc.). A resource-based policy allows you to control access to resources for , and . To see a list of all available services for which you can view policies, go to the **Resource Policies** page. It represents a screen with services sorted into panes with corresponding names. To view a list of all resource-based policies for a service, click on its name.



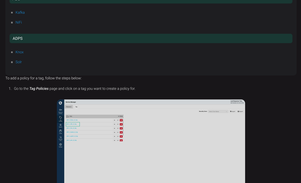
Resource-based services

A tag is a special service type object like HDFS, HBase, etc. A tag-based policy allows you to control access to tags or classifications that are defined in a governance service like Apache Atlas. To see a list of all policies for a tag, head to the **Tag Policies** page and click on its name.



Tag-based services

To see all the Ranger policies, go to the **Reports** page. This page allows you to filter the displayed policies and to export the search results as XLS, CSV, and JSON files.



Reports page

All the further described actions are applicable to both resource-based and tag-based policies.

Add a policy

To add a policy for a service, please, see the **Add a new policy in Ranger** section in the article about a Ranger plugin for the corresponding service. A table with a service list for each product is provided below.

ADH

- HDFS
- HBase
- Hive
- Impala
- Kudu
- Solr
- Sqoop2
- Yarn
- YARN

ADS

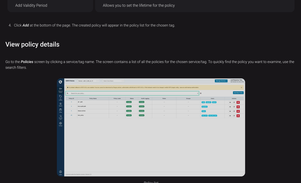
- Kafka
- NiFi

ADPS

- Knox
- Solr

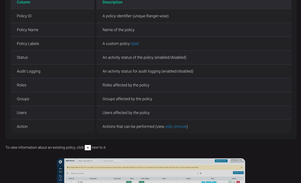
To add a policy for a tag, follow the steps below:

1. Go to the **Tag Policies** page and click on a tag you want to create a policy for.



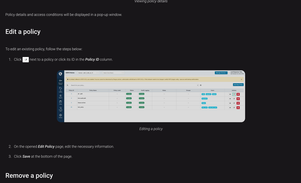
Tag-based services

2. Click **Add New Policy**.



List of policies for a tag

3. On the opened **Create Policy** page, fill in the necessary details and set up access conditions.



Tag policy details

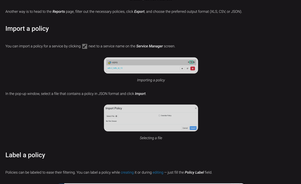
Policy details parameters

Parameter	Description
Policy Name	The policy name. Must be unique across the system
Enabled	Indicates whether to enable the policy after creation
Normal/Override	Allows you to specify an override policy. When **override** style is selected, the access permissions of the new policy override the access permissions in existing policies
Policy Label	Allows grouping of sets of policies with one or more labels and searching for policies by label names. You can use search on the **Tag Policies**, **Resource Policies**, and **Reports** pages. Also helps to export/import policies. For example, to export/import specific set of policies, then they can search for a policy label and export the specific set of policies
TAG	A tag that data should have for this policy to be applied to it
Description	Describes the purpose of the policy
Audit Logging	Enables audit for the policy
Add Validity Period	Allows you to set the lifetime for the policy

4. Click **Add** at the bottom of the page. The created policy will appear in the policy list for the chosen tag.

View policy details

Go to the **Policies** screen by clicking a service tag name. The screen contains a list of all the policies for the chosen service/tag. To quickly find the policy you want to examine, use the search filters.



Policy list

Policies screen columns

Column	Description
Policy ID	A policy identifier (unique Ranger-wise)
Policy Name	Name of the policy
Policy Labels	A custom policy label
Status	An activity status of the policy (enabled/disabled)
Audit Logging	An activity status for audit logging (enabled/disabled)
Roles	Roles affected by the policy
Groups	Groups affected by the policy
Users	Users affected by the policy
Action	Actions that can be performed (view, edit, remove)

To view information about an existing policy, click next to it.

Viewing policy details

Policy details and access conditions will be displayed in a pop-up window.

Edit a policy

To edit an existing policy, follow the steps below:

1. Click next to a policy or click its ID in the **Policy ID** column.

Editing a policy

2. On the opened **Edit Policy** page, edit the necessary information.
3. Click **Save** in the bottom of the page.

Remove a policy

To remove an existing policy, follow the steps below:

1. Click next to a policy.

Removing a policy

2. Confirm the action by clicking **OK** in the pop-up window. The removed policy will disappear from the **Policies** screen.

Export policies

You can export all policies for a service/tag by clicking next to a service/tag name on the **Service Manager** screen.

Exporting policies

In the pop-up window, select the services/tags which you want to export policies for and click **Export**. Policies will be exported as a single JSON file.

Choosing service instances

Another way is to head to the **Reports** page, filter out the necessary policies, click **Export**, and choose the preferred output format (XLS, CSV, or JSON).

Import a policy

You can import a policy for a service by clicking next to a service name on the **Service Manager** screen.

Importing a policy

In the pop-up window, select a file that contains a policy in JSON format and click **Import**.

Selecting a file

Label a policy

Policies can be labeled to ease their filtering. You can label a policy while creating it or during editing – just fill the **Policy Label** field.

Policy details

Ranger audit

Ranger Overview

ADPS

Contents

To Table of Contents

Overview
Access
Admin
Login Sessions
Plugins
Plugin Status
User Sync

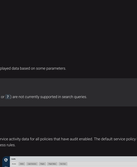
Overview

Ranger allows you to collect access audit history. It uses Solr to store audit logs and to implement a user interface for searching through them. The default configuration for Ranger audit in Solr uses a shared Solr instance provided by the ADPS bundle. Make sure that the Solr host has enough memory, CPU, and disk capacity.

NOTE

Solr must be installed and configured before installing Ranger Admin.

1. To authorize as Ranger admin, enter the administrative credentials and click **Sign In**.



Log in to Ranger

2. To manage audit policies in Ranger, open the Ranger console. Then, click **Audits** and choose a relevant option.

The **Audits** section includes the following tabs:

- Access
- Admin
- Login Sessions
- Plugins
- Plugin Status
- User Sync

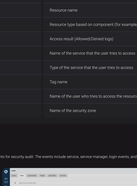
Each tab allows you to filter displayed data based on some parameters.

NOTE

Wildcards (for example, `X` or `2`) are not currently supported in search queries.

Access

The **Access** tab provides the service activity data for all policies that have audit enabled. The default service policy is configured to log all user activity. This default policy does not contain the user and group access rules.



Access tab

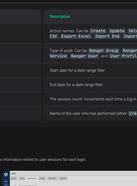
You can filter the data based on the criteria listed in the table below.

Access tab

Search criteria	Description
Access Enforcer	Access enforcer name
Access Type	Access type like `READ_EXECUTE`, `WRITE_READ_EXECUTE`
Agent Host Name	Hostname of the agent
Application	Application name
Audit ID	Audit identifier
Client IP	IP address that was used to access the resources
Cluster Name	Name of the cluster
Start Date	Start date for a date range filter
End Date	End date for a date range filter
Exclude User	Name of the user to exclude
Policy ID	Policy identifier
Resource Name	Resource name
Resource Type	Resource type based on component (for example, path in HDFS, database, table in Hive)
Result	Access result (Allowed/Denied logs)
Service Name	Name of the service that the user tries to access
Service Type	Type of the service that the user tries to access
Tags	Tag name
User	Name of the user who tries to access the resource
Zone Name	Name of the security zone

Admin

The **Admin** tab contains all events for security audit. The events include service, service manager, login events, and actions such as creating, updating, deleting, changing a password.



Admin tab

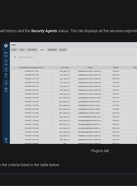
You can filter the data based on the criteria listed in the table below.

Admin tab

Search criteria	Description
Actions	Action names. Can be: `Create`, `Update`, `Delete`, `Password Change`, `Export JSON`, `Export CSV`, `Export Excel`, `Import End`, `Import Start`, `Import Create`, and `Import Delete`
Audit Type	Type of audit. Can be: `Ranger Group`, `Ranger Policy`, `Ranger Security Zone`, `Ranger Service`, `Ranger User`, and `User Profile`
Start Date	Start date for a date range filter
End Date	End date for a date range filter
Session ID	The session count. Increments each time a log in to the system is attempted
User	Name of the user who has performed either `Create`, `Update`, or `Delete` operation

Login Sessions

The **Login Sessions** tab displays information related to user sessions for each login.



Login Sessions tab

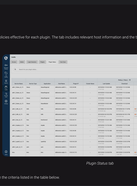
You can filter the data based on the criteria listed in the table below.

Login sessions tab

Search criteria	Description
IP	The IP address that the user utilized for login
Login ID	Name of the user who tried to access to the system
Login Type	The mode through which the user tries to login (by entering username and password)
Result	The result of the login attempt. Possible results are: `Success`, `Wrong Password`, `Account Disabled`, `Locked`, `Password Expired`, and `User Not Found`
Session ID	The session count. Increments every time a log in to the system is attempted
Start Date	Start date for a date range filter
End Date	End date for a date range filter
User Agent	The browser or library version used to login for the specific event (e.g. Mozilla, Java, Python)

Plugins

The **Plugins** tab shows the upload history and the **Security Agents** status. This tab displays all the services exported from the system.



Plugins tab

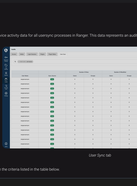
You can filter the data based on the criteria listed in the table below.

Plugins tab

Search criteria	Description
Cluster Name	Name of a cluster
HTTP Response Code	HTTP code returned when trying to export the service
Plugin ID	Name of the agent that tried to export the service
Plugin IP	IP address of the agent that tried to export the service
Service Name	The name of the service that was exported
Start Date	Start date for a date range filter
End Date	End date for a date range filter

Plugin Status

The **Plugin Status** tab shows policies effective for each plugin. The tab includes relevant host information and the time when the plugin was downloaded and started enforcing the policies.



Plugin Status tab

You can filter the data based on the criteria listed in the table below.

Plugin Status tab

Search criteria	Description
Application	Application name
Cluster Name	Name of a cluster
Host Name	Name of a host; can be FQDN
Plugin IP	IP Address of the agent that uses the plugin
Service Name	Name of the service that contains the policies
Service Type	Type of a service

User Sync

The **User Sync** tab provides service activity data for all usersync processes in Ranger. This data represents an audit trail for users and groups synchronized with each run of usersync.

User Sync tab

You can filter the data based on the criteria listed in the table below.

User Sync tab

Search criteria	Description
Sync Source	Source type (e.g. file, LDAP/AD, or Unix)
User Name	Name of the user who tried to access the resource
Start Date	Start date for a date range filter
End Date	End date for a date range filter

Manage security zones in Ranger

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Overview

Create a security zone

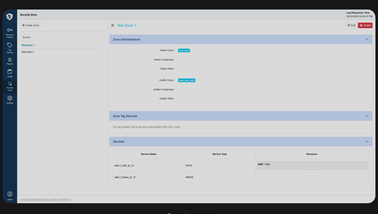
Edit a security zone

Delete a security zone

Overview

Ranger's **security zones** allow you to separate resource policies into various zones for convenience. Such separation simplifies administration of policies and also lowers the amount of policies that need to be checked during authorization, since only policies under a particular zone that contains the requested resource are loaded and checked. Also, it allows administrators to set up different policies based on the zones they have admin rights for.

You can see all the zones on the ***Security Zone*** page. To see information about a particular zone, click on its name in the list.

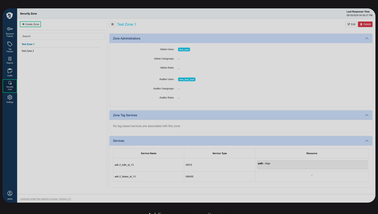


Security zones

Create a security zone

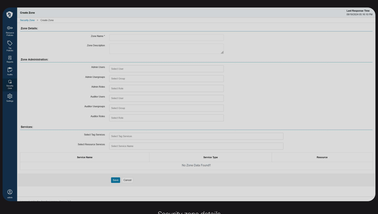
To create a security zone, follow the steps below:

1. On the ***Security Zone*** page, click + ***Create Zone*** above the list of the existing security zones.



Adding a new security zone

2. On the opened ***Create Zone*** page, fill in the necessary details and choose services that will be assigned to the zone. Each service has its own resources which can only be assigned to one security zone.



Security zone details

Zone parameters

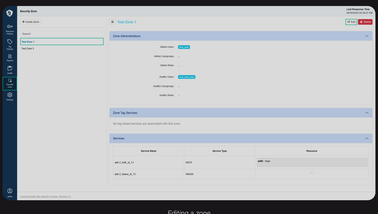
Parameter	Description
Zone Name	Name of the zone
Zone Description	Description of the zone's purpose
Admin Users	Users that will have admin rights within the zone
Admin Usergroups	Groups that will have admin rights within the zone
Auditor Users	Users that will have auditor rights withing the zone
Auditor Usergroups	Groups that will have auditor rights within the zone
Select Tag Services	Tag-based services to be added to the zone
Select Resource Services	Resource-based services to be added to the zone.

3. Click ***Save*** at the bottom of the page. The newly created zone will be added to the bottom of the zone list.

Edit a security zone

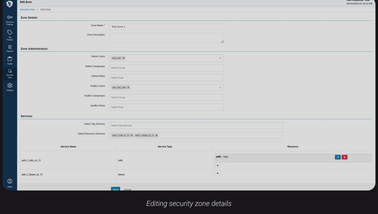
To edit a security zone, follow the steps below:

1. On the ***Security Zone*** page, select a security zone you want to edit by clicking on its name and click ✎ ***Edit***.



Editing a zone

2. On the opened ***Zone Edit*** page, edit the necessary information.



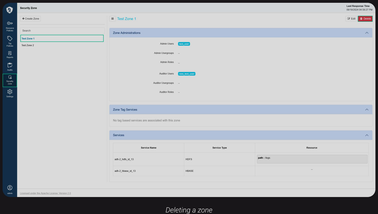
Editing security zone details

3. Click ***Save*** at the bottom of the page.

Delete a security zone

To delete a security zone, follow the steps below:

1. On the ***Security Zone*** page, select a security zone you want to delete by clicking on its name and click 🗑 ***Delete***.



Deleting a zone

2. Confirm the action by clicking ***OK*** in the pop-up window.

Manage users in Ranger

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Overview

Add a user

Edit a user profile

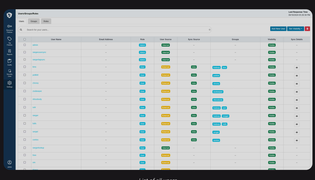
Remove a user

Change user visibility

Overview

A user is a basic entity in Ranger that can be a part of a group or a role. In Ranger, users can be **internal** and **external**. Internal users are created by a Ranger admin, external ones are imported from other systems like Active Directory, LDAP, or Unix.

To view the list of all users in Ranger Admin, head to the ***Settings*** → ***Users*** page.



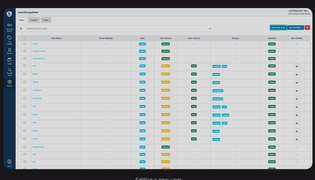
List of all users

Add a user

> **NOTE**
>
> Only users with the ***Admin*** role can create new users.

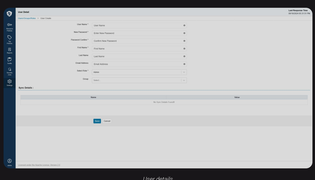
To add a new user, follow the steps below:

1. On the ***Users*** tab, click ***Add New User***.



Adding a new user

2. On the opened ***User Create*** page, fill in the necessary details.



User details

User Detail fields

Field	Description
User Name	Username that will be used to log into Ranger Admin. Must be unique across the system
New Password	Password for the user
Password Confirm	Password confirmation check (repeat the password)
First Name	First name of the user
Last Name	Last name of the user (optional)
Email Address	Email address of the user (optional)
Select Role	User role (Admin/User/Auditor)
Group	Groups that the user will be added to (optional)

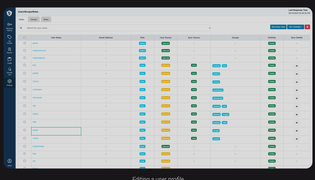
3. Click ***Save*** at the bottom of the page. The created user will appear at the end of the user list.

Edit a user profile

> **NOTE**
>
> Only profiles of the internal users can be edited fully, for external users you can only edit their role.

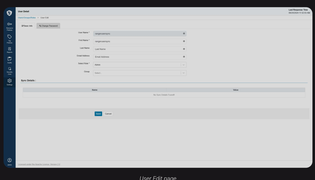
To edit information about a user, follow the steps below:

1. On the ***Users*** tab, click the username of the user whose profile you want to edit.



Editing a user profile

2. On the opened ***User Edit*** page, edit the information on the ***Basic Info*** tab or change the user's password on the ***Change Password*** tab.



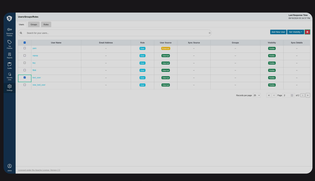
User Edit page

3. Click ***Save*** at the bottom of the page.

Remove a user

To Remove a user, follow the steps below:

1. On the ***Users*** tab, tick the checkbox next to a user (can choose multiple) you want to remove.
2. Click 🗑



Deleting a user

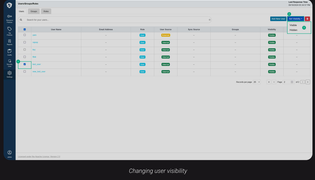
3. Confirm the action by clicking **OK** in the pop-up window.

Change user visibility

During LDAP/AD sync a lot of users are imported into the database. Most of the users might not be using Hadoop and associated services, so it's convenient to hide all the irrelevant user profiles so that they don't show up during policy creation.

To change user visibility, follow the steps below:

1. On the ***Users*** tab, tick the checkbox next to a user you want to hide or make visible (can choose multiple users).
2. Click ***Set Visibility*** and choose the relevant option (visible or hidden).



Changing user visibility

Manage groups in Ranger

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Overview

Create a group

Change group visibility

Edit group information

Delete a group

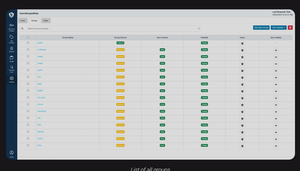
See the group members

Add a user to a group

Overview

A group in Ranger is a collection of users. It is convenient to create a group when you want to control access for several users at once. Groups can be **internal** and **external**. Internal groups are created by Ranger Admin, while the external ones are imported during a sync with other systems like LDAP, Active Directory, or Unix.

You can see all the groups in Ranger on the **Settings** → **Groups** page.

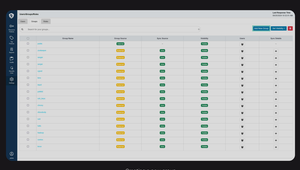


List of all groups

Create a group

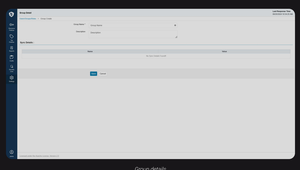
To create a new group, follow the steps below:

1. On the **Groups** tab, click ***Add New Group***.



Creating a new group

2. On the opened ***Group Create*** page, fill in the necessary information.



Group details

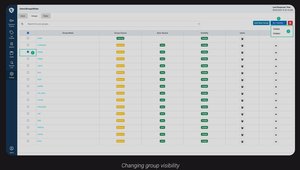
3. Click ***Save*** at the bottom of the page. The created group will be added to the bottom of the group list.

Change group visibility

During LDAP/AD sync a lot of groups are imported into the database. Some groups might not be using Hadoop and associated services, so it's convenient to hide all the irrelevant groups so that they don't show up during policy creation.

To change group visibility, follow the steps below:

1. On the **Groups** tab, tick the checkbox next to a group you want to hide or make visible (can choose multiple groups).
2. Click ***Set Visibility*** and choose the relevant option (visible or hidden).

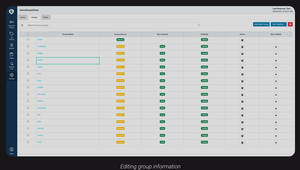


Changing group visibility

Edit group information

To edit information about a group, follow the steps below:

1. On the **Groups** tab, click the name of the group whose details you want to edit in the ***Group Name*** column.



Editing group information

2. On the opened ***Group Edit*** page, edit the necessary information.

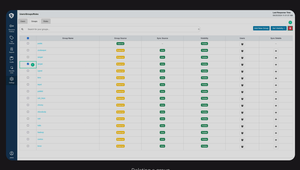
Group details

3. Click ***Save*** at the bottom of the page.

Delete a group

To delete a group, follow the steps below:

1. On the **Groups** tab, tick a checkbox next to the name of the group you want to delete.
2. Click 🗑



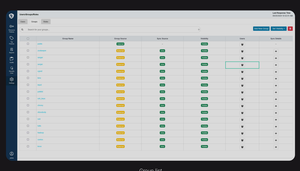
Deleting a group

3. Confirm the action by clicking **OK** in the pop-up window.

See the group members

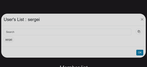
To see the members of a group, follow the steps below:

1. Open the **Groups** tab.
2. In the row of the group whose members you want to see, click the 👤 icon in the **Users** column.



Group list

3. The user list will appear in a pop-up window.



Member list

Add a user to a group

To manually add an internal user to a group, follow the steps below:

1. On the ***Settings*** → ***Users*** page, click on the username of the user you want to add to a group (essentially, initiate the editing process for a user).
2. In the **Group** field, select a group you want to add the user to (can choose multiple).
3. Click ***Save*** at the bottom of the page.

Manage roles in Ranger Admin UI

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Overview

Create a custom role

Edit a custom role

Delete a custom role

Role assignment

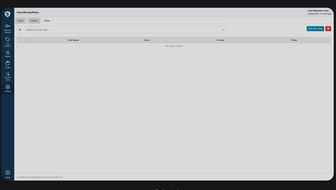
Overview

A role in Apache Ranger combines users, groups, and other roles. Role-based authorization is a well-established security practice that is also supported in Ranger.

Each user in Ranger Admin UI has one of the three internal system roles:

- `User` – a role with basic permissions.
- `Admin` – a role that has all the privileges for Ranger modules except for Key Manager. However, they can issue Key Manager permissions to themselves.
- `Auditor` – a role that has all the privileges for Ranger modules.

There are also custom roles which are created by admins. You can manage such roles that are used for Ranger service policies in Ranger Admin UI on the ***Settings*** → ***Roles*** tab. It allows you to set permissions for a role, a group, or a user within a policy. Internal permissions for Ranger modules (***Settings*** → ***Permissions***) are not managed by custom roles. Also, custom roles cannot be mapped to a user in ***User*** → ***Profile***.

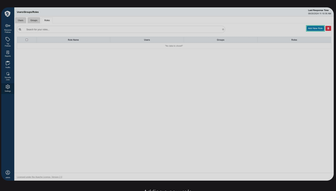


Roles tab

Create a custom role

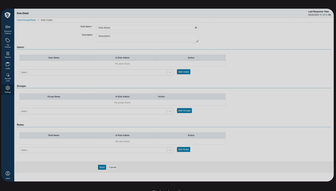
To create a custom role, follow the steps below:

1. On the ***Roles*** tab, click ***Add New Role***.



Adding a new role

2. On the opened ***Role Create*** page, fill in the necessary details.



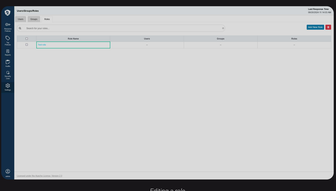
Role details

3. Click ***Save*** at the bottom of the page. The newly created role will appear on the ***Roles*** tab.

Edit a custom role

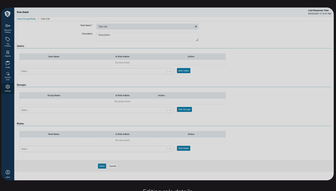
To edit a custom role, follow the steps below:

1. On the ***Roles*** tab, click the name of the role you want to edit in the ***Role Name*** column.



Editing a role

2. On the opened ***Role Edit*** page, edit the necessary information.



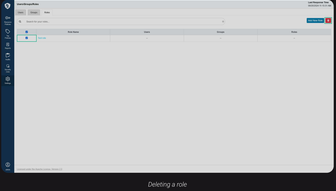
Editing role details

3. Click ***Save*** at the bottom of the page.

Delete a custom role

To delete a custom role, follow the steps below:

1. On the ***Roles*** tab, tick a checkbox next to the name of the role you want to delete.
2. Click 🗑.



Deleting a role

3. Confirm the action by clicking ***OK*** in the pop-up window.

Role assignment

When LDAP users are imported into Ranger, they are assigned an internal role, by default it is `User`. You can change the role for users or groups by applying the mapping filter. To do that, customize your *ranger-ugsync-site.xml* file contents:

```
ranger.usersync.group.based.role.assignment.rules: ROLE_SYS_ADMIN:g:group_name1, group_name2
ranger.usersync.group.based.role.assignment.rules: ROLE_SYS_ADMIN:u:username1, username2
```

Roles mapping

External role	Internal role
ROLE_USER	User
ROLE_SYS_ADMIN	Admin
ROLE_ADMIN_AUDITOR	Auditor

The `ROLE_KEY_ADMIN` and `ROLE_KEY_ADMIN_AUDITOR` Ranger roles are not available for mapping and cannot be used in filter. The users or groups mapped to unavailable roles will not be added to Ranger Admin UI.

Important notes

Make sure to restart Ranger to apply the newly modified filter. To remove a filter and restore default settings — remove imported users and restart Ranger.

The ***LDAP sync source for Ranger Admin authentication*** option should be enabled in the ADCM service configuration for Ranger.

Also, see that the LDAP parameters have values like shown below:

```
ranger.usersync.ldap.deltasync: true
ranger.usersync.group.searchenabled: true
ranger.usersync.group.search.first.enabled: false
ranger.usersync.group.usermapsyncenabled: true
```

See Configuration parameters for more parameter details.

Manage permissions in Ranger

Sergei Tikhomirov

Contents

To Table of Contents

Overview

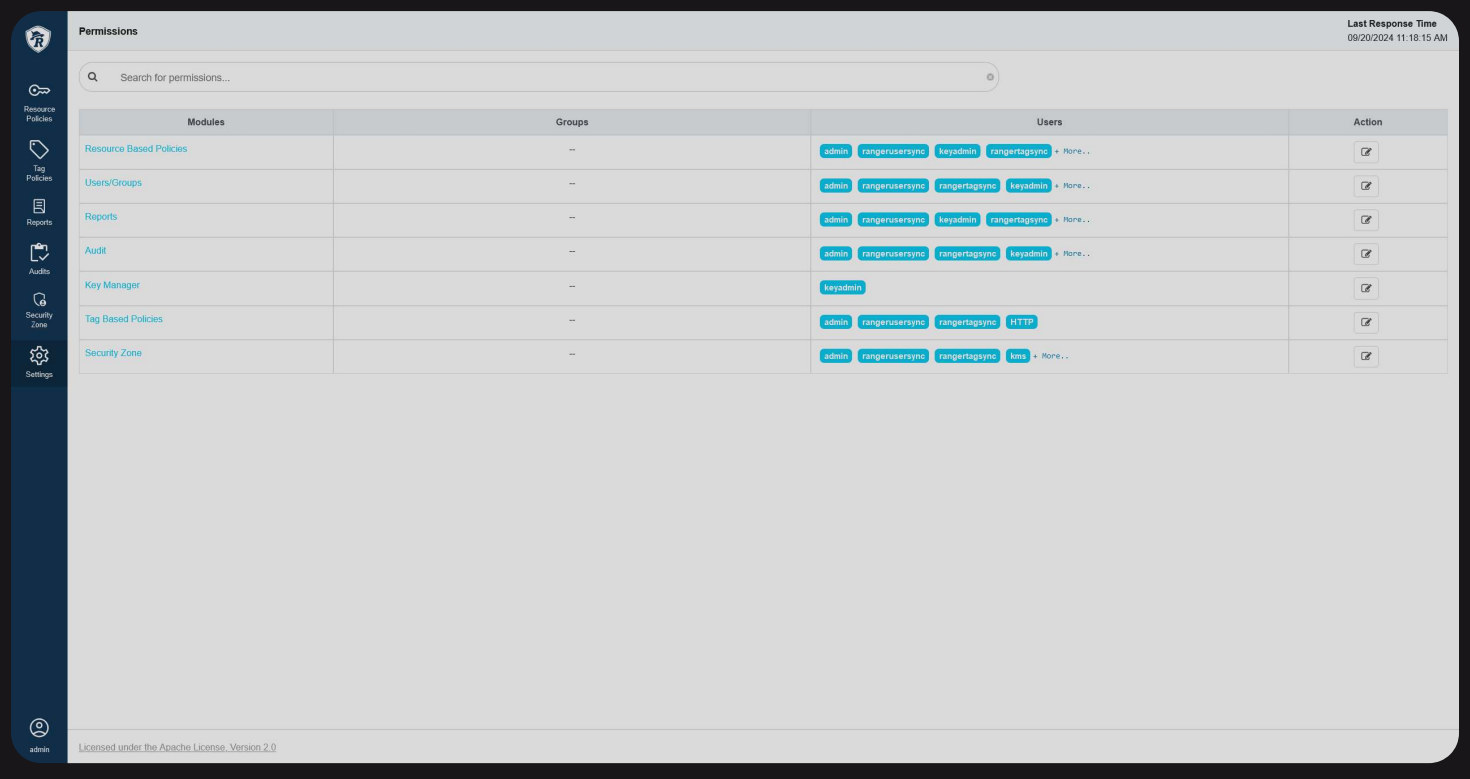
Grant permissions

Revoke permissions

Overview

Ranger module access control for users and groups is done via permissions. Access rights allow users/groups to see information about a module and use options within that module.

You can view all the permissions on the ***Settings*** → ***Permissions*** page.



Permissions screen

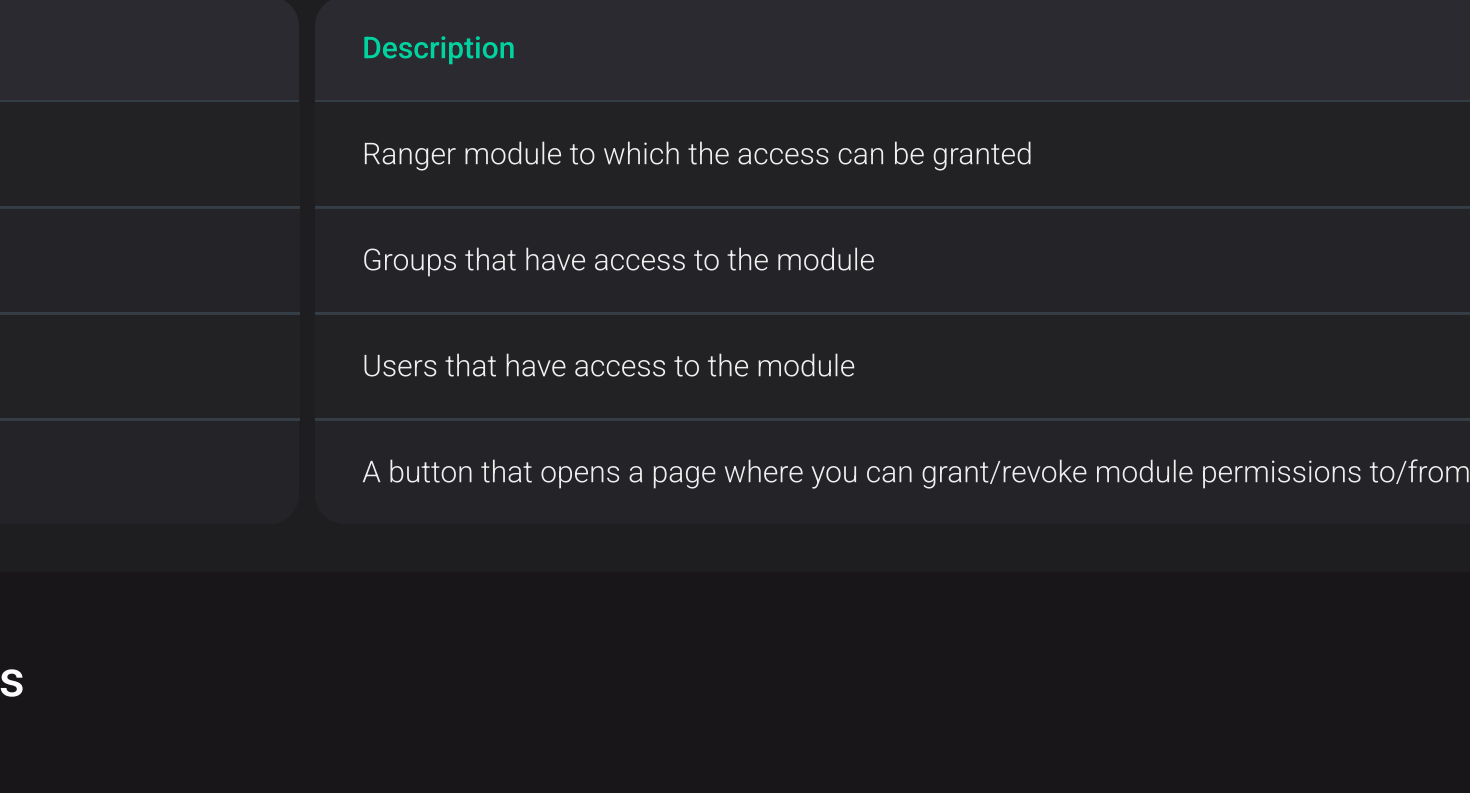
Permissions table columns

Column	Description
Modules	Ranger module to which the access can be granted
Groups	Groups that have access to the module
Users	Users that have access to the module
Action	A button that opens a page where you can grant/revoke module permissions to/from users and groups

Grant permissions

To grant module permissions to a user/group, follow the steps below:

1. On the ***Settings*** → ***Permissions*** page, click on the ✎ icon in the ***Action*** column in the row of the relevant module.
2. Find the user/group in the search bar in the corresponding section and click ➕.



Granting permissions

3. Click ***Save*** at the bottom of the page.

Revoke permissions

To revoke module permissions from a user/group, follow the steps below:

1. On the ***Settings*** → ***Permissions*** page, click the ✎ icon in the ***Action*** column in the row of the relevant module.
2. Click on the ✖ icon on a user/group you want to revoke the rights from.
3. Click ***Save*** at the bottom of the page.

Configure Ranger User Sync

Sergei Tikhomirov

Contents

To Table of Contents

Overview

LDAP sync configuration

Overview

Ranger User Sync is an optional Ranger component that allows you to keep the list of imported users and groups in Ranger up to date. For example, if an imported user was deleted in the source system, with a configured User Sync, it will become hidden in Ranger. It synchronizes users when Ranger is imported into a cluster from a preinstalled ADPS cluster (see the ***Overview*** section in Ranger plugins for ADH). After the import, Ranger will synchronize users periodically, by default — once an hour.

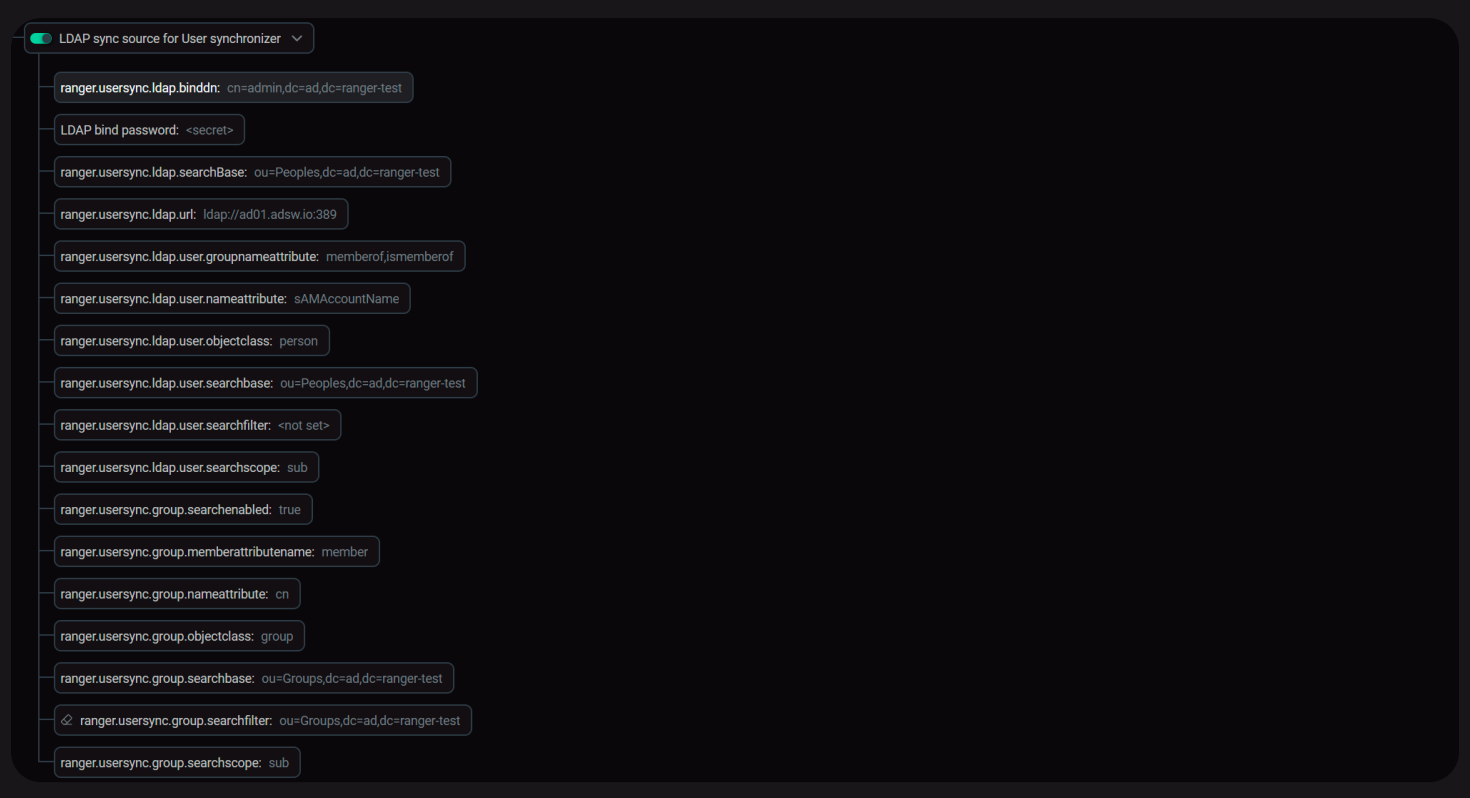
NOTE

The default search mode looks for groups first and adds users based on their group membership. To change it so that the users are looked up first and groups are added based on the users found, activate the ***Show advanced*** switch in the configuration menu and set the `ranger.usersync.group.search.first.enabled` parameter to `false`.

LDAP sync configuration

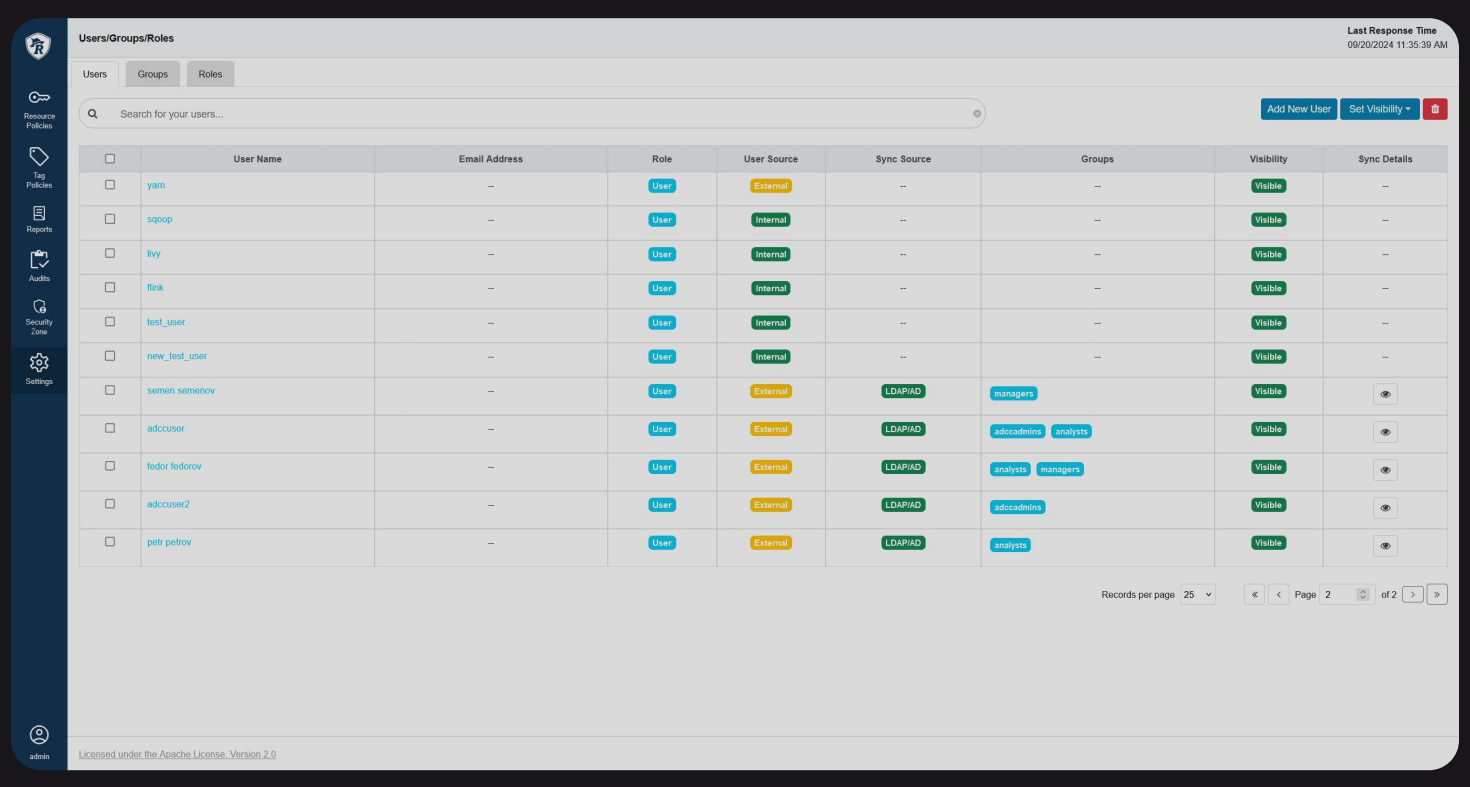
Additionally to the default sync method via importing Ranger into a cluster, you can add an LDAP sync source. To do that, follow the steps below:

1. On the ***Clusters*** page, click the name of your ADPS cluster, head to the ***Services*** tab, and click ***Ranger***.
2. In the configuration parameters, find ***LDAP sync source for User synchronizer*** and activate it. After that, fill in the LDAP parameters (see Configuration parameters → ***Ranger*** → ***LDAP sync source for User synchronizer***) and click ***Save***.



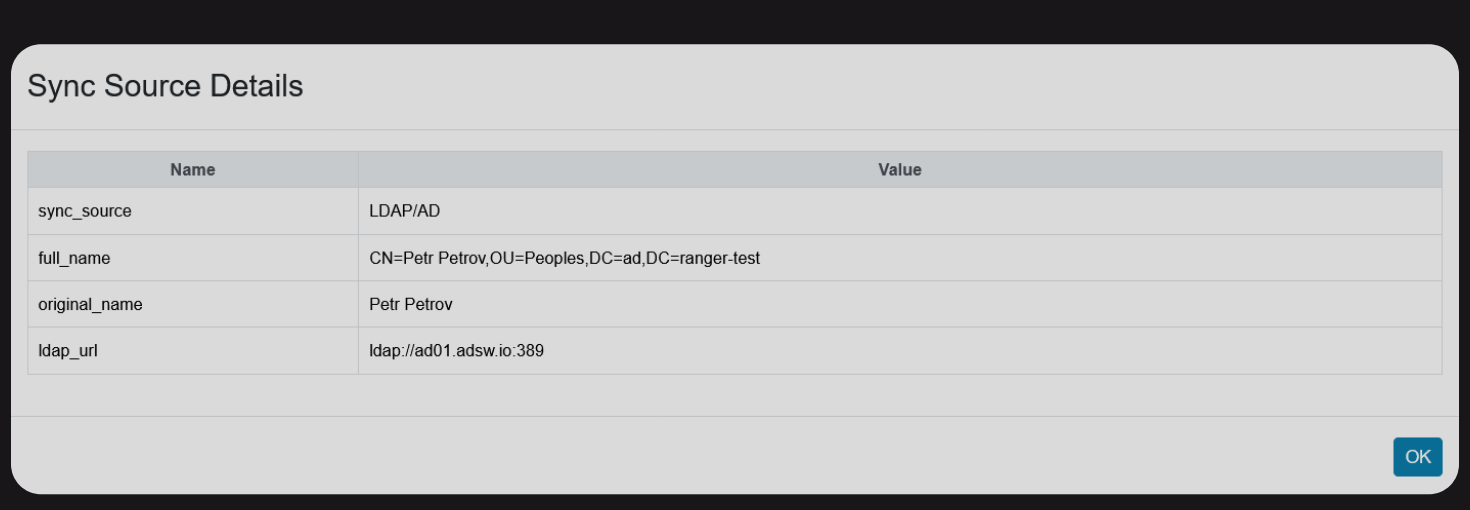
Example of the right LDAP parameters

3. Restart Ranger. The synchronized users/groups will appear on the relevant tabs with their source being `LDAP/AD`.



LDAP synchronized users

You can find special data on users/groups syncronized from external systems in the ***Sync Details*** column of the ***Users***/***Groups*** table. For example, users from Unix have parameters like `sync_source`, `full_name`, and `original_name`. The LDAP users have the forementioned and `ldap_url`, also their `full_name` is a DN.



LDAP sync details

Now, you can manage policies for the imported entities.

Configure authentication with LDAP/Active Directory

Sergei Tikhomirov

Contents

To Table of Contents

Overview

LDAP authentication configuration

Active Directory authentication configuration

Overview

Ranger allows you to add an extra source for Ranger Admin authentication. It can be an LDAP or Active Directory one.

IMPORTANT

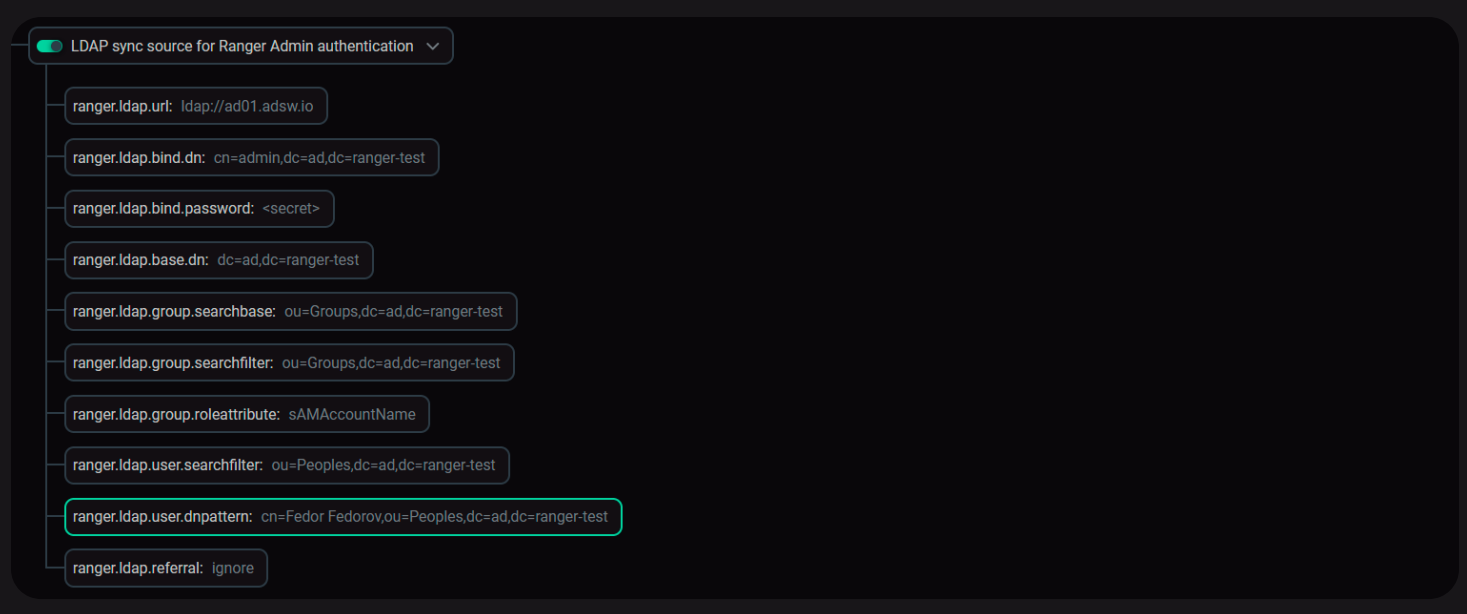
Only one extra type of authentication can be enabled at a time.

You can see the logs for login attempts in the *var/log/ranger/admin/<ranger_host>.log* file, where `<ranger_host>` is a host on which Ranger is installed.

LDAP authentication configuration

To configure an LDAP authentication source, follow the steps below:

1. On the ***Clusters*** page, click the name of your ADPS cluster, head to the ***Services*** tab, and click ***Ranger***.
2. In the configuration parameters, find ***LDAP sync source for Ranger Admin authentication*** and activate it. After that, fill in the LDAP parameters (see Configuration parameters → ***Ranger*** → ***LDAP sync source for Ranger Admin authentication***) and click ***Save***.



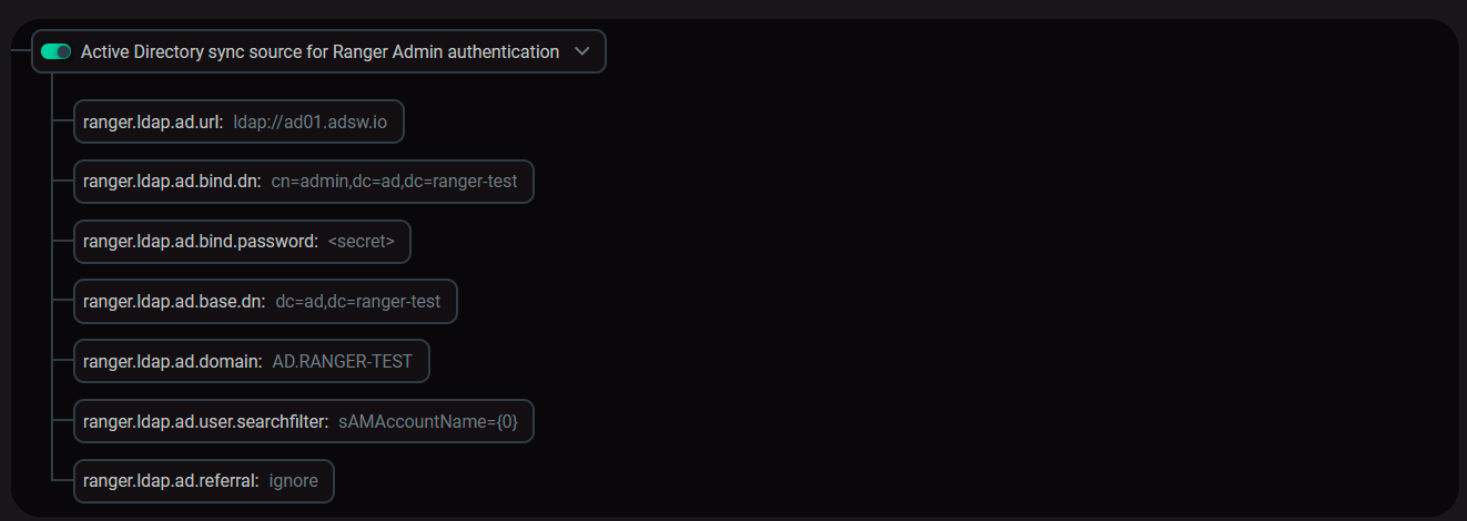
LDAP parameters

3. Restart Ranger (on your ADPS cluster's ***Services*** page, launch the ***Restart*** action for Ranger).

Active Directory authentication configuration

To configure an Active Directory authentication source, follow the steps below:

1. On the ***Clusters*** page, click the name of your ADPS cluster, head to the ***Services*** tab, and click ***Ranger***.
2. In the configuration parameters, find ***Active Directory sync source for Ranger Admin authentication*** and activate it. After that, fill in the AD parameters (see Configuration parameters → ***Ranger*** → ***Active Directory sync source for Ranger Admin authentication***) and click ***Save***.



Active Directory parameters

3. Restart Ranger: on your ADPS cluster's ***Services*** page, launch the ***Restart*** action for Ranger.

Configure Ranger Admin HA

Sergei Tikhomirov

Contents

To Table of Contents

Nginx configuration

Configuration in ADCM

Check

High availability (HA) allows the Ranger Admin service to deploy a backup in case the main host goes down. In order to support HA for Ranger Admin, you can add an Nginx load balancer.

Nginx configuration

IMPORTANT

Nginx should be installed on a cluster host that doesn't have the Ranger User synchronizer component on it.

On a dedicated Nginx host, do the following:

1. Install Extra Packages for Enterprise Linux.

```
$ sudo yum install epel-release -y
```

2. Install Nginx.

```
$ sudo yum install nginx -y
```

3. Add the following lines to the empty */etc/nginx/conf.d/ranger.conf* file.

```
upstream ranger {
  server <ranger_admin_1>:6080;
  server <ranger_admin_2>:6080 backup;
}
server {
  listen <port>;
  location / {
    proxy_pass http://ranger;
  }
}
```

where:

- `<ranger_admin_1>` and `<ranger_admin_2>` are the hosts with the Ranger Admin component installed.
- `<port>` is a free port, on which Nginx should listen.

4. Launch Nginx.

```
$ sudo systemctl enable nginx --now
```

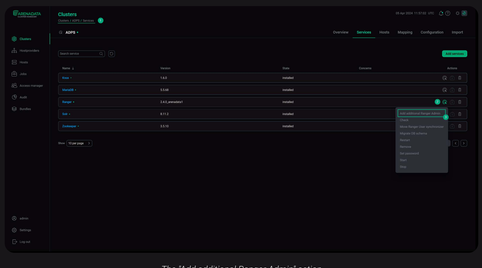
If everything is correct, the output should be similar to:

```
Created symlink from /etc/systemd/system/multi-user.target.wants/nginx.service to /usr/lib/systemd/system/nginx.service.
```

Configuration in ADCM

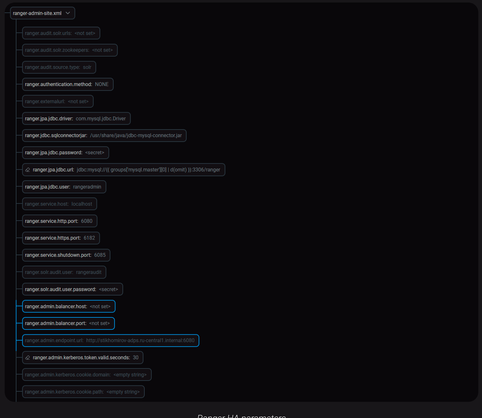
To configure HA for Ranger Admin, follow the steps below:

1. Add an extra Ranger Admin component if you haven't already. The action will open the host-component mapping screen, on which you can add another host. This step should be completed before kerberization.



The "Add additional Ranger Admin" action

2. In the Ranger service configuration, switch on the ***Show advanced*** flag, open the ***ranger-admin-site.xml*** parameter group, and fill the `ranger.admin.balancer.host` (Nginx host URL) and `ranger.admin.balancer.port` (port, on which Nginx listens) parameters.



Ranger HA parameters

3. Restart Ranger.

Check

To check if HA works, follow the steps below:

1. Stop the Ranger Admin service on the primary host.

```
$ sudo systemctl stop ranger-admin.service
```

2. See if the Ranger Admin service is available on the Nginx host on the `<port>` port.
3. Launch the service that was shut down.

```
$ sudo systemctl start ranger-admin.service
```

4. Complete the previous steps for each host with Ranger Admin installed.
5. Separately check if the service is available on the Ranger Admin hosts.

Configure Ranger KMS HA

Sergei Tikhomirov

Contents

To Table of Contents

Configuration in ADCM

Check

High availability (HA) allows the Ranger KMS service to deploy a backup in case the main host goes down.

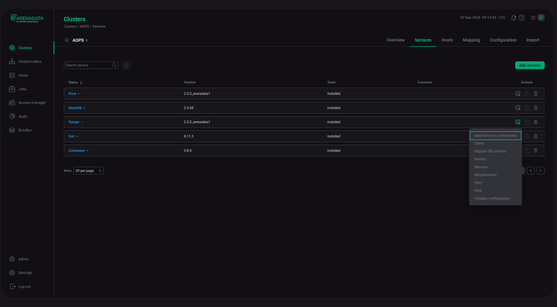
IMPORTANT

The HA mode is enabled automatically when two or more Ranger KMS instances are detected.

Configuration in ADCM

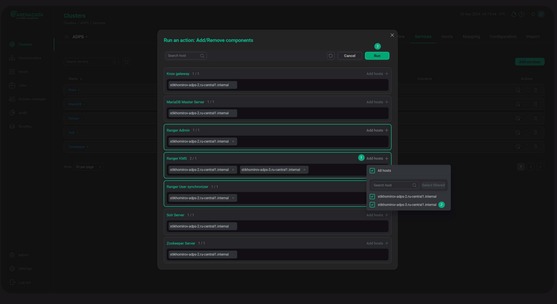
To add an extra Ranger KMS component, follow the steps below:

1. In the cluster configuration menu, open the ***Services*** tab and run the ***Add Ranger KMS*** action for Ranger. Alternatively, in the Ranger service menu, you can go to the ***Components*** tab and run the ***Add*** action for Ranger KMS that leads to the same result.



The Add Ranger KMS action

2. In the opened window, add an extra host for the Ranger KMS component and click ***Run***.



The host-component mapping window

3. Confirm the action in the pop-up window and wait for it to finish.

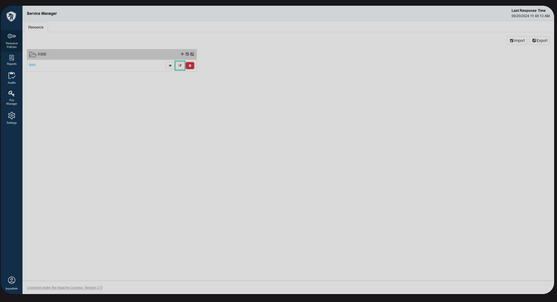
NOTE

For a kerberized cluster, SASL is used for authentication with the Kerberos user credentials. The name resolution rules are defined by the `hadoop.kms.authentication.kerberos.name.rules` parameter in the ***kms-site.xml*** parameter group available with the ***Show advanced*** flag enabled. By default, these rules concern only the service users (e.g. `ranger`, `hive`, `yarn`, etc.).

Check

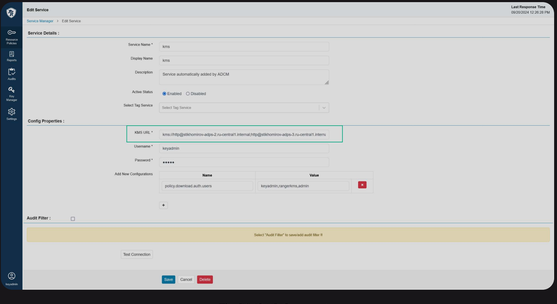
To check if the HA mode is enabled, follow the steps below:

1. Log into Ranger Admin UI as `keyadmin`.
2. Enter the edit page for the Ranger KMS service.



Edit the Ranger KMS service

3. If the HA mode is enabled, the ***KMS URL*** parameter should have several KMS host URLs (`kms://http@<kms_host_1>;<kms_host_2>:9292/kms`). Otherwise — there should be only one URL (`dbks://http@<kms_host>:9292/kms`). If SSL is enabled, `http` will be changed to `https` and the port `9292` will be changed to `9393`.



The "KMS URL" parameter

Configure an external PostgreSQL database for Ranger metadata

Sergei Tikhomirov

Contents

To Table of Contents

Overview
PostgreSQL server configuration
Host setup
Migration

Overview

The Ranger's ***Migrate DB schema*** action allows you to migrate a database schema and service users from an internal to an external database. The migration is available for the Ranger KMS and Ranger Admin service components.

Follow the steps below to configure the database server and cluster hosts for the migration.

> **NOTE**
>
> Before migrating to an external database, make sure that it's set up and running.

PostgreSQL server configuration

The example below demonstrates how to set up a PostgreSQL server on a CentOS 7 host:

1. To prevent a faulty dependency resolution, edit the */etc/yum.repos.d/CentOS-Base.repo* file by adding the following line in the `[base]` and `[updates]` sections:

```
exclude=postgresql*
```

2. Install the PostgreSQL configuration package.

```
$ sudo yum install https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-x86_64/pgdg-redhat-repo-latest.noarch.rpm
```

3. Install the necessary version of the PostgreSQL server.

```
$ sudo yum install postgresql<N>-server
```

   where `<N>` is a supported version of the server (12 is used from here on out).

4. Create a new database cluster.

```
$ sudo /usr/pgsql-12/bin/postgresql-12-setup initdb
```

5. Start the PostgreSQL service using `systemctl`.

```
$ sudo systemctl enable postgresql-12 --now
```

6. See if you can access the psql console.

```
$ sudo -i -u postgres; psql
```

From this point, refer to the relevant tab below. The commands should be launched in the psql console.

Ranger KMS

1. Create a user for the Ranger KMS service.

```
CREATE USER rangerkms WITH PASSWORD '<password>';
```

   where `<password>` is a password for the `rangerkms` user.

2. Create a database for the service.

```
CREATE DATABASE rangerkms;
```

3. Grant permissions to the database for the created user.

```
GRANT ALL PRIVILEGES ON DATABASE rangerkms TO rangerkms;
```

4. Add the following line to the */var/lib/pgsql/12/data/pg_hba.conf* file:

```
host    rangerkms          rangerkms     <ranger_kms_host>/32            trust
```

   where `<ranger_kms_host>` is a host with an installed Ranger KMS component. On your system, the file may be located in a different directory.

Ranger Admin

1. Create a user for the Ranger Admin service.

```
CREATE USER rangeradmin WITH PASSWORD '<password>';
```

   where `<password>` is a password for the `rangeradmin` user.

2. Create a database for the service.

```
CREATE DATABASE ranger;
```

3. Grant permissions to the database for the created user.

```
GRANT ALL PRIVILEGES ON DATABASE ranger TO rangeradmin;
```

4. Add the following line to the */var/lib/pgsql/12/data/pg_hba.conf* file:

```
host    ranger          rangeradmin     <ranger_admin_host>/32            trust
```

   where `<ranger_admin_host>` is a host with an installed Ranger Admin component. On your system, the file may be located in a different directory. Also, if there are several Ranger Admin components in your cluster, you should add a similar record for each Ranger Admin host.

To finish off, edit the */var/lib/pgsql/12/data/postgresql.conf* file by setting the `listen_addresses` parameter to `*` and the `port` parameter to `5432`.

Once it's done, restart the PostgreSQL service:

```
$ sudo systemctl restart postgresql-12
```

Host setup

If you haven't previously, you need to install a JDBC driver on the Ranger KMS and/or Ranger Admin hosts. To do that, follow the steps below:

1. Download the JDBC driver for PostgreSQL. You can obtain it from the PostgreSQL website
2. Put the driver into the */usr/share/java/* directory and rename it to *jdbc-postgresql.jar*.
3. Configure the service in ADCM.

Ranger KMS

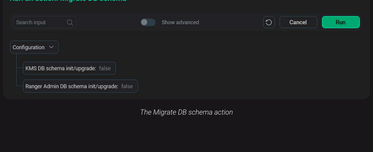
1. On the ***Clusters*** page, find your ADPS cluster and click its name.
2. On the ***Services*** tab, click ***Ranger***.
3. On the opened configuration page, expand the ***dbks-site.xml*** parameter group and fill in the following parameters:
   - `ranger.ks.jpa.jdbc.password` – password for a user in an external database.
   - `ranger.ks.jpa.jdbc.url` – a JDBC connection URL, e.g. `jdbc:postgresql://<database_host>:5432/rangerkms`.
   - `ranger.ks.jpa.jdbc.driver` – a classname for a JDBC driver. For PostgreSQL, it's `org.postgresql.Driver`.
   - `ranger.ks.jdbc.sqlconnectorjar` – a path to an SQL connector JAR, e.g. `/usr/share/java/jdbc-postgresql.jar`.
   - `ranger.ks.jpa.jdbc.user` – name of a user in an external database, e.g. `rangerkms`.
4. In the ***Ranger KMS install.properties*** parameter group, change the `DB_FLAVOR` to `POSTGRES`.
5. Click ***Save***.

Ranger Admin

1. On the ***Clusters*** page, find your ADPS cluster and click its name.
2. On the ***Services*** tab, click ***Ranger***.
3. On the opened configuration page, expand the ***ranger-admin-site.xml*** parameter group and fill in the following parameters:
   - `ranger.jpa.jdbc.password` – password for a user in an external database.
   - `ranger.jpa.jdbc.url` – a JDBC connection URL, e.g. `jdbc:postgresql://<database_host>:5432/ranger`.
   - `ranger.jpa.jdbc.driver` – a classname for a JDBC driver. For PostgreSQL, it's `org.postgresql.Driver`.
   - `ranger.jdbc.sqlconnectorjar` – a path to an SQL connector JAR, e.g. `/usr/share/java/jdbc-postgresql.jar`.
   - `ranger.jpa.jdbc.user` – name of a user in an external database, e.g. `rangeradmin`.
4. In the ***Ranger Admin install.properties*** parameter group, change the `DB_FLAVOR` to `POSTGRES`.
5. Click ***Save***.

Migration

1. Launch the ***Migrate DB schema*** action for Ranger.
2. Choose the components for which you want to launch the migration.



The Migrate DB schema action

3. Confirm the action.

Configure an external MySQL database for Ranger metadata

Sergei Tikhomirov

Contents

To Table of Contents

Overview
MySQL server configuration
Host setup
Migration

Overview

The Ranger's ***Migrate DB schema*** action allows you to migrate a database schema and service users from an internal to an external database. The migration is available for the Ranger KMS and Ranger Admin service components.

Follow the steps below to configure the database server and cluster hosts for the migration.

NOTE

Before migrating to an external database, make sure that it's set up and running.

MySQL server configuration

The example below demonstrates how to set up a MySQL server on a CentOS 7 host.

1. Install the necessary version of the MySQL (MariaDB) server. The CentOS 7 distribution includes a MariaDB 5.5 server by default. To install a different version, refer to the MariaDB documentation.

```
$ sudo yum install mariadb-server
```

2. Start the MariaDB service using `systemctl`.

```
$ sudo systemctl enable mariadb.service --now
```

3. See if you can access the mysql console.

```
$ mysql -u root
```

From this point, refer to the relevant tab below. The commands should be launched in the mysql console.

Ranger KMS

1. Create a database for the service.

```
CREATE DATABASE rangerkms;
```

2. Create a user for the Ranger KMS service.

```
CREATE USER 'rangerkms'@'%' IDENTIFIED BY '<password>';
```

where `<password>` is a password for the `rangerkms` user.

3. Grant permissions to the database for the created user.

```
GRANT ALL PRIVILEGES ON rangerkms.* TO 'rangerkms'@'%';
```

Ranger Admin

1. Create a database for the service.

```
CREATE DATABASE ranger;
```

2. Create a user for the Ranger Admin service.

```
CREATE USER 'rangeradmin'@'%' IDENTIFIED BY '<password>';
```

where `<password>` is a password for the `rangeradmin` user.

3. Grant permissions to the database for the created user.

```
GRANT ALL PRIVILEGES ON ranger.* TO 'rangeradmin'@'%';
```

Once everything is done, restart the MariaDB service:

```
$ sudo systemctl restart mariadb
```

Host setup

If you haven't previously, you need to install a MySQL JDBC driver on the Ranger KMS and/or Ranger Admin hosts. To do that, follow the steps below:

1. Download the driver.

```
$ wget https://downloads.adsw.io/ADH/3.2.4_arenadata2/centos/7/community/x86_64/jdbc-mysql-connector-8.0.33-1036.el7.noarch.rpm
```

2. Install the driver using yum.

```
$ yum localinstall jdbc-mysql-connector-8.0.33-1036.el7.noarch.rpm
```

3. See if the *jdbc-mysql-connector.jar* file has appeared in the */usr/share/java/* directory.
4. Configure the service in ADCM.

Ranger KMS

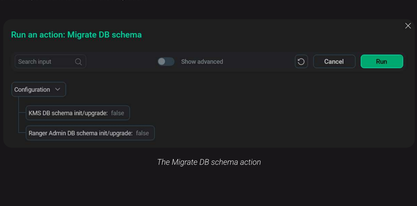
1. On the ***Clusters*** page, find your ADPS cluster and click its name.
2. On the ***Services*** tab, click ***Ranger***.
3. On the opened configuration page, expand the ***dbks-site.xml*** parameter group and fill in the following parameters:
   - `ranger.ks.jpa.jdbc.password` – password for a user in an external database.
   - `ranger.ks.jpa.jdbc.url` – a JDBC connection URL, e.g. `jdbc:mysql://<database_host>:3306/rangerkms`.
   - `ranger.ks.jpa.jdbc.driver` – a classname for a JDBC driver. For MySQL, it's `com.mysql.jdbc.Driver`.
   - `ranger.ks.jpa.jdbc.sqlconnectorjar` – a path to an SQL connector JAR, e.g. `/usr/share/java/jdbc-mysql-connector.jar`.
   - `ranger.ks.jpa.jdbc.user` – name of a user in an external database, e.g. `rangerkms`.
4. In the ***Ranger KMS install.properties*** parameter group, change the `DB_FLAVOR` to `MYSQL`.
5. Click ***Save***.

Ranger Admin

1. On the ***Clusters*** page, find your ADPS cluster and click its name.
2. On the ***Services*** tab, click ***Ranger***.
3. On the opened configuration page, expand the ***ranger-admin-site.xml*** parameter group and fill in the following parameters:
   - `ranger.jpa.jdbc.password` – password for a user in an external database.
   - `ranger.jpa.jdbc.url` – a JDBC connection URL, e.g. `jdbc:mysql://<database_host>:3306/ranger`.
   - `ranger.jpa.jdbc.driver` – a classname for a JDBC driver. For MySQL, it's `com.mysql.jdbc.Driver`.
   - `ranger.jdbc.sqlconnectorjar` – a path to an SQL connector JAR, e.g. `/usr/share/java/jdbc-mysql-connector.jar`.
   - `ranger.jpa.jdbc.user` – name of a user in an external database, e.g. `rangeradmin`.
4. In the ***Ranger Admin install.properties*** parameter group, change the `DB_FLAVOR` to `MYSQL`.
5. Click ***Save***.

Migration

1. Launch the ***Migrate DB schema*** action for Ranger.
2. Choose the components for which you want to launch the migration.



The Migrate DB schema action

3. Confirm the action.

Logging in Ranger

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Available logs

Log settings

- Logback file structure
- Logging levels
- Configuration in ADCM

Available logs

While working, each Ranger component outputs logs into several files. The default log path is */var/log/ranger/<component>/<file>*, where `<component>` is a Ranger component name in lowercase and `<file>` is a log file name. It is only possible to view these logs on each host as they are not displayed in the Ranger Admin UI. For example, here's how to see one of the Ranger Admin log files:

```
$ cat /var/log/ranger/admin/ranger_admin_sql.log
```

You can see the mapping of the Ranger components with the log files in the table below.

Component	Log file	Purpose
Admin	catalina.out	The Tomcat service run logs. It contains everything that is written to Tomcat's `system.out` and `system.err` streams
	access-localhost<date>	Tomcat access log for `<date>`
	ranger_admin_sql.log	Log events of the DBService (see the schema in Ranger architecture) retrieval
	gc-worker.log	The Ranger Admin garbage collection logs
	ranger-admin-<host_fqdn>-ranger.log	The Ranger Admin component run logs
	ranger_admin_perf.log	Ranger performance logs
	ranger_db_patch.log	Patch log for upgrades, installation, and migrations
KMS	catalina.out	The Tomcat service run logs
	kms-audit-<host_fqdn>-kms.log	The Ranger KMS component audit logs
	ranger-kms-<host_fqdn>-kms.log	The Ranger KMS component run logs
UserSync	auth.log	Original service logs before the standard logger interception
	usersync-<host_fqdn>-ranger.log	The Ranger UserSync component run logs

Log settings

Ranger uses the logback framework for logging. It's a successor to widely known Log4j with better performance, native slf4j support, and extended filtration options.

Logback file structure

For example, the default *logback.xml* configuration file for Ranger KMS (with the license information, comments, and some duplicating tags removed) is presented below:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration scan="true">
  <appender name="kms-audit" class="ch.qos.logback.core.rolling.RollingFileAppender"> 1
    <Append>true</Append> 2
    <File>${kms.log.dir}/kms-audit-${hostname}-${user}.log</File> 3
    <encoder> 4
      <pattern>%d{ISO8601} %m%n</pattern>
    </encoder>
    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy"> 5
      <fileNamePattern>${kms.log.dir}/kms-audit-${hostname}-${user}.log.%d{yyyy-MM-dd}</fileNamePattern> 6
      <maxHistory>15</maxHistory> 7
      <cleanHistoryOnStart>true</cleanHistoryOnStart> 8
    </rollingPolicy>
  </appender>
  <appender name="kms" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <File>${kms.log.dir}/ranger-kms-${hostname}-${user}.log</File>
    <Append>true</Append>
    <encoder>
      <pattern>%d{ISO8601} %-5p [%t] %c{1} \(%F:%L\) - %m%n</pattern>
    </encoder>
    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
      <fileNamePattern>${kms.log.dir}/ranger-kms-${hostname}-${user}.log.%d{yyyy-MM-dd}</fileNamePattern>
      <maxHistory>15</maxHistory>
      <cleanHistoryOnStart>true</cleanHistoryOnStart>
    </rollingPolicy>
  </appender>
  <logger name="kms-audit" additivity="false" level="INFO"> 9
    <appender-ref ref="kms-audit"/> 10
  </logger>
  <root level="WARN"> 11
    <appender-ref ref="kms"/>
  </root>
</configuration>
```

The *logback.xml* file can have many sections, information about which you can find in the logback documentation, but here are descriptions of the used sections:

1. appender. The `<appender>` tag represents an object that contains all the information necessary for logging: output file, log format, what to log and when to log it. Logback offers several types of appenders, but by default only RollingFileAppender is used.
2. Append. The `<Append>` tag contains a boolean value that defines whether to append the log file or truncate it each time.
3. File. The `<File>` tag contains the name of the output log file. If the file doesn't exist, it is created.
4. encoder. The `<encoder>` object transforms an event into a byte array. The `<pattern>` parameter inside contains a layout defining what the output should look like.
5. rollingPolicy. This policy is responsible for rollover which involves file moving and renaming. There are many types of rolling policies, but the one used in Ranger is TimeBasedRollingPolicy which defines the rolling policy based on time (e.g. day, week, etc.).
6. fileNamePattern. This parameter's value consists of a filename and `%d` section which may contain the date and time pattern. If the pattern is omitted, the default `yyyy-MM-dd` pattern is used. The rollover period is derived from this parameter's value.
7. maxHistory. Defines how many log files can be kept at the same time. When the value gets exceeded, the older files will be deleted asynchronously.
8. cleanHistoryOnStart. Defines whether to delete an archive when the appender starts up.
9. **logger**. A logger definition. Here, you can specify the logging level for this logger.
10. **appender-ref**. An appender for this logger to use.
11. **root**. The root logger definition.

Logging levels

Logback offers the following logging levels:

- `OFF` — logging is turned off.
- `ERROR` — logs all the errors or events that may lead to an unexpected behavior.
- `WARN` — logs all the unusual events that may signal the unintended use.
- `INFO` — logs all the irregular or rare operations such as configuration loading.
- `DEBUG` — logs all the major events. Mostly used for error localization as the name suggests.
- `TRACE` — logs all the events. You can use it to localize an error if the `DEBUG` level isn't sufficient.

Each subsequent level includes the logs of all the previous levels.

Configuration in ADCM

Each Ranger component has a configuration setting called ***logback.xml***, where you can see the current log format, output file location, log level for certain services, and maximum history records. To edit this file in ADCM, follow the steps below:

1. On the ***Clusters*** page, select your ADPS cluster.
2. Go to the ***Services*** tab and select ***Ranger***.
3. Head to the ***Components*** tab. On this tab, select the component, for which you want to edit the log settings.
4. Edit the ***logback.xml*** parameter to your needs, save the configuration, and restart Ranger.

Configure policies in Ranger: advanced examples

A. Serge Tchuisseu

Contents

To Table of Contents

Overview
Geolocation-based policy
- Preparation
- Update the service definition
- Create/edit a policy

Time-based policy
- Update the service definition
- Create/edit a policy

Custom script condition

Overview

A Ranger policy is a handy tool for access control. This article contains some theory about policy conditions and examples of complex policies. You can read about the basic policy configuration in the Ranger policies in Ranger article.

Advanced Ranger policies require us to update the service definition which can be done via REST API. For some policies, you may need to add a context enricher that allows Ranger to get additional information from the context. The following context enrichers are available:

- `RangerTrustedProxyAuditProvider` – adds geolocation information to an access request.
- `RangerUgiGroupEnricher` – adds information about datasets, resources included in each dataset, and accesses allowed through datasets to an access request.

You can create your own context enricher by implementing the `RangerContextEnricher` interface to supply custom information with a request.

If you want to add a non-default policy condition, you can do so by updating the `policyConditions` section of the service definition. A policy condition requires you to define a condition evaluator and, optionally, evaluator options. The following condition evaluators are available:

- `RangerAccessedFromClusterCondition` – checks the name of the cluster from which the request came.
- `RangerAccessedFromClusterTypeCondition` – checks the type of the cluster from which the request came.
- `RangerAccessedFromFromClusterCondition` – reverse of the `RangerAccessedFromClusterCondition` condition.
- `RangerAccessedFromFromClusterTypeCondition` – reverse of the `RangerAccessedFromClusterTypeCondition` condition.
- `RangerAnyExpressionTagsPresentConditionEvaluator` – checks if a resource contains any of the policy condition tags.
- `RangerContextAttributeIsInEvaluatorCondition` – checks if a value is in some whitelist.
- `RangerContextAttributeIsNotInEvaluatorCondition` – reverse of the `RangerContextAttributeIsInEvaluatorCondition` condition.
- `RangerHierResourcesAccessedTogetherCondition` – checks if a set of resources can be accessed simultaneously.
- `RangerHierResourcesNotAccessedTogetherCondition` – reverse of the `RangerHierResourcesAccessedTogetherCondition` condition.
- `RangerHiveUdfAllow` – evaluates if a Hive UDF request from some user is allowed to access the resource.
- `RangerNoneOfExpectedTagsPresentConditionEvaluator` – checks if a resource contains none of the policy condition tags.
- `RangerScriptConditionEvaluator` – evaluates a JS script condition.
- `RangerTagsAtMostNConditionEvaluator` – checks if a resource contains all of the policy condition tags.
- `RangerTimeOfDayMatcher` – evaluates to true if the request came within the allowed time frame.

You can mix and match these conditions based on your needs. Below, you can find some examples of useful policies.

Geolocation-based policy

This type of policy allows you to control the resource access based on where geographically it was requested from.

Preparation

Prepare a text file with the information about the allowed IPs for certain locations. See the example below:

```
IP_FROM,IP_TO,COUNTRY_CODE,COUNTRY_NAME,REGION_CITY
91.198.240.1,91.198.240.45,FR,France,Ile-de-France,Paris
```

Here, the `IP_FROM` value marks the start of the allowed IP range, while `IP_TO` marks its end. Other fields specify the location for which the IP range is applicable. This file should be placed on a Ranger Admin host in location where it can be accessed by Ranger. For example, /etc/ranger/geo-data.txt.

Update the service definition

Update the `service-definition` file for the necessary service. The following example updates a configuration for the HBase service:

1. Get the current service definition.

```
$ curl --user <admin_user>:<admin_pwd> --get "http://<ranger_host>/service/public/v2/api/servicedef/name/hbase"
```

when `<admin_user>` and `<admin_pwd>` are credentials for a Ranger user with admin rights, and `<ranger_host>` is a URL of the Ranger Admin host including the port number (6080 for HTTP and 6182 for HTTPS).

As a response, you should receive a JSON string that contains the service definition.

2. Edit the JSON by appending it with the following section:

```
"contextEnrichers": [
  {
    "enricher": "org.apache.ranger.plugin.contextenricher.RangerFileBasedGeolocationProvider",
    "enricherOptions": {
      "filePath": "/etc/ranger/geo-data.txt",
      "fileFormat": "csv"
    }
  }
]
```

The `FilePath` here should contain the path to the IP information file you have created earlier.

3. Append the `policyConditions` section with the following lines to add the ability to base your policies on a request being sent from a certain location:

```
{
  "itemId": 1,
  "name": "location-inside",
  "label": "Requested from inside of the location",
  "description": "Enter location",
  "evaluator": "org.apache.ranger.plugin.conditionevaluator.RangerContextAttributeValueInCondition",
  "evaluatorOptions": {
    "attributeName": "LOCATION_COUNTRY_CODE"
  }
}
```

Alternatively, you can create a reverse condition that will check if a request came from outside a certain location:

```
{
  "itemId": 2,
  "name": "location-outside",
  "label": "Requested from outside of the location",
  "description": "Enter location",
  "evaluator": "org.apache.ranger.plugin.conditionevaluator.RangerContextAttributeValueNotInCondition",
  "evaluatorOptions": {
    "attributeName": "LOCATION_COUNTRY_CODE"
  }
}
```

Note that the `attributeName` parameter value starts with `LOCATION_` followed by the column name from the IP information file. The `LOCATION_` prefix here is required since it's automatically added by the context enricher.

4. Send the updated service definition to Ranger Admin using the command below:

```
$ curl -i -X "Content-Type: application/json" --user <admin_user>:<admin_pwd> -X PUT --data "<json>"
"http://<ranger_host>/service/public/v2/api/servicedef/name/hbase"
```

It is expected that your JSON has been formatted into a single string, you can use services like JSON-Minifier to achieve that. Alternatively, you can send a JSON file. The correct response contains the `200 OK` HTTP code, some meta information, and the new service definition.

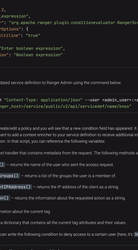
Create/edit a policy

You can create/edit a policy with the Ranger's REST API, but the example below utilizes the web interface:

1. If you are creating a policy, follow the common creation steps described in Add a policy. In case you want to edit an existing policy, see Edit a policy.
2. In the allow/deny conditions section, you will see that new conditions have appeared:

New conditions

3. Set up the condition. In this example, the condition checks the country code:



A geolocation policy

4. Once you're done, click **Add** or **Save**, at the bottom of the page.

Time-based policy

This type of policy allows you to control the resource access based on some changing aspect of the time of day it was requested at.

Update the service definition

Update the service definition file for the necessary service. The following example updates a configuration for the HBase service:

1. Get the current service definition.

```
$ curl --user <admin_user>:<admin_pwd> --get "http://<ranger_host>/service/public/v2/api/servicedef/name/hbase"
```

when `<admin_user>` and `<admin_pwd>` are credentials for a Ranger user with admin rights, and `<ranger_host>` is a URL of the Ranger Admin host including the port number (6080 for HTTP and 6182 for HTTPS).

As a response, you should receive a JSON string that contains the service definition.

2. Append the `policyConditions` section with the following lines to base your policies on a request being sent during a certain time frame:

```
{
  "itemId": 4,
  "name": "time-frame",
  "label": "Requested within the time frame",
  "description": "Enter time frame",
  "evaluator": "org.apache.ranger.plugin.conditionevaluator.RangerTimeOfDayMatcher",
}
```

3. Send the updated service definition to Ranger Admin using the command below:

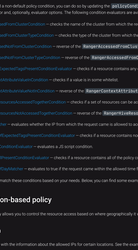
```
$ curl -i -X "Content-Type: application/json" --user <admin_user>:<admin_pwd> -X PUT --data "<json>"
"http://<ranger_host>/service/public/v2/api/servicedef/name/hbase"
```

It is expected that your JSON has been formatted into a single string, you can use services like JSON-Minifier to achieve that. The correct response contains the `200 OK` HTTP code, some meta information, and the new service definition. Alternatively, you can send a JSON file.

Create/edit a policy

You can create/edit a policy with the Ranger's REST API, but the example below utilizes the web interface:

1. If you are creating a policy, follow the common creation steps described in Add a policy. In case you want to edit an existing policy, see Edit a policy.
2. In the allow/deny conditions section, you will see that new conditions have appeared:



A new time condition

3. Set up the condition. The condition expects a time frame in the `Oct-MM3 AM - Cec-MM3 PM` format, for example, `9:30 AM - 1:15 PM`.

A time frame policy

4. Once you're done, click **Add** or **Save**, if you edited an existing policy) at the bottom of the page.

The result of such policy can be seen on the **Access** tab of the **Audits** section.

Custom script condition

By default, some Ranger plugins have the `RangerScriptConditionEvaluator` policy condition enabled. For others (e.g. Knox), you can enable it by editing the service definition:

1. Get the current service definition:

```
$ curl --user <admin_user>:<admin_pwd> --get "http://<ranger_host>/service/public/v2/api/servicedef/name/knox"
```

when `<admin_user>` and `<admin_pwd>` are credentials for a Ranger user with admin rights, and `<ranger_host>` is a URL of the Ranger Admin host including the port number (6080 for HTTP and 6182 for HTTPS).

As a response, you should receive a JSON string that contains the service definition.

2. Append the `policyConditions` section with the following lines to base your policies on a custom script condition:

```
{
  "itemId": 2,
  "name": "_expression",
  "evaluator": "org.apache.ranger.plugin.conditionevaluator.RangerScriptConditionEvaluator",
  "evaluatorOptions": {
    "script": "true"
  },
  "label": "Enter boolean expression",
  "description": "Boolean expression"
}
```

3. Send the updated service definition to Ranger Admin using the command below:

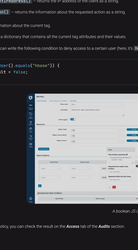
```
$ curl -i -X "Content-Type: application/json" --user <admin_user>:<admin_pwd> -X PUT --data "<json>"
"http://<ranger_host>/service/public/v2/api/servicedef/name/knox"
```

After that, try to create/edit a policy and you will see that a new condition field has appeared. It allows you to write a JS script which evaluates to `true` or `false`. Depending on your needs, you may want to add a context enricher to your service definition to receive additional information with the access request, see how to do it in second step of the Geolocation-based policy section. In that script, you can access the following variables:

- `ctx` – context handler that contains metadata from the request. The following methods are available:
  - `getUser()` – returns a name of the user who sent the request.
  - `getUserGroups()` – returns a list of the groups the user is a member of.
  - `getClientIPAddress()` – returns the IP address of the client as a string.
  - `getAction()` – returns the information about the requested action as a string.
- `tag` – information about the current tag.
- `tagAttr` – a dictionary that contains all the current tag attributes and their values.

For example, you can write the following condition to deny access to a certain user (here, it's `hbase` for the demonstrative purposes):

```
if (ctx.getUser().equals("hbase")) {
    ctx.result = false;
}
```



A boolean JS condition

After saving the policy, you can check the result on the **Access** tab of the **Audits** section.

Ranger service management via ADCM

Sergei Tikhomirov

To Table of Contents

The ADCM UI provides actions to manage the Ranger service and its components. For information on how to run service actions, refer to ADPS service actions.

The actions available for the Ranger service are listed in the table below.

Action	Description
Check	Runs service-specific tests to check the health of the service and its components
Start	Starts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied on the service startup. Otherwise, the service starts without applying configurations from ADCM
Stop	Stops the service
Restart	Restarts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied during the service restart. Otherwise, the service restarts without applying configurations from ADCM
Remove	Removes the service from the cluster. This action should be used to remove already installed services. Whereas the control can be used to remove a non-mapped service (a service which components have not been distributed among cluster hosts)
Add/Remove components	Reconfigures the service topology. With it, you can change the amount of Ranger Admin, KMS, and User synchronizer components. Running this action opens the host-component mapping interface where you can add new service components
Validate configuration	Checks ports for conflicts
Set password	Allows you to set a password for a user. Running this action opens a window in which you need to specify a username and a password that will be set for that user
Migrate DB schema	Allows you to migrate from an internal database to an external one (MySQL/PostgreSQL)

Ranger Solr plugin

Sergei Tikhomirov

To Table of Contents

The Ranger Solr plugin set up process for ADPS is identical to the one for ADH.

Solr service management via ADCM

Sergei Tikhomirov

To Table of Contents

The ADCM UI provides actions to manage the Solr service and its components. For information on how to run service actions, refer to ADPS service actions.

The actions available for the Solr service are listed in the table below.

Action	Description
Check	Runs service-specific tests to check the health of the service and its components
Start	Starts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied on the service startup. Otherwise, the service starts without applying configurations from ADCM
Stop	Stops the service
Restart	Restarts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied during the service restart. Otherwise, the service restarts without applying configurations from ADCM
Remove	Removes the service from the cluster. This action should be used to remove already installed services. Whereas the control can be used to remove a non-mapped service (a service which components have not been distributed among cluster hosts)
Add/Remove components	Reconfigures the service topology. With it, you can change the amount of the Solr Server components. Running this action opens the host-component mapping interface where you can add new service components
Manage Ranger plugin	Enables or disables the Ranger Solr plugin
Update collection expiration	Updates the expiration time (the `_expire_at_` field) for a specific Solr collection. When you run this action, the following options are available: ■ ***Collection name*** — the Solr collection to update. ■ ***Expire at*** — the timestamp string that sets the expiration time for documents indexed in Solr. For example, `2023-07-28T18:05:03.371Z`. ■ ***Batch size*** — the number of documents to process per one batch.

ZooKeeper service management via ADCM

Sergei Tikhomirov

To Table of Contents

The ADCM UI provides actions to manage the ZooKeeper service and its components. For information on how to run service actions, refer to ADPS service actions.

The actions available for the ZooKeeper service are listed in the table below.

Action	Description
Check	Runs service-specific tests to check the health of the service and its components
Start	Starts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied on the service startup. Otherwise, the service starts without applying configurations from ADCM
Stop	Stops the service
Restart	Restarts the service. When you run this action, the option ***Apply configs from ADCM*** is available. When selected, all service configurations defined in ADCM settings will be applied during the service restart. Otherwise, the service restarts without applying configurations from ADCM
Remove	Removes the service from the cluster. This action should be used to remove already installed services. Whereas the control can be used to remove a non-mapped service (a service which components have not been distributed among cluster hosts)
Expand	Adds ZooKeeper Server components to your cluster. Running this action opens the host-component mapping interface where you can add new ZooKeeper Server components
Shrink	Removes ZooKeeper Server components from your cluster. Running this action opens the host-component mapping interface where you can remove ZooKeeper Server components from specific hosts
Move Zookeeper Server(s)	Moves ZooKeeper Server components among hosts. Running this action opens the host-component mapping interface where you can move ZooKeeper Server components

Arenadata Platform Security cluster actions

Sergei Tikhomirov

Contents

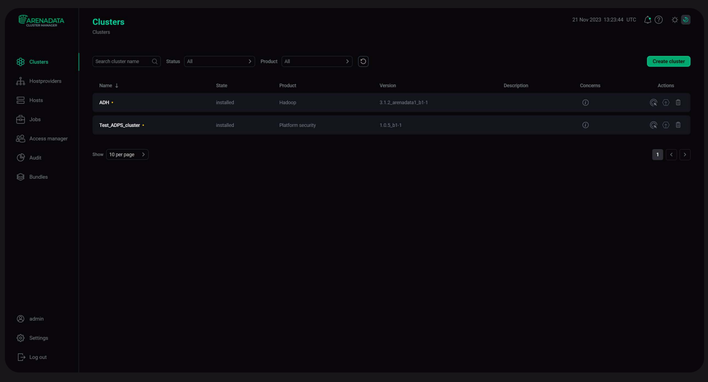
To Table of Contents

Overview

Actions

Overview

This article describes the actions available for the ADPS cluster in the ADCM UI. You can find cluster actions on the ***Clusters*** page.

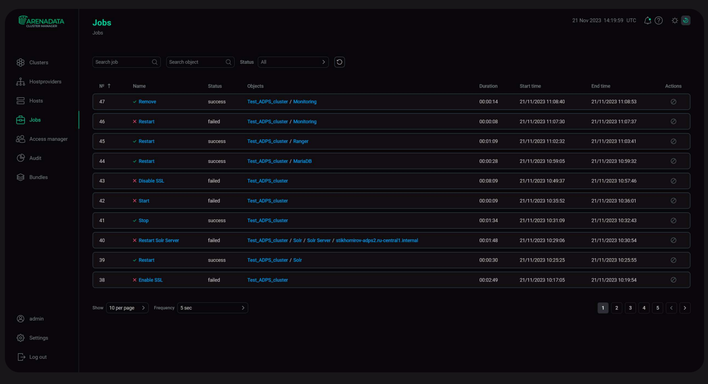


The Clusters page

The ***Clusters*** page displays a table with the following columns:

- ***Name*** — the cluster name specified during the creation.
- ***State*** — the current cluster state (e.g., `created`, `installed`, `running`).
- ***Product*** — product name.
- ***Version*** — version of a bundle used to install the cluster.
- ***Description*** — the description specified during the cluster creation.
- ***Concerns***. The ⓘ icon indicates that the cluster is configured correctly. If critical errors are detected in the cluster configuration, the icon image is changed to ⓘ — when you hover over it, a tooltip appears showing an error description with a link that you can follow to perform the required configuration.
- ***Actions***. The column shows icons for managing a cluster:
  - — opens a drop-down menu that offers actions to manage the cluster.
  - — indicates whether a new version of a bundle is available and allows you to upgrade the cluster version.
  - — deletes information about the cluster from ADCM (it does not remove ADPS or make any changes to hosts that belong to the cluster).

You can use icons described above to perform cluster actions. You can monitor the execution process on the ***Jobs*** page.



The Jobs page

Actions

TIP

Most of the actions have the ***Verbose*** flag used to generate more detailed action execution logs.

The following actions are available for the ADPS cluster:

- Check
- Manage Kerberos
- Manage SSL
- Reinstall status-checker
- Start
- Stop
- Validate configuration

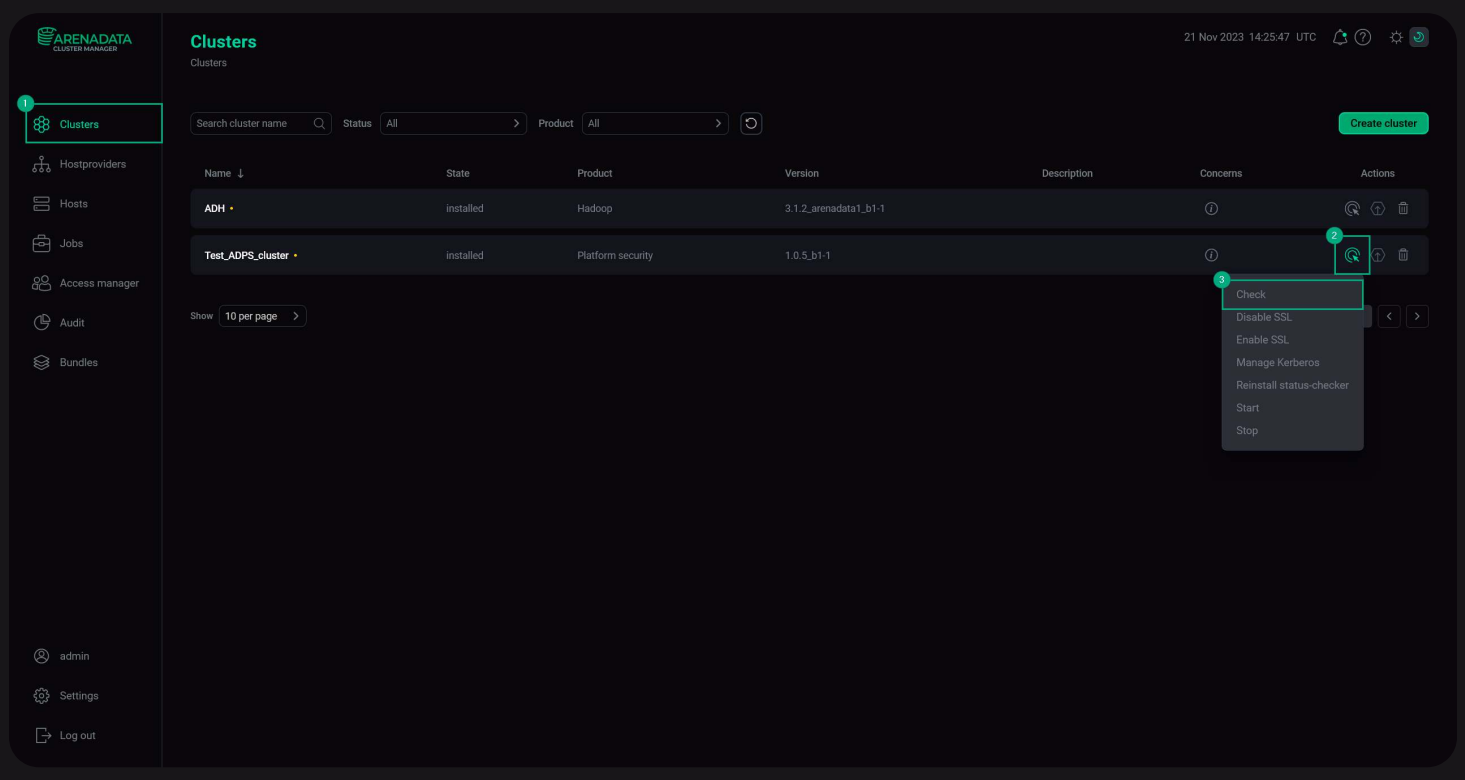
Check

Sergei Tikhomirov

To Table of Contents

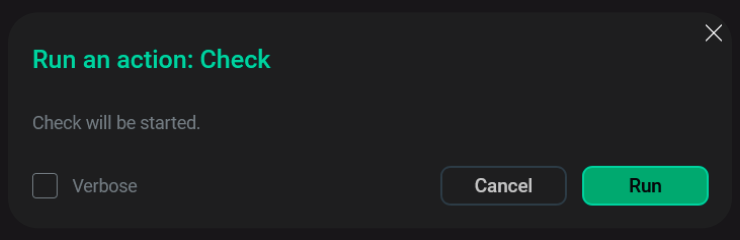
The ***Check*** action verifies whether ADPS services work correctly. It sequentially runs service-specific tests for each service to check the health of the service and its components.

To run the action, go to the ***Clusters*** page, find your ADPS cluster, click in the ***Actions*** column, and select the action from the drop-down menu that appears.



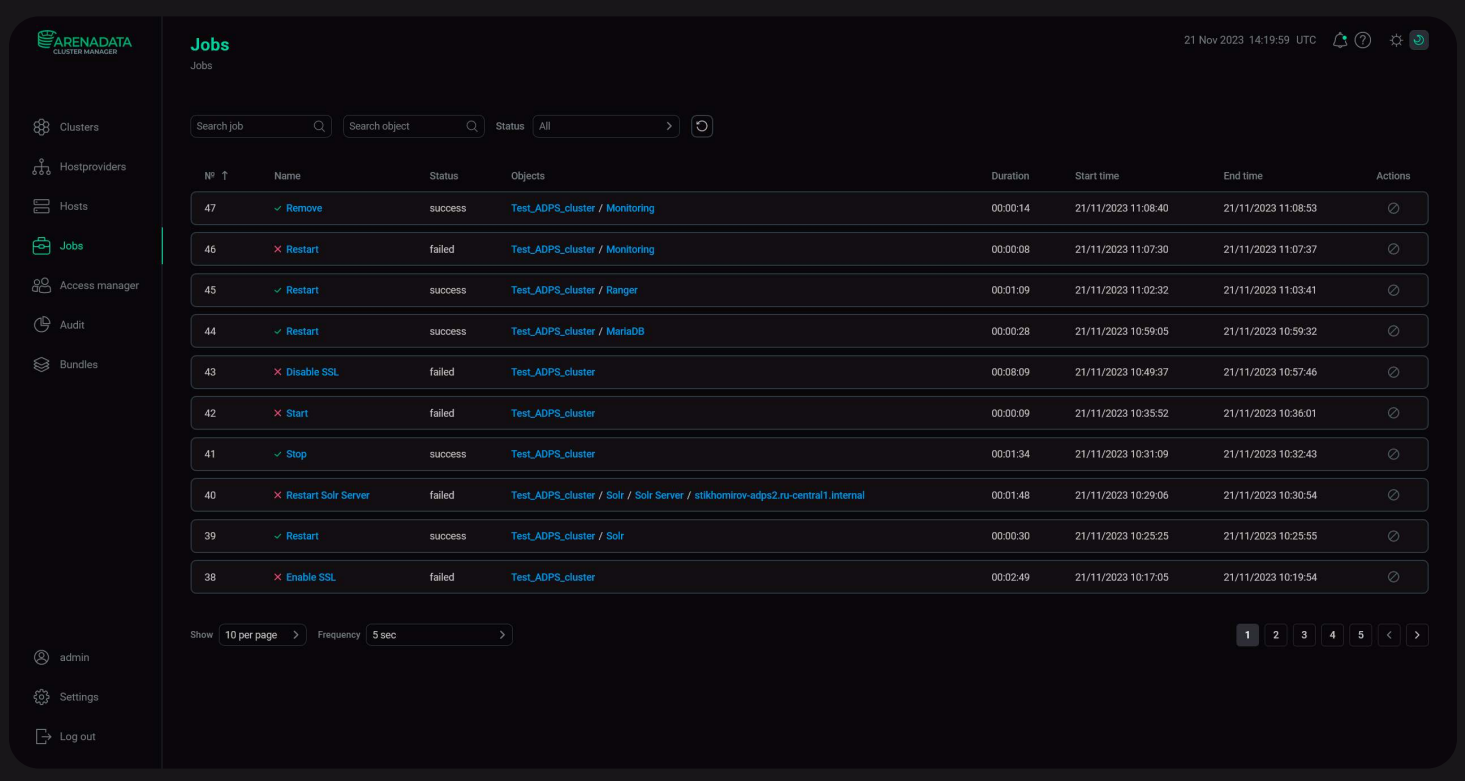
Running an action

When you choose an action, ADCM displays a dialog window to confirm the choice. In this dialog window, you can select the ***Verbose*** checkbox to see additional execution details.



Confirmation window

You can find execution process and result of the action on the ***Jobs*** page. On this page you can view action execution details, including verbose Ansible STDOUT/STDERR outputs.



The Jobs page

Manage SSL

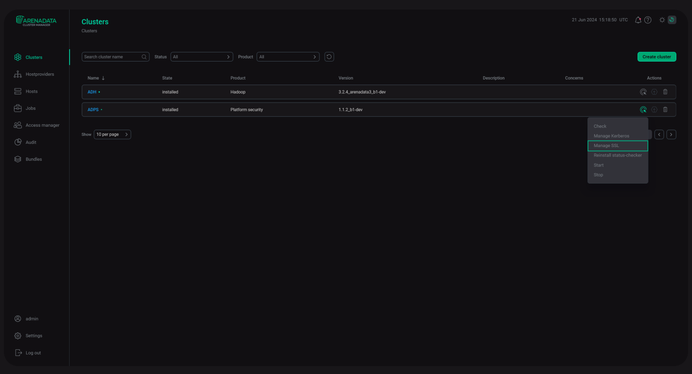
Sergei Tikhomirov

To Table of Contents

The ***Manage SSL*** action allows you to control the SSL encryption of all services in an ADPS cluster. For additional information on SSL settings, see Requirements for SSL certificates.

In ADPS, SSL is enabled via Knox. You can read more about it in the Enable SSL on ADPS cluster with Knox article.

To run the action, go to the ***Clusters*** page, click in the ***Actions*** column, and select the action from the drop-down list that appears.



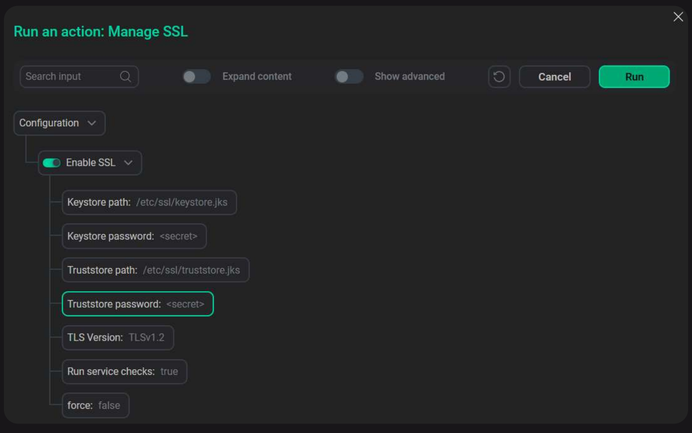
Execute the "Manage SSL" action

When you choose an action, ADCM displays a dialog window where you need to specify SSL settings listed in the table below.

SSL settings

Parameter	Description	Default value
Keystore path	A path to the keystore file	/etc/ssl/keystore.jks
Keystore password	A password for the keystore file	—
Truststore path	A path to the truststore file	/etc/ssl/truststore.jks
Truststore password	A password for the truststore file	—
TLS Version	Version of the TLS protocol. See Set TLS version for an ADH cluster	TLSv1.2
Run service checks	Runs the service checks after the setting change	True
force	Defines whether current SSL parameters should be overwritten with the specified values	False

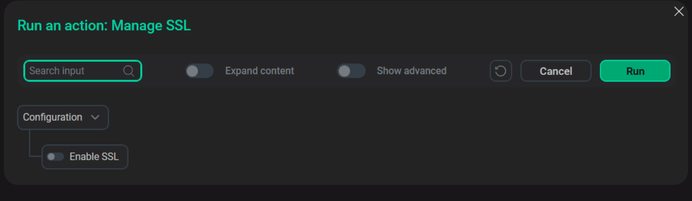
To enable SSL, run the action with the ***Enable SSL*** parameter group active and the parameters filled in.



Enable SSL with the "Manage SSL" action

After you specify the parameters, click ***Run***. In the action confirmation window that appears, you can select the ***Verbose*** checkbox to see additional execution details.

To disable SSL, run the ***Manage SSL*** action with the ***Enable SSL*** parameter group inactive.



Disable SSL with the "Manage SSL" action

During the execution of the ***Manage SSL*** action, ADCM stops all services one by one, reconfigures them to enable or disable SSL encryption, then starts the services and checks them.

You can find execution process and result of the action on the ***Jobs*** page. On this page you can view action execution details, including verbose Ansible STDOUT/STDERR outputs.

IMPORTANT

If the Ranger Solr plugin is enabled, make sure to manually restart Solr after enabling/disabling SSL.

Manage Kerberos

Sergei Tikhomirov

Contents

To Table of Contents

Custom kerberization settings

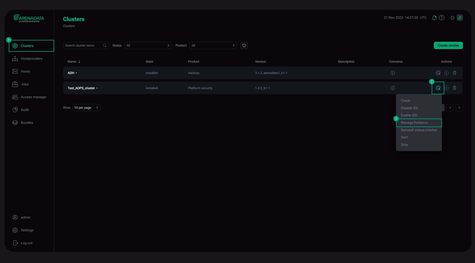
Custom keytab recommendations

The ***Manage Kerberos*** action encapsulates enabling, reconfiguring, and disabling of Kerberos for the cluster services. To run it, you need an installed ADPS cluster (see Get started with Arenadata Platform Security) and one or more KDCs (Key Distribution Center).

NOTE

This action has the same purpose as the Manage Kerberos action in ADH.

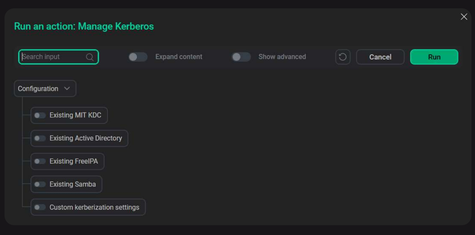
To run the action, go to the ***Clusters*** page, find your ADPS cluster, click in the ***Actions*** column, and select the action from the drop-down menu that appears.



Running an action

The pop-up window suggests several options to run the action:

- Using an existing MIT KDC.
- Using an existing MS Active Directory.
- Using an existing FreeIPA.
- Using an existing Samba.



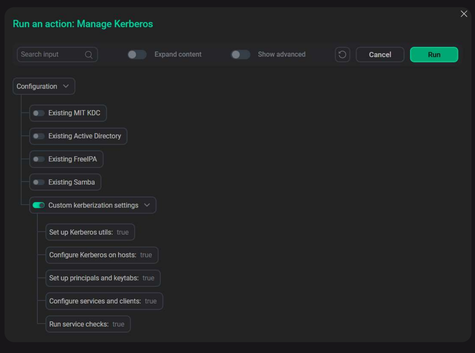
Kerberos activation options

IMPORTANT

Running the action with one KDC type enabled will trigger Kerberos activation.

Custom kerberization settings

The ***Custom kerberization settings*** option allows the user to choose kerberization steps, for example, creation of principals and keytabs.



Custom kerberization settings parameters

Custom kerberization settings parameter description

Parameter	Description	Default value
Set up Kerberos utils	Enables installation or removal of Kerberos clients and utils. Affects the ***Expand*** and ***Install*** actions	True
Configure Kerberos on hosts	Enables cluster configuration, including *krb5.conf*, *ldap.conf*	True
Set up principals and keytabs	Enables creation, recreation, or removal of principals and keytabs. Passwords for principals are generated randomly before keytab creation. Affects the ***Expand*** and ***Install*** actions. ADCM bundle will set up owner and permissions for keytabs only if this checkbox is selected in the cluster configuration. In case of absence of admin permissions, a customer should provide the prepared keytabs with correctly set owner and permissions (see Custom keytab recommendations)	True
Configure services and clients	Enables updating of services and clients configuration	True
Run service checks	Enables service check runs	True

Custom keytab recommendations

Below is the table with recommendations for owners, groups, and permissions for keytabs.

Keytab recommendations

Component short name	Keytab owner	Keytab group	Permissions
zookeeper	zookeeper	zookeeper	600
solr	solr	solr	600
rangeradmin	ranger	ranger	600
rangerkms	kms	kms	600
rangerlookup	ranger	ranger	600
rangertagsync	ranger	ranger	600
rangerusersync	ranger	ranger	600
knox	knox	knox	600
HTTP	root	hadoop	640

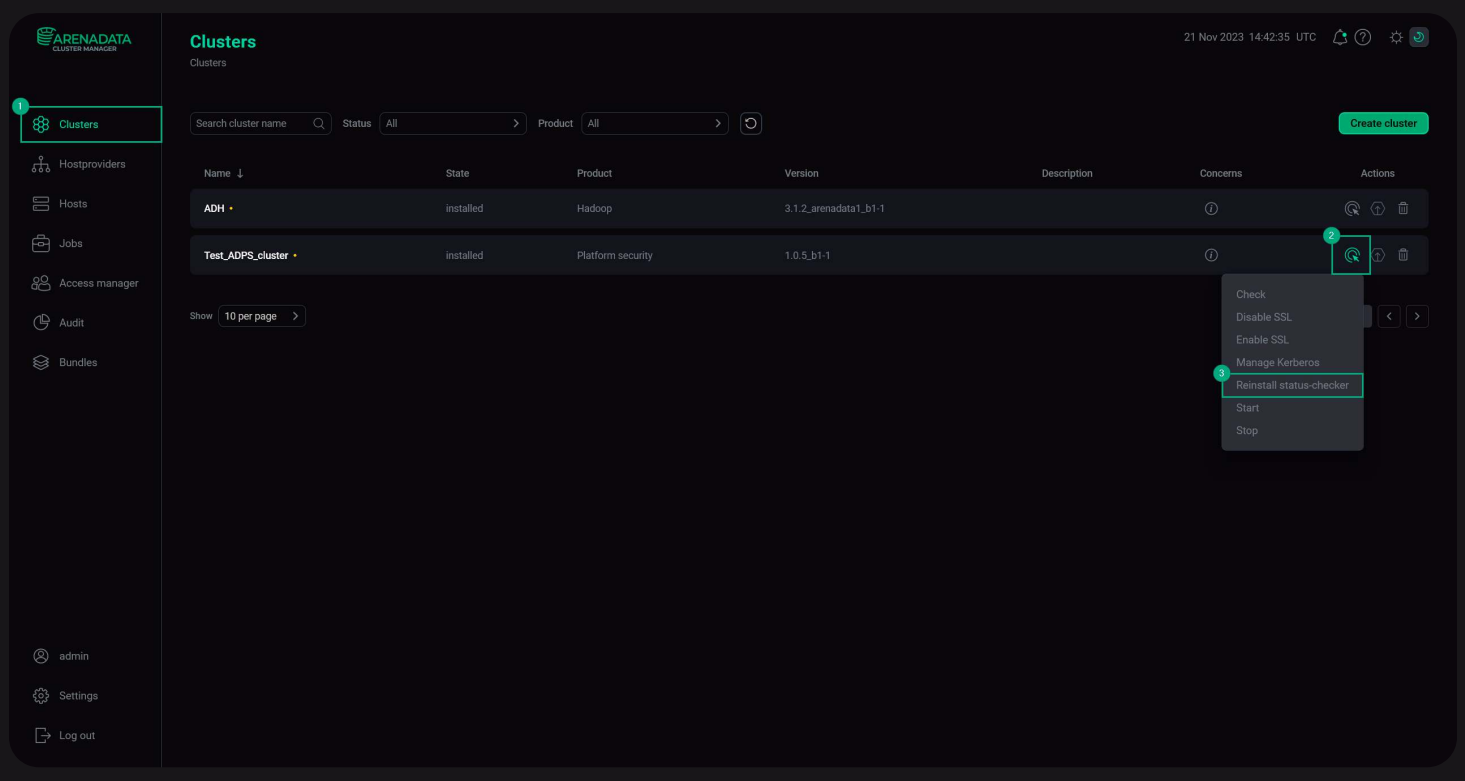
Reinstall status-checker

Sergei Tikhomirov

To Table of Contents

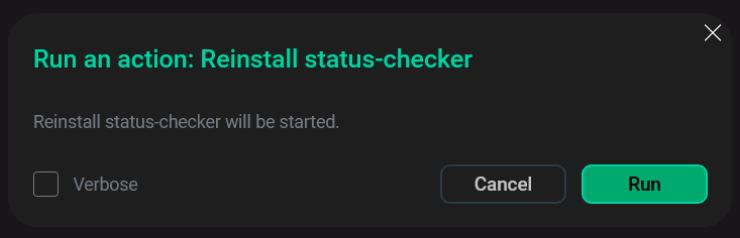
The ***Reinstall status-checker*** action reinstalls status-checker for all services one by one.

To run the action, go to the ***Clusters*** page, find your ADPS cluster, click in the ***Actions*** column, and select the action from the drop-down menu that appears.



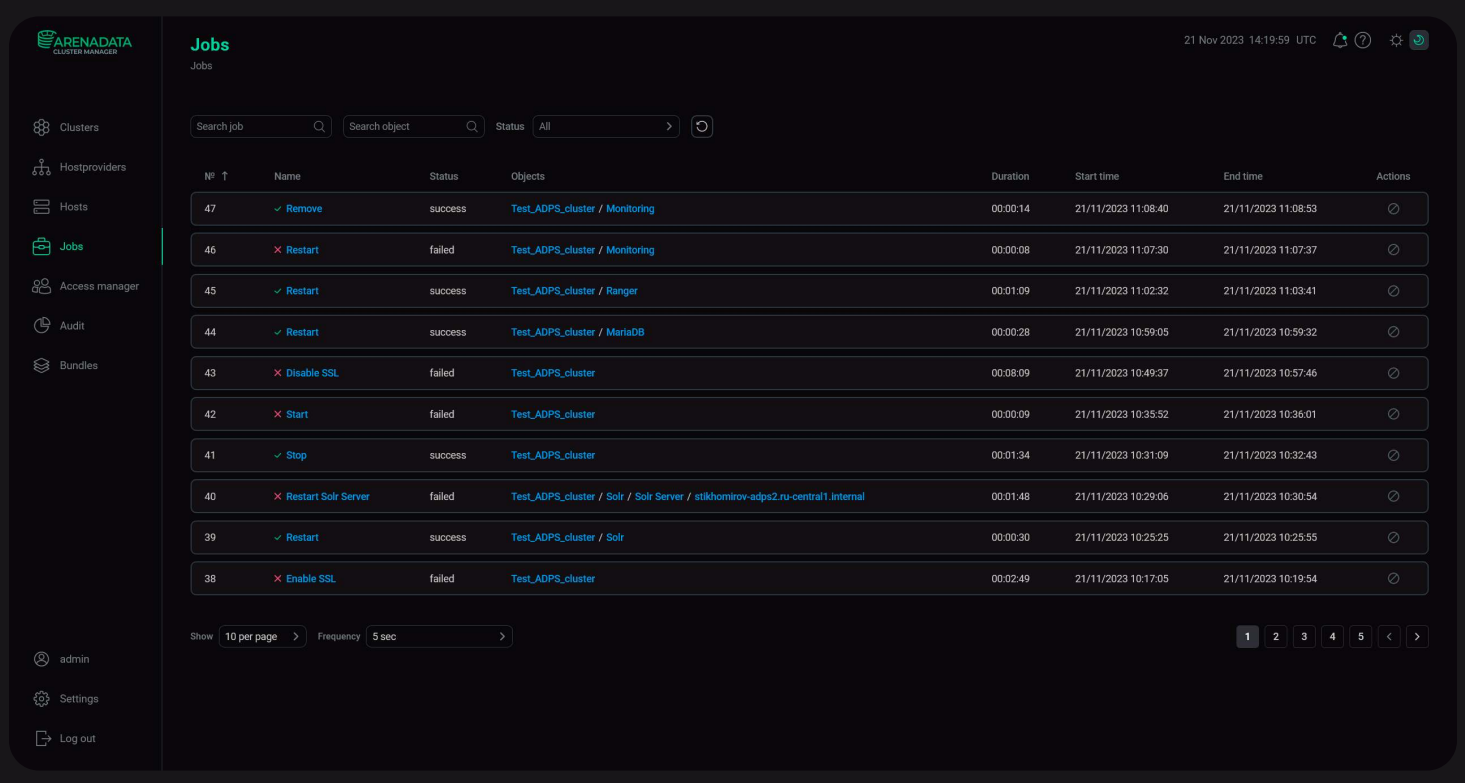
Running an action

When you choose an action, ADCM displays a dialog window to confirm the choice. In this dialog window, you can select the ***Verbose*** checkbox to see additional execution details.



Confirmation window

You can find execution process and result of the action on the ***Jobs*** page. On this page you can view action execution details, including verbose Ansible STDOUT/STDERR outputs.



The Jobs page

Start

Sergei Tikhomirov

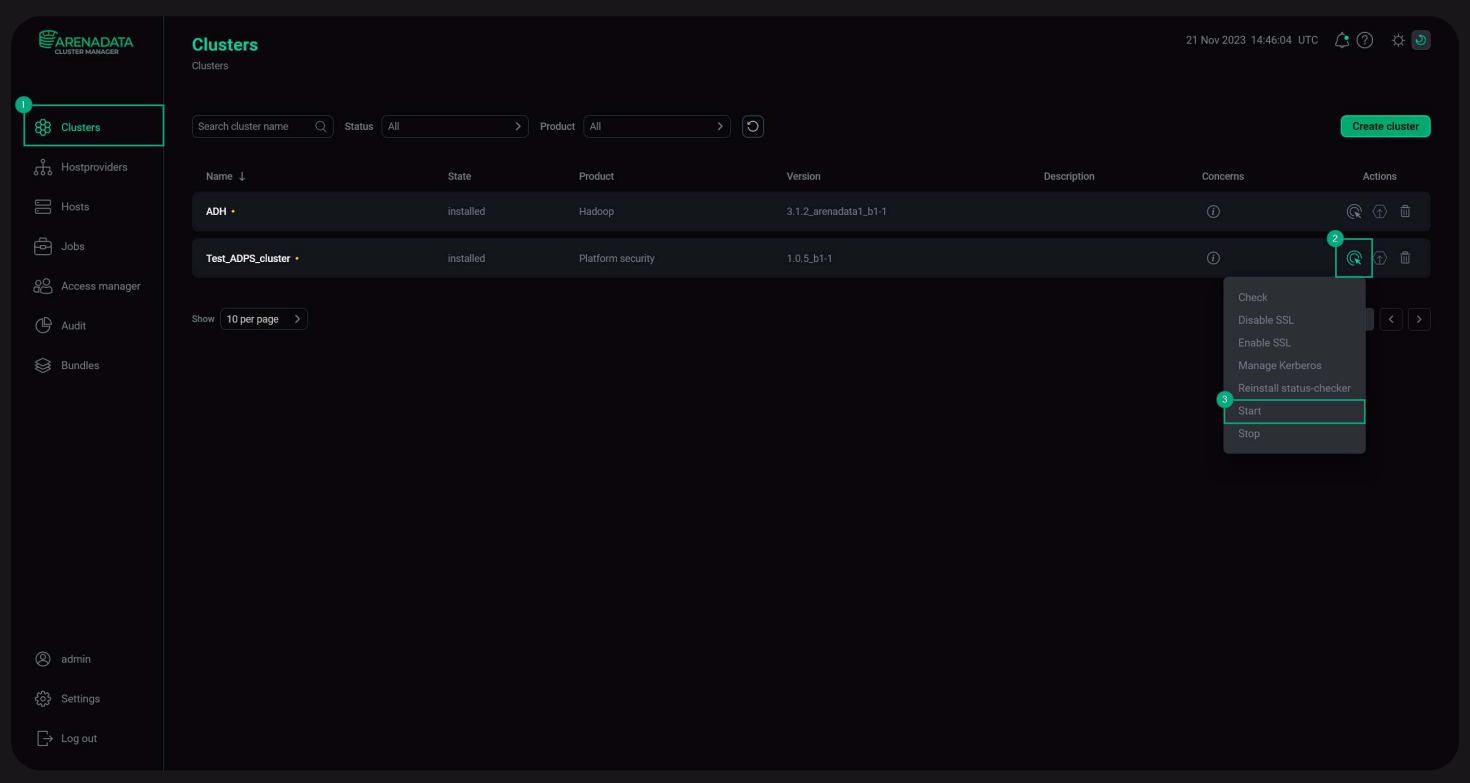
To Table of Contents

IMPORTANT

If you installed a monitoring cluster and want to use it with the ADPS cluster, start the monitoring cluster first. To do this, run the ***Start*** action of the monitoring cluster in the same way as described for the ADPS cluster.

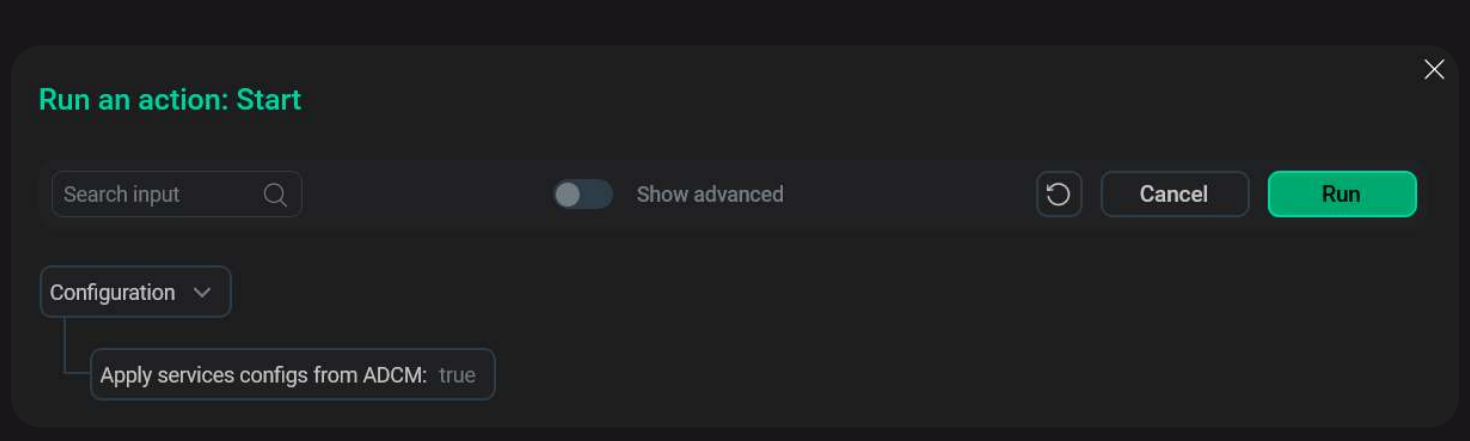
The ***Start*** action starts all the services installed in the ADPS cluster, one by one, in the correct order.

To run the action, go to the ***Clusters*** page, find your ADPS cluster, click in the ***Actions*** column, and select the action from the drop-down menu that appears.



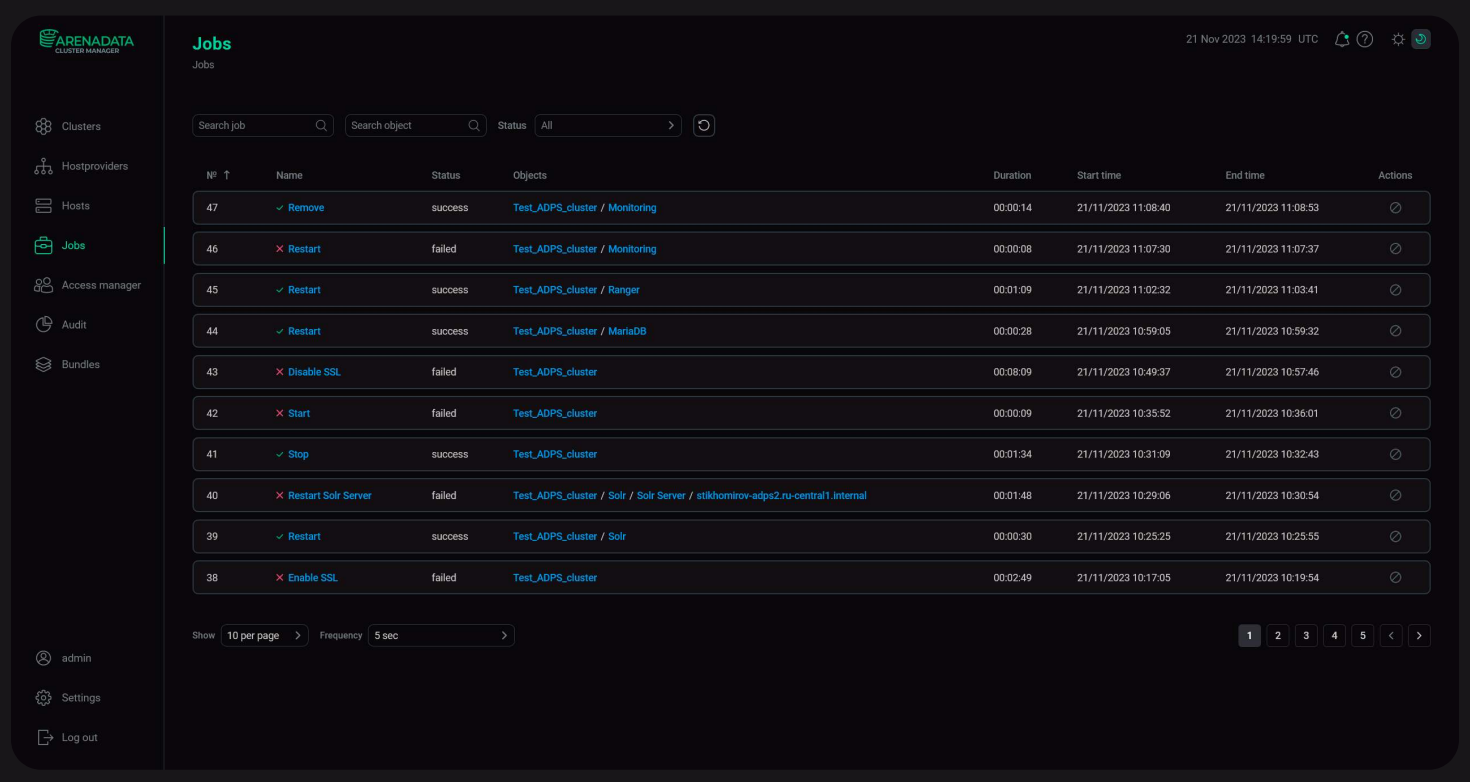
Running an action

When you choose an action, ADCM displays a dialog window to confirm the choice. In this dialog window, you can select the ***Verbose*** checkbox to see additional execution details. The dialog window of the ***Start*** action also contains the ***Apply services configs from ADCM*** chekbox that allows you to determine whether configuration options specified in ADCM UI should be applied to ADPS services.



Confirmation window

You can find execution process and result of the action on the ***Jobs*** page. On this page you can view action execution details, including verbose Ansible STDOUT/STDERR outputs.



The Jobs page

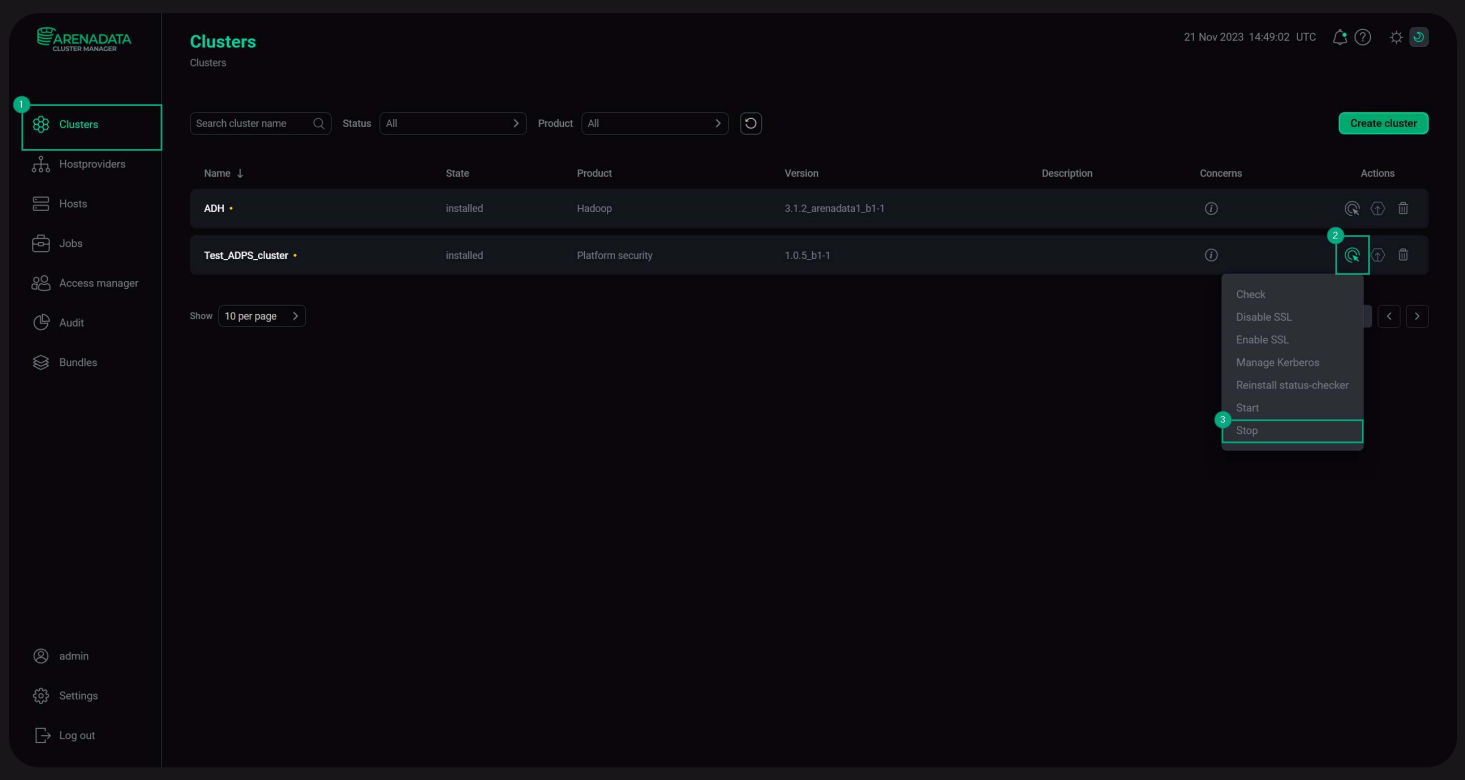
Stop

Sergei Tikhomirov

To Table of Contents

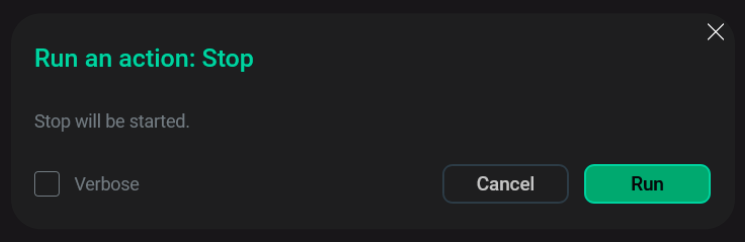
The ***Stop*** action stops all the services installed in the ADPS cluster in the correct order.

To run the action, go to the ***Clusters*** page, find your ADPS cluster, click in the ***Actions*** column, and select the action from the drop-down menu that appears.



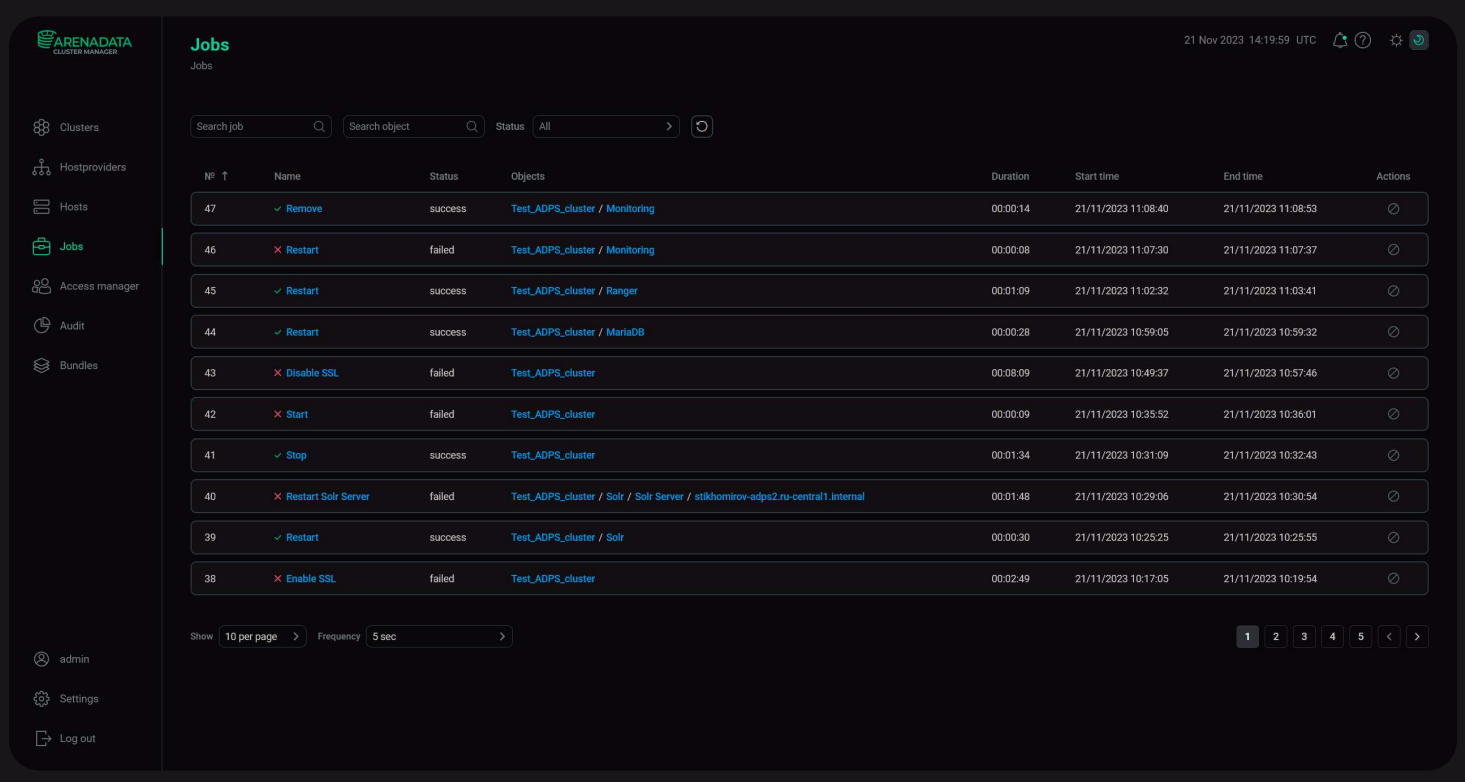
Running an action

When you choose an action, ADCM displays a dialog window to confirm the choice. In this dialog window, you can select the ***Verbose*** checkbox to see additional execution details.



Confirmation window

You can find execution process and result of the action on the ***Jobs*** page. On this page you can view action execution details, including verbose Ansible STDOUT/STDERR outputs.



The Jobs page

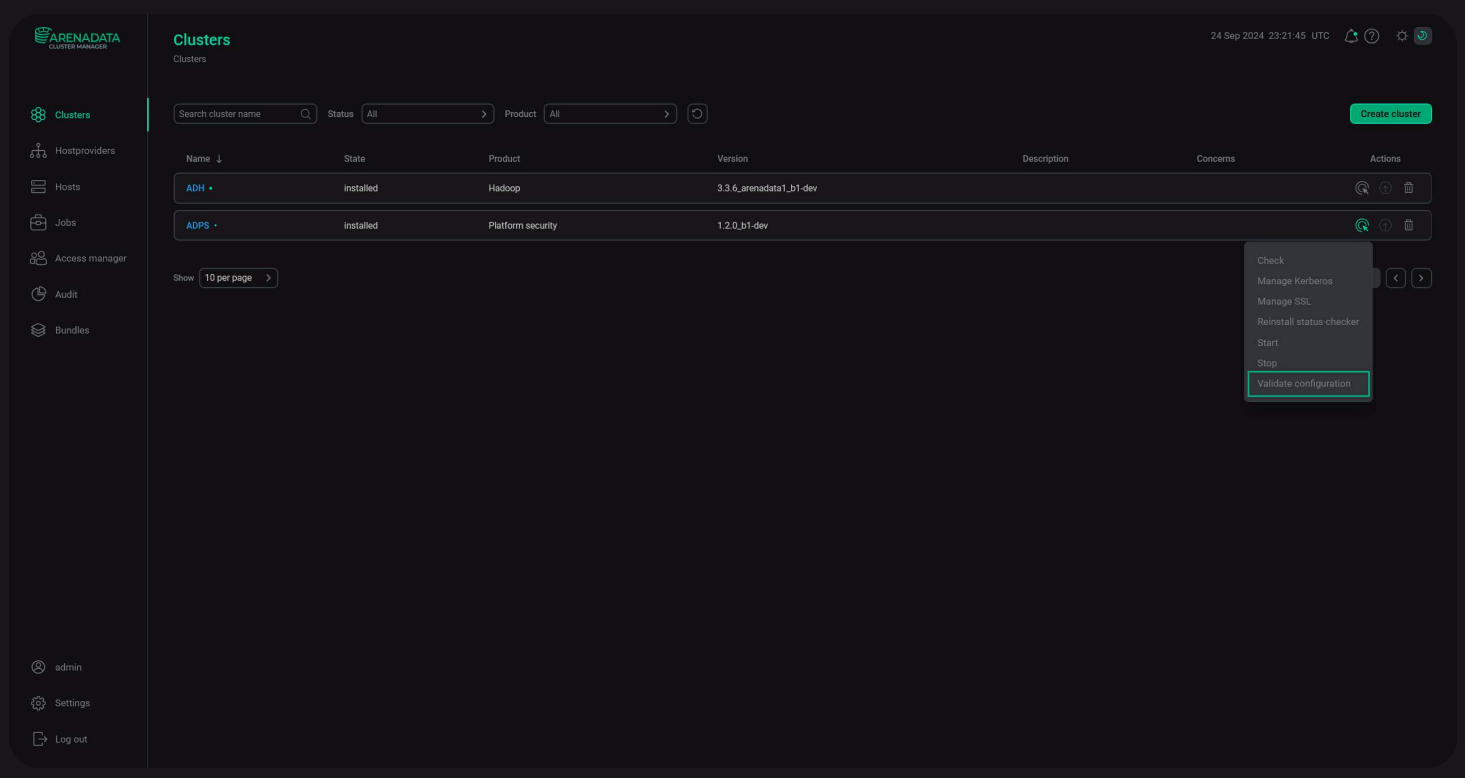
Validate configuration

Sergei Tikhomirov

To Table of Contents

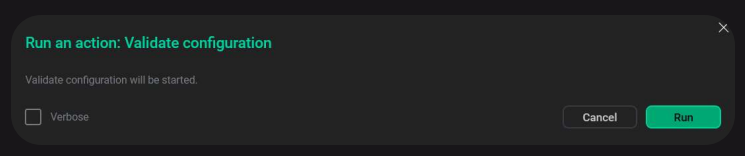
The ***Validate configuration*** action verifies the current port configuration for conflicts before it is applied on hosts. In case there are conflicting ports, you will receive an error.

To run the action, go to the ***Clusters*** page, find your ADPS cluster, click in the ***Actions*** column, and select the action from the drop-down menu that appears.



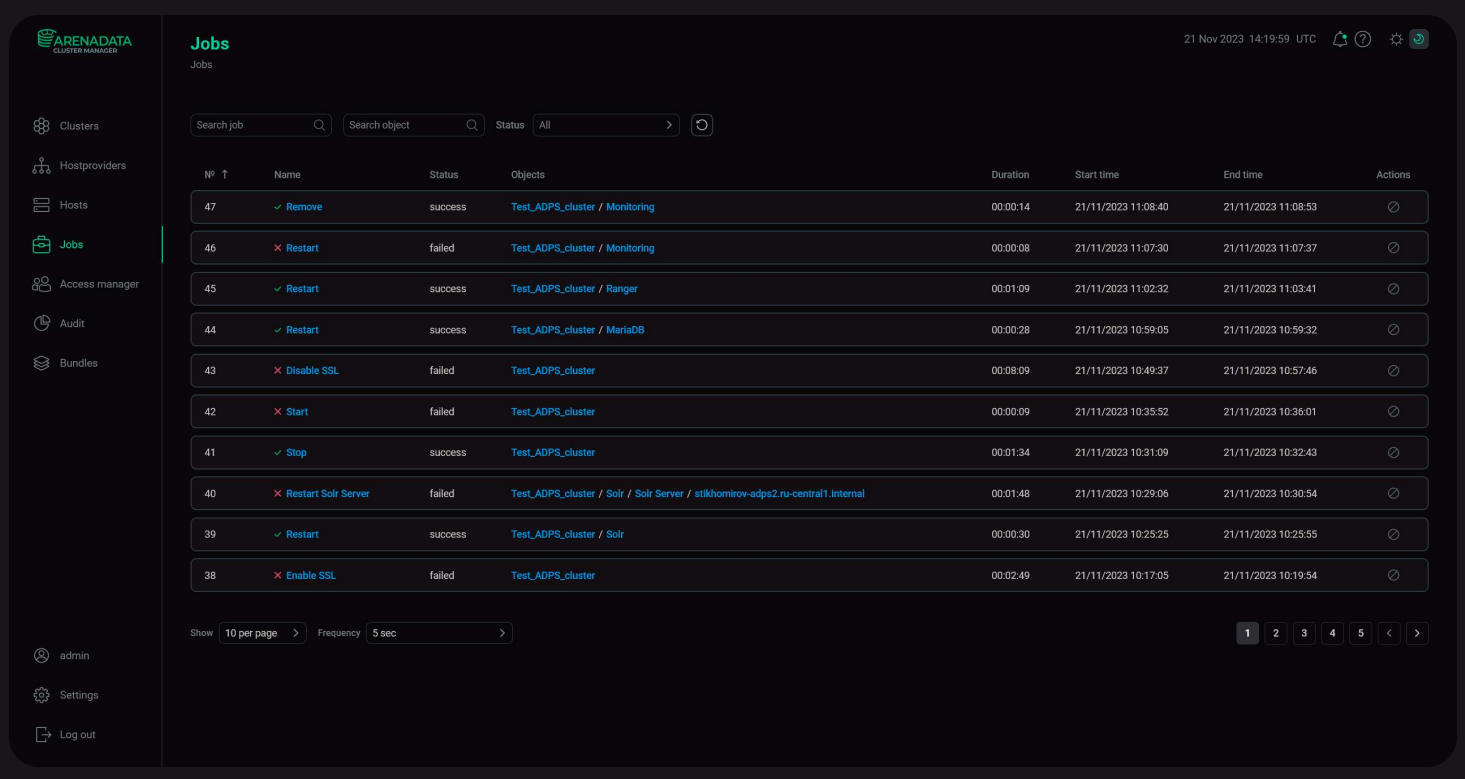
Running an action

When you choose an action, ADCM displays a dialog window to confirm the choice. In this dialog window, you can select the ***Verbose*** checkbox to see additional execution details.



Confirmation window

You can find execution process and result of the action on the ***Jobs*** page. On this page you can view action execution details, including verbose Ansible STDOUT/STDERR outputs.



The Jobs page

Cluster upgrade

Sergei Tikhomirov

Contents

To Table of Contents

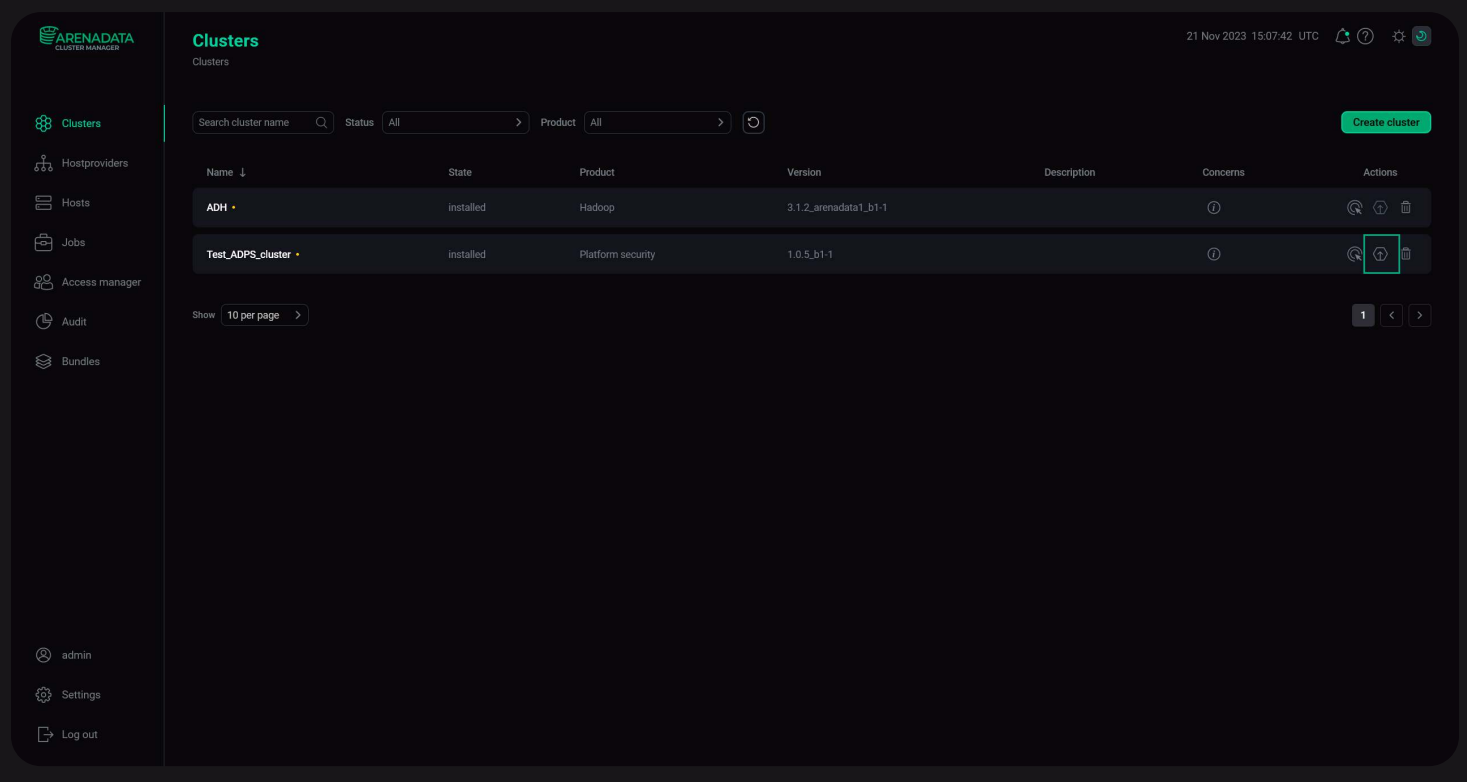
Upgrade process

Upgrade ADPS with SSL enabled

This article provides guidance on upgrading an ADPS cluster to a newer version.

Upgrade process

To upgrade an ADPS cluster, you need to upload the new ADPS bundle to ADCM. Once uploaded, ADCM recognizes the fresh bundle version and the ***Upgrade*** cluster action appears in the ADCM UI.



Upgrade action

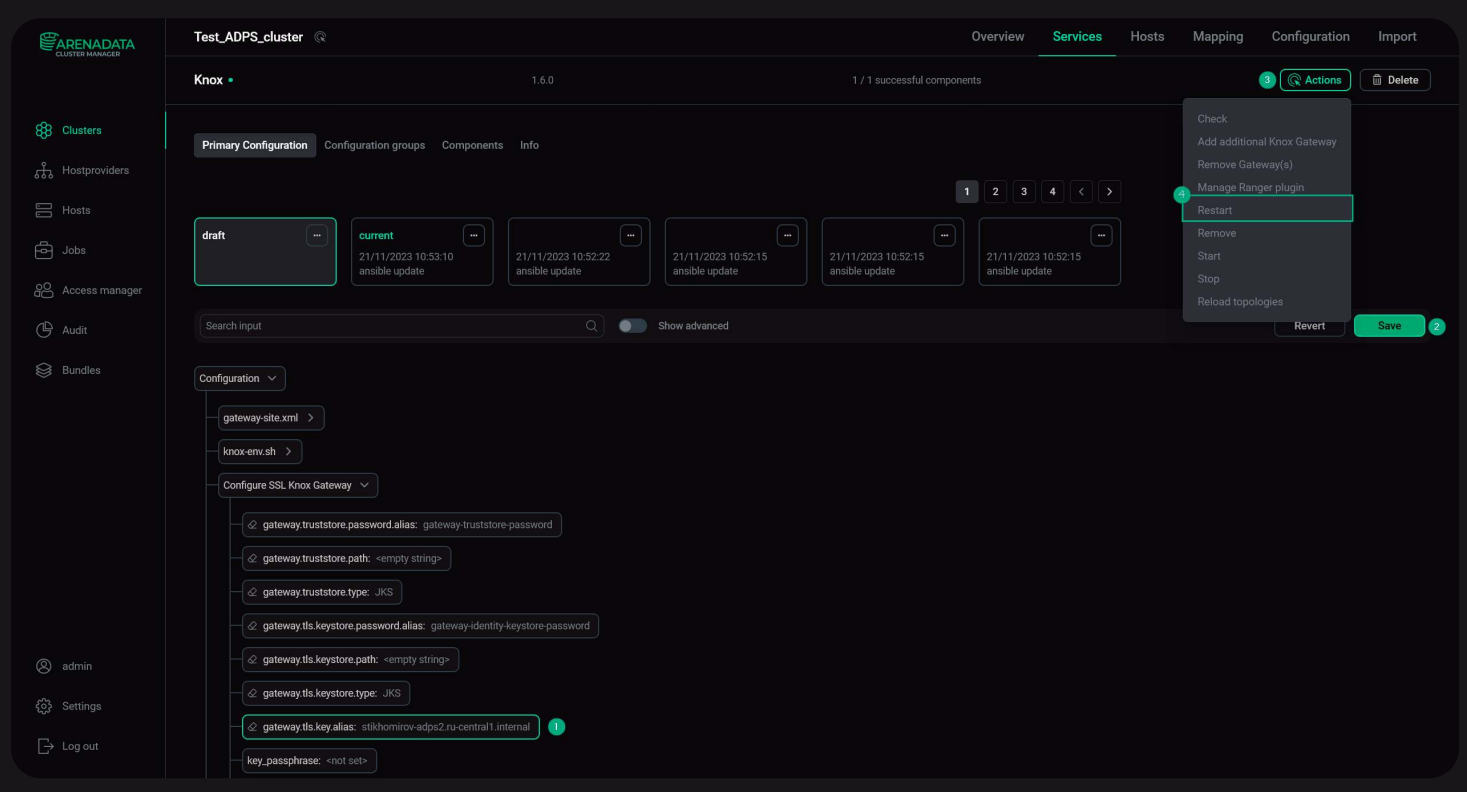
CAUTION

When you upgrade an ADPS cluster, the products must be upgraded in the following order:

1. ADCM
2. ADPS
3. ADS / ADH

Upgrade ADPS with SSL enabled

In the ADPS cluster SSL is enabled via Knox. While upgrading, keep an eye on the `gateway.tls.key.alias` parameter in the Knox service configuration. It should be set to an actual alias of a Knox host (e.g. its FQDN). Make sure to restart Knox after changing and saving the value.



Knox configuration

After that is done you can upgrade your ADPS cluster as usual.

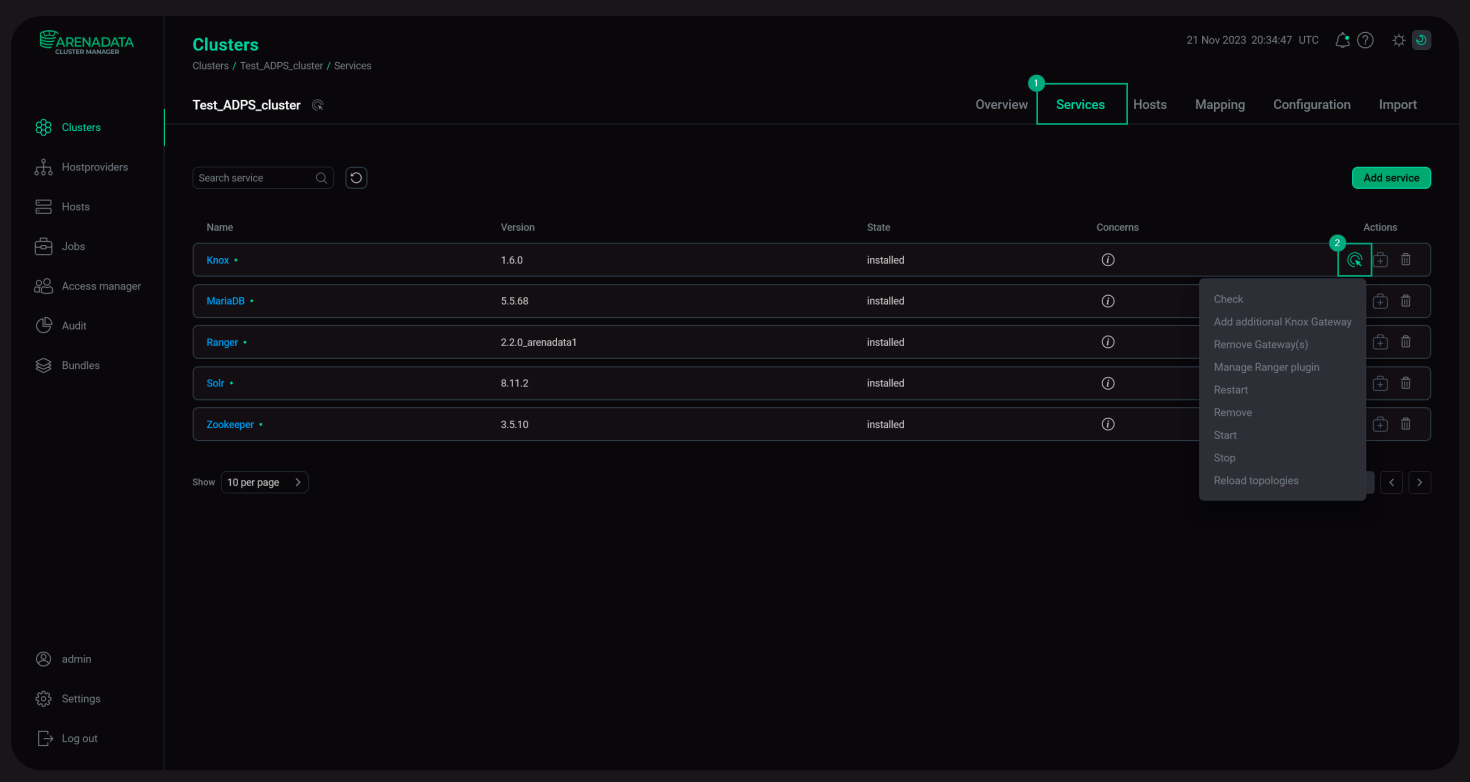
ADPS service actions

Sergei Tikhomirov

To Table of Contents

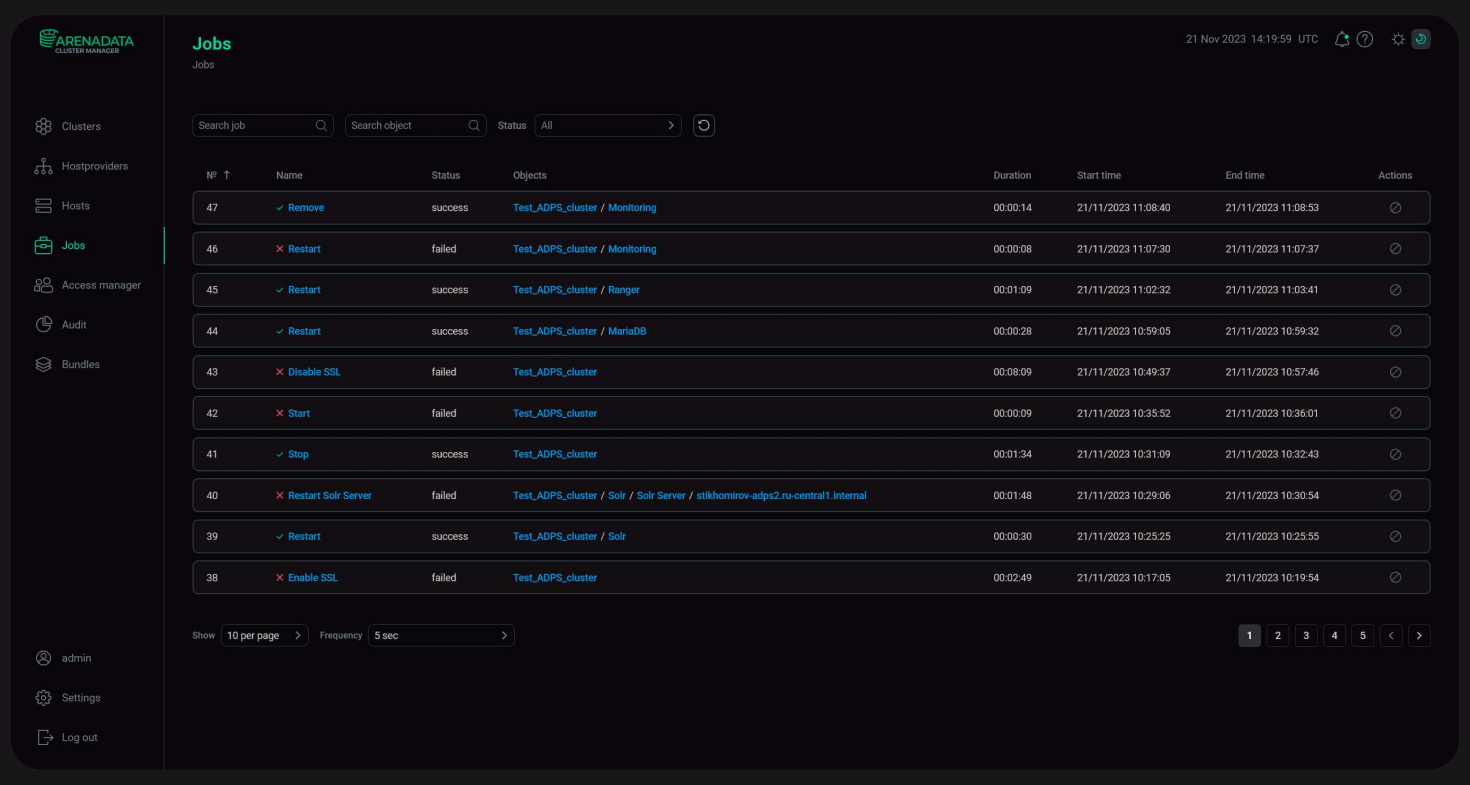
ADCM UI provides a convenient way to interact with your ADPS cluster, hosts, and services via actions. To perform an action on a service, do the following:

1. On the ***Clusters*** page, select the ADPS cluster you want to work with.
2. On the cluster page, click ***Services***.
3. Click the icon in the ***Actions*** column for the service you need. The list of actions available for the given service is displayed in the drop-down menu.



Service actions

To view the progress of an action or the action outcome, use the ***Jobs*** page that is shown below.



Monitor actions

On this page you can view action execution details, including verbose Ansible STDOUT/STDERR outputs. To view the action execution log, click the corresponding action name on the ***Jobs*** page. Action logs are stored for a configurable time period that can be changed in ADCM settings (the Data retention policy section).

TIP

Most of the actions have the ***Verbose*** flag used to generate more detailed action execution logs.

For more information on action for each ADPS service, refer to the following links:

- Knox
- MariaDB
- Ranger
- Solr
- ZooKeeper

Configure Kerberos authentication based on MIT Kerberos via ADCM

Sergei Tikhomirov

Contents

To Table of Contents

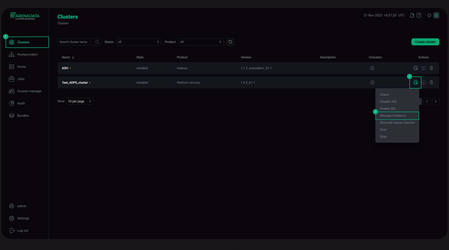
Overview

MIT Kerberos parameters

Overview

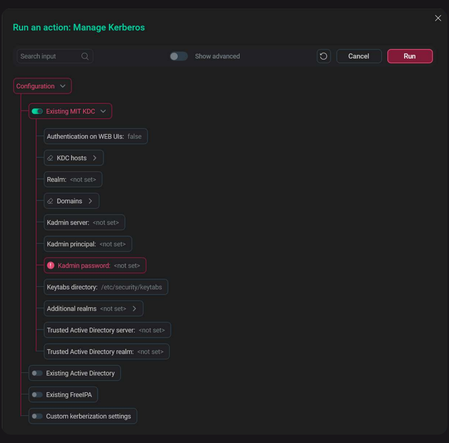
To kerberize a cluster using MIT KDC, follow the steps below:

1. In ADCM web UI, go to the ***Clusters*** page. Select an installed and prepared ADPS cluster, and run the ***Manage Kerberos*** action.



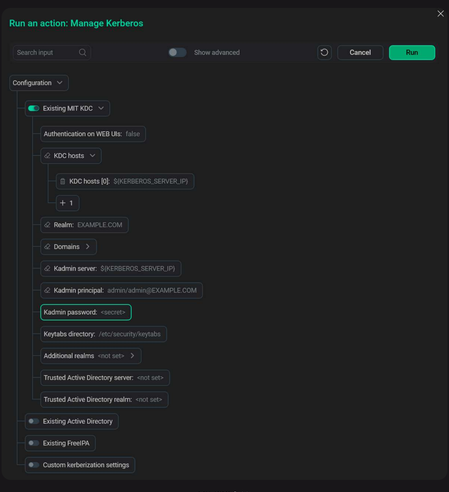
Manage Kerberos

2. In the pop-up window, turn on the ***Existing MIT KDC*** option.



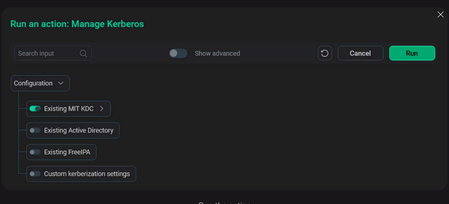
Choose the relevant option

3. Fill in the prepared MIT Kerberos parameters.



MIT KDC fields

4. Click ***Run***, wait for the job to complete and proceed to setting up Kerberos in the cluster.



Run the action

MIT Kerberos parameters

Parameter	Description
Authentication on WEB UIs	Enables Kerberos authentication on Web UIs
KDC hosts	One or more MIT KDC hosts
Realm	A Kerberos realm — a network containing KDC hosts and clients
Domains	Domains associated with hosts
Kadmin server	A host where `kadmin` is running
Kadmin principal	A principal name used to connect via `kadmin`, for example `admin@RU-CENTRAL1.INTERNAL`
Kadmin password	A principal password used to connect via `kadmin`
Keytabs directory	Directory of the keytab file that contains one or several principals along with their keys
Additional realms	Additional Kerberos realms
Trusted Active Directory server	An Active Directory server for one-way cross-realm trust from the MIT Kerberos KDC
Trusted Active Directory realm	An Active Directory realm for one-way cross-realm trust from the MIT Kerberos KDC

Configure Kerberos authentication based on Active Directory via ADCM

Sergei Tikhomirov

Contents

To Table of Contents

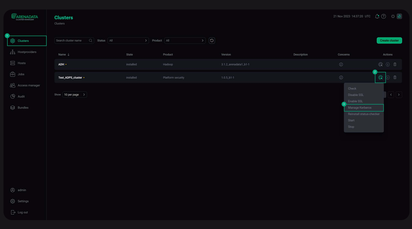
Overview

Active Directory parameters

Overview

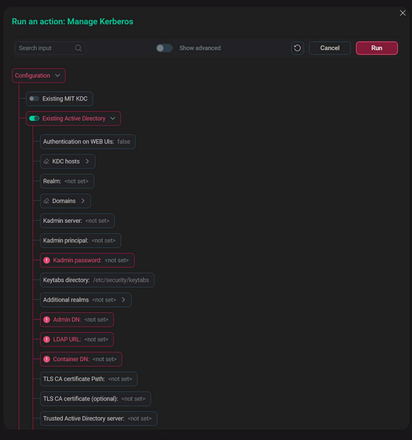
To kerberize a cluster using Active Directory, follow the steps below:

1. In ADCM web UI, go to the ***Clusters*** page. Select an installed and prepared ADPS cluster, and run the ***Manage Kerberos*** action.



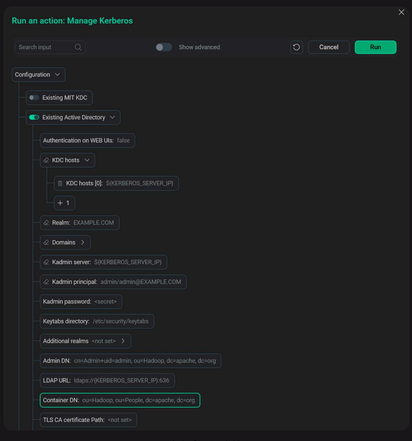
Manage Kerberos

2. In the pop-up window, turn on the ***Existing Active Directory*** option.



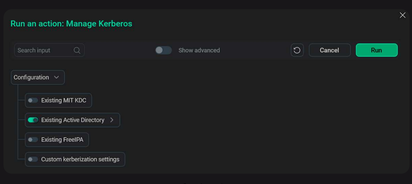
Choose the relevant option

3. Fill in the Active Directory parameters.



Active Directory fields

4. Click ***Run***, wait for the job to complete and proceed to setting up Kerberos in the cluster.



Run the action

Active Directory parameters

Parameter	Description
Authentication on WEB UIs	Enables Kerberos authentication on Web UIs
KDC hosts	One or more KDC hosts
Realm	A Kerberos realm
Domains	Domains associated with hosts
Kadmin server	A host where `kadmin` is running
Kadmin principal	A principal name used to connect via `kadmin`, for example `admin@RU-CENTRAL1.INTERNAL`
Kadmin password	A principal password used to connect via `kadmin`
Keytabs directory	Directory of the keytab file that contains one or several principals along with their keys
Additional realms	Additional Kerberos realms
Admin DN	Full distiguished name of admin user with rights create, modify, delete, pwdchange user accounts in the target Organizational Unit
LDAP URL	LDAP URL consists of `ldap://` or `ldaps://`, hostname or IP address, and port of the AD server
Container DN	Container distinguished name
TLS CA certificate Path	CA certificate path on the host's filesystem. Should be preplaced or will be written down from the `TLS CA certificate` field
TLS CA certificate (optional)	CA certificate to be written on a host using the path from the `TLS CA certificate Path` field
Trusted Active Directory server	An Active Directory server for one-way cross-realm trust from the MIT Kerberos KDC
Trusted Active Directory realm	An Active Directory realm for one-way cross-realm trust from the MIT Kerberos KDC

Configure Kerberos authentication based on FreeIPA via ADCM

Sergei Tikhomirov

Contents

To Table of Contents

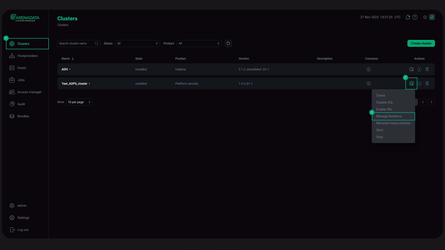
Overview

FreeIPA Kerberos parameters

Overview

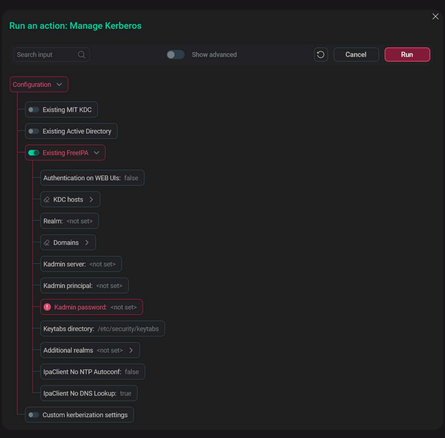
To kerberize a cluster using FreeIPA, follow the steps below:

1. In ADCM web UI, go to the ***Clusters*** page. Select an installed and prepared ADPS cluster, and run the ***Manage Kerberos*** action.



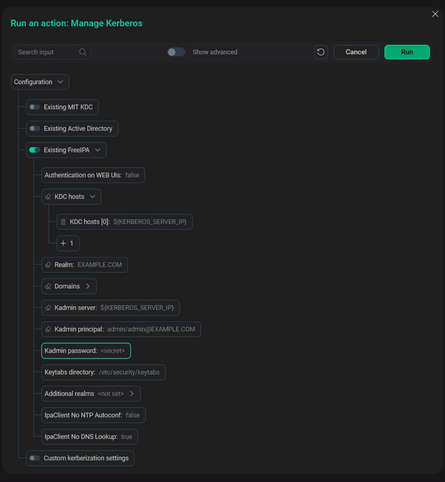
Manage Kerberos

2. In the pop-up window, turn on the ***Existing FreeIPA*** option.



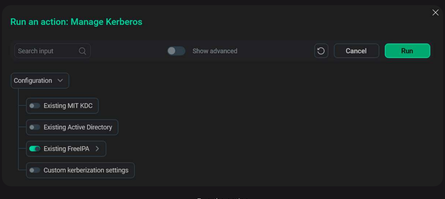
Choose the relevant option

3. Fill in the FreeIPA Kerberos parameters.



FreeIPA Kerberos fields

4. Click ***Run***, wait for the job to complete and proceed to setting up Kerberos in the cluster.



Run the action

FreeIPA Kerberos parameters

The following parameters are required to kerberize your ADPS cluster with FreeIPA. You can get values for these by running `ipa user-find <ipa_admin>` on your FreeIPA server, where `<ipa_admin>` is your IPA Admin user.

Parameter	Description
Authentication on WEB UIs	Enables Kerberos authentication on Web UIs
KDC hosts	One or more KDC hosts with running FreeIPA server(s). Only FQDN is acceptable
Realm	A Kerberos realm to connect to the FreeIPA server
Domains	One or more domains associated with FreeIPA
Kadmin server	A host where `kadmin` is running. Only FQDN is acceptable
Kadmin principal	A principal name used to connect via `kadmin`, for example `admin@RU-CENTRAL1.INTERNAL`
Kadmin password	An IPA Admin password
Keytabs directory	Directory of the keytab file that contains one or several principals along with their keys
Additional realms	Additional Kerberos realms
IpaClient No NTP Autoconf	Disables the NTP configuration during the IPA client installation
IpaClient No DNS Lookup	Disables the DNS lookup for FreeIPA server during the IPA client installation

Configure Kerberos authentication based on Samba via ADCM

Sergei Tikhonov

Contents

To Table of Contents

- Overview
- Kerberization in ADCM
- Samba Kerberos parameters

Overview

Samba acts as an Active Directory domain controller, so its configuration is very similar to that of Active Directory.

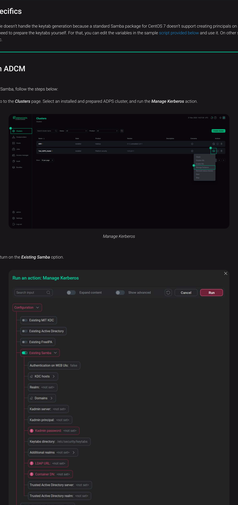
CentOS 7 specifics

On CentOS 7, the bundled *docutils* version is too old for Samba's documentation generation because of a standard Samba package for CentOS 7 doesn't support creating principals on a remote domain controller. Hence, you need to prepare the keytabs yourself. For that, you can edit the variables in the sample script provided below and use it. On other systems, the generation is automatic.

Kerberization in ADCM

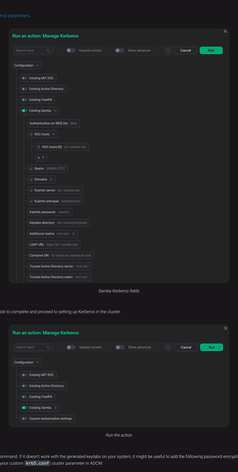
To kerberize a cluster using Samba, follow the steps below:

1. In the ADCM web UI, go to the **Clusters** page. Select an installed and prepared ADPS cluster, and run the **Manage Kerberos** action.



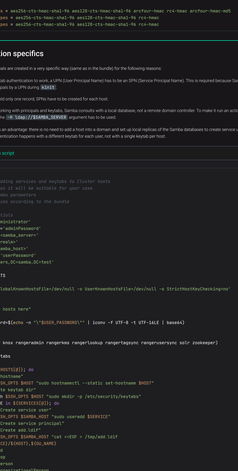
Manage Kerberos

2. In the pop-up window, turn on the *Existing Samba* option.



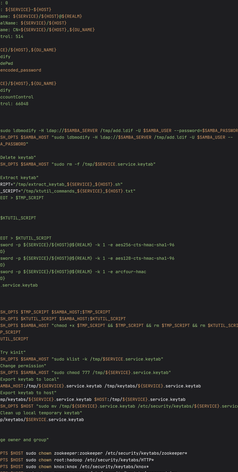
Choose the relevant option

3. Fill in the Samba Kerberos parameters.



Samba Kerberos fields

4. Click **Run**, wait for the job to complete and proceed to setting up Kerberos in the cluster.



Run the action

To check, run the `klist` command. If it doesn't exist with the generated keytabs on your system, it might be useful to add the following password encryption options to the `[libdefaults]` section of your custom `krb5.conf` cluster parameter in ADCM:

```
[libdefaults]
permitted_enctypes = aes128-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
```

Implementation specifics

In the script, the principals are created in a very specific way (same as in the bundle) for the following reasons:

- In order for the keytab authentication to work, a UPN (User Principal Name) has to be an SPN (Service Principal Name). This is required because Samba doesn't allow searching for principals by a UPN alone, `$SERV`.
- Since a UPN can hold only one record, SPNs have to be created for each host.
- By default, when working with principals and keytabs, Samba consults with a local database, not a remote domain controller. To make it run an action on the actual domain controller, the `-H ldap://$SAMBA_SERVER` argument has to be used.

This approach also has an advantage: there is no need to add a host into a domain and set up local replicas of the Samba databases to create service users and principals. Also, the principal authentication happens with a different keytab for each user, not with a single keytab per host.

Keytab generation script

```
#!/bin/bash

# Script for adding services and keytabs to Cluster hosts
# Fix:  NOTE so it will be existed for per name
# FIX3 your Samba parameters
# Strict services according to the bundle

# Samba credentials
SAMBA_USER='Administrator'
SAMBA_PASSWORD='adminPassword'
SAMBA_SERVER='samba_servers'
REALM='samba_realm'
SAMBA_HOST='samba_host'
USER_PASSWORD='userPass'
OU_NAME='CN=Users,DC=samba,DC=test'

declare -A HOSTS

IDM_OPTS="-o UserAtKnownHostsFile=/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)

cache_service
SERVICES='HTTP knox ranger$rack ranger$key ranger$solr ranger$usersync solr zookeeper'

mkdir /tmp/keytabs

for HOST in ${HOSTS[@]}; do
    echo "Set hostname"
        ssh $IDM_OPTS $HOST "sudo hostnamectl --static set-hostname $HOST"
    echo "Create keytab dir"
        ssh $IDM_OPTS $HOST "sudo mkdir -p /etc/security/keytabs"
    for SERVICE in ${SERVICES[@]}; do
        echo "Create service user"
        ssh $IDM_OPTS $SAMBA_HOST "sudo samba-tool user$SERVICE"
        echo "Create service principal"
        echo "Create add ldif"
        ssh $IDM_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 "Update keytab"
        ssh $IDM_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 -k $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}
wst ${SERVICE}.service.keytab
wst *
EOT

        scp $IDM_OPTS $TMP_SCRIPT $SAMBA_HOST:$TMP_SCRIPT
        ssh $IDM_OPTS $KTUTIL_SCRIPT $SAMBA_HOST:$KTUTIL_SCRIPT
        ssh $IDM_OPTS $SAMBA_HOST "chmod +x $TMP_SCRIPT && $TMP_SCRIPT && rm $KTUTIL_SCRIPT"
        rm $TMP_SCRIPT
        rm $KTUTIL_SCRIPT

        echo "Try ktlist"
        ssh $IDM_OPTS $SAMBA_HOST "sudo klist -k /tmp/$SERVICE.service.keytab"
        echo "Change permission"
        ssh $IDM_OPTS $SAMBA_HOST "sudo chmod 777 /tmp/${SERVICE}.service.keytab"
        echo "Export keytab to local"
        ssh $SAMBA_HOST"//${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 $IDM_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 $IDM_OPTS $HOST sudo chown zookeeper:zookeeper /etc/security/keytabs/zookeeper*
    ssh $IDM_OPTS $HOST sudo chown root:hadoop /etc/security/keytabs/HTTP*
    ssh $IDM_OPTS $HOST sudo chown knox:knox /etc/security/keytabs/knox*
    ssh $IDM_OPTS $HOST sudo chown ranger:ranger /etc/security/keytabs/rangeradmin*
    ssh $IDM_OPTS $HOST sudo chown kms:kms /etc/security/keytabs/rangerkms*
    ssh $IDM_OPTS $HOST sudo chown ranger:ranger /etc/security/keytabs/rangerlookup*
    ssh $IDM_OPTS $HOST sudo chown ranger:ranger /etc/security/keytabs/rangertagsync*
    ssh $IDM_OPTS $HOST sudo chown ranger:ranger /etc/security/keytabs/rangerusersync*
    ssh $IDM_OPTS $HOST sudo chown solr:solr /etc/security/keytabs/solr*

    echo "Change permissions"

    ssh $IDM_OPTS $HOST "sudo chmod -R 777 /etc/security/keytabs/"

done
```

Samba Kerberos parameters

Parameter	Description
Authentication on WEB UIs	Enables Kerberos authentication on Web UIs
KDC hosts	One or more domain controller hosts
Realm	A Kerberos realm
Domains	Domains associated with hosts
Kadmin server	A host where `kadmin` is running
Kadmin principal	A principal name used to connect via `kadmin`, for example `admin@MY-CENTRAL5.INTERNAL`
Kadmin password	A principal password used to connect via `kadmin`
Keytabs directory	Directory of the keytab file that contains one or several principals along with their keys
Additional realms	Additional Kerberos realms
LDAP URL	LDAP URL consists of `ldap://`, or `ldaps://`, hostname or IP address, and port of the LDAP server
Container DN	Container distinguished name
Trusted Active Directory server	A trusted DC server
Trusted Active Directory realm	A realm for cross-realm trust

Password encryption

Sergei Tikhomirov

Contents

To Table of Contents

Overview

Ranger

Knox

Solr

Overview

JCEKS is an improved file-based keystore format introduced with the Java Cryptography Extension (JCE). It provides a stronger protection due to Triple DES encryption, which makes it a better alternative to JKS. Even if a store is not protected by a password (in ADCM, password are used by default), an adversary would be unable to extract plaintext data from it.

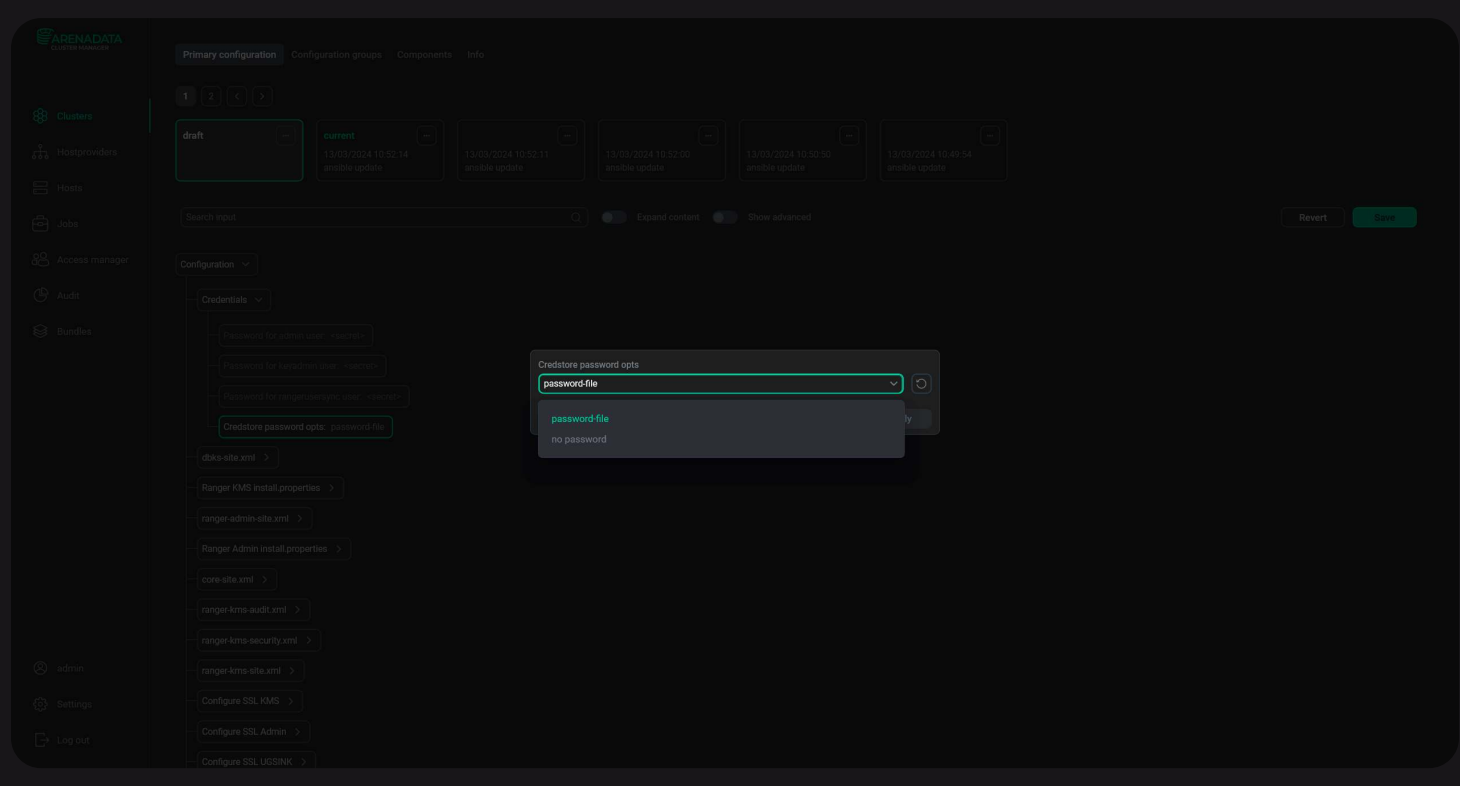
IMPORTANT

Starting with ADPS 1.1.1.b1, all ADPS passwords are stored in JCEKS files (i.e. credential stores) instead of configuration files.

Below, you can find more details about this feature for each service.

Ranger

Ranger has two options to deal with a credstore password: use a password file or use no password. By default, the password file is used. You can change this parameter in ADCM.



Ranger credstore options

A credential store password is autogenerated and stored in a password file defined separately for each service. Its location is */etc/ranger/<service_name>/conf/encrypt_pass*, where `<service_name>` is the name of a Ranger service.

Knox

Passwords for each cluster topology are stored in */usr/lib/knox/data/security/keystores/*. All cluster's JCEKS files are secured by the Knox master password. A password for the Knox Ranger plugin is stored in */etc/knox/conf/ranger-knox.jceks* file which is created if both SSL and the Knox Ranger plugin are enabled. To apply Knox passwords that were changed manually (without the ***Enable SSL*** action or enabling LDAP), it's required to run the ***Update passwords*** action.

The Update Knox passwords action

Solr

Solr has all of its passwords in a JCEKS file, while the password for it is stored in the environment.

NOTE

By default, the encryption is active for all the services except Solr. To turn it on for Solr, activate the ***Credential encryption*** switch in the service configuration and restart Solr with the ***Update jceks*** parameter set to `true`.

Hardware requirements

Sergei Tikhomirov

Contents

To Table of Contents

External database

Ranger admin

Solr

Arenadata Enterprise Tools

Arenadata Monitoring

External database

Sizing for an external PostgreSQL/MySQL database used for Ranger Metadata depends, among other things, on the following factors:

- **Workload**. Estimate the concurrency of queries and the number of users who will access the system simultaneously.
- **Size of metadata**. Try and predict the size of metadata that is going to be stored.
- **Expected growth rate**. Consider your data's historical growth rate and expected future growth. Regularly monitor the database size and scale it as needed.

IMPORTANT

The following system requirements are **minimal**. The target sizing should be calculated based on the customer requirements.

The table below shows some recommendations.

External database minimal requirements

Requirement	Minimum Value
System Architecture	64-bit is recommended
RAM	8 GB
Disk space	100 GB SSD RAID
CPU	4 cores and 2.4 GHz

Ranger admin

Minimum RAM required for operation — 1.5 GB.

Solr

Apache Ranger uses Apache Solr to store and index logs. Apache Solr nodes can be installed on the same hosts as Apache Ranger components.

Ensure that enough resources are allocated for Solr instances. For example, a cluster of 10 nodes consisting of about 15-20 components can generate 1 GB of audit logs in one day.

Arenadata Enterprise Tools

If you use ET along with your product, take into account the following requirements:

- Virtual server.
- CPU: 4 cores or more.
- RAM: 8 GB or more.
- Disk: 50 GB or more.

Arenadata Monitoring

If you use ADM along with your product, take into account the following requirements:

- Virtual server.
- CPU: 8 cores or more.
- RAM: 16 GB or more.
- Disk: 10 GB per monitored host or more.

Network requirements

Eugenia Kuzina

Contents

To Table of Contents

ADPS network port mapping
Client ports
ADCM ports
Arenadata Enterprise Tools ports
Arenadata Monitoring ports

ADPS network port mapping

Service	Component/Server	Port	Protocol	Description
Arenadata PostgreSQL	Arenadata PostgreSQL	5432	TCP	PostgreSQL DB connection port
Knox	Knox Gateway	8443	HTTPS	Gateway port
MariaDB	MariaDB Master Server	3306	TCP	MariaDB connection port
Monitoring	Prometheus Server	11200	HTTP	Prometheus server port
Monitoring	Grafana	11210	HTTP	Grafana web UI port
Monitoring	Node Exporter	11203	HTTP	Node Exporter listen port
OpenBao	OpenBao Server	8201	TCP	Cluster port for server-to-server communication
OpenBao	OpenBao Server	8200	TCP	API port
OpenBao	OpenBao Server	8200	TCP	Listener port
OpenBao	OpenBao Server	8125	TCP	Telemetry port
Ranger	Ranger Admin	6080/6182	HTTP/HTTPS	Ranger Admin web UI and API port
Ranger	Ranger Admin	6085	TCP (local)	Ranger Shutdown port
Ranger	Ranger Admin	9201	HTTP	Prometheus metrics port
Ranger	Ranger KMS	9292/9393	HTTP/HTTPS	Port for Ranger KMS
Ranger	Ranger KMS	7085	TCP (local)	Ranger KMS Shutdown port
Ranger	Ranger KMS	9202	HTTP	Prometheus metrics port
Ranger	Ranger Usersync	5151	TCP/SSL	Port for UNIX Auth service
Ranger	Ranger Usersync	9202	HTTP	Prometheus metrics port
Solr	Solr Server	8983	HTTP	Solr web UI and API port
Solr	Jetty	7983	TCP (local)	Jetty Shutdown port for Solr
Solr	Solr prometheus exporter	9983	HTTP	Solr exporter HTTP port
Zookeeper	Zookeeper Server	2888	TCP	Port for cluster coordinating (internal cluster use only)
Zookeeper	Zookeeper Server	3888	TCP	Port for cluster coordinating (internal cluster use only)
Zookeeper	Zookeeper Server	2181	TCP	Port for clients connection
Zookeeper	Zookeeper Server	5181	HTTP/HTTPS	Zookeeper HTTP REST API interface

Local ports are ports of loopback addresses, with local access only. These ports are marked as `local` in the ***Protocol*** column.

Client ports

Client ports are generated in the local port range. You can determine a specific port from this range using a known service port. Use the lsof command as follows:

```
$ sudo lsof -i -P -n | grep <known port number>
```

For example, the Ranger client runs on a local random port and connects to MariaDB that listens to the `3306` port.

```
$ sudo lsof -i -P -n | grep 3306
```

```
java  21227  ranger  59u  IPv6 43665098  0t0  TCP 10.92.16.192:40388->10.92.16.192:3306 (ESTABLISHED)
```

Examples of client ports in the local port range

Component	Port	Protocol	Description
Curl	45866 or another random port from the local port range	HTTPS	Curl client starts on port 45866 in service checks for sending request to Knox Admin Web UI: 10.92.16.192:45866→10.92.16.192:8443
Ranger	45832 or another random port from the local port range	TCP	The Ranger client connects to MariaDB: 10.92.16.192:45832→10.92.16.192:3306
Solr	43162 or another random port from the local port range	TCP	The Solr client and Solr Server connect to each other: 10.92.16.192:49764→10.92.16.192:8983 10.92.16.192:8983→10.92.16.192:49764

ADCM ports

Port	Protocol	Description
22	TCP	ADCM connection port

Arenadata Enterprise Tools ports

Port	Protocol	Description
443	TCP	Port that is used by Docker Registry
81	TCP	Port that is used to access repositories during offline installation

Arenadata Monitoring ports

Port	Protocol	Description
3000	HTTP	HTTP port to access the Grafana web interface. You can change the default value when installing the monitoring cluster on the configuration page of the Grafana service in the ***ip_and_ports*** section
80	TCP	TCP port to access the Graphite web interface. You can change the default value when installing the monitoring cluster on the configuration page of the Graphite service in the ***ip_and_ports*** section

Software requirements

Sergei Tikhomirov

Contents

To Table of Contents

Operating systems

Java

Databases

Antivirus

Operating systems

The following platforms are supported:

- RedHat 7.9.
- CentOS 7.9.
- ALT Linux 8.4 SP.
- ALT Linux 10 SP.
- Astra Linux 1.7 "Orel" SE. The `LC_TIME` locale should be set to `en_US.utf8`.
- Astra Linux 1.7 "Voronezh".
- Ubuntu 22.04.2 LTS.
- RED OS 7.3 Certified edition (for the Enterprise edition of ADPS, starting with the version 1.2.1.b1)

Java

JDK 8 or later is required (compatible with OpenJDK and Oracle JDK).

Databases

ADPS requires MariaDB 5.5.68 or later. Alternatively, PostgreSQL 9 or later can be used.

The database server can be installed on the same host as Ranger. When you use different hosts for the database server and Ranger, then Ranger service must have access to the database server.

Antivirus

It is recommended to switch off any antivirus software before ADPS installation and during the ADPS operation process. In case of using antivirus, please contact your antivirus software vendor to find out the settings necessary to allow ADPS.

Custom Java settings

Sergei Tikhomirov

Contents

To Table of Contents

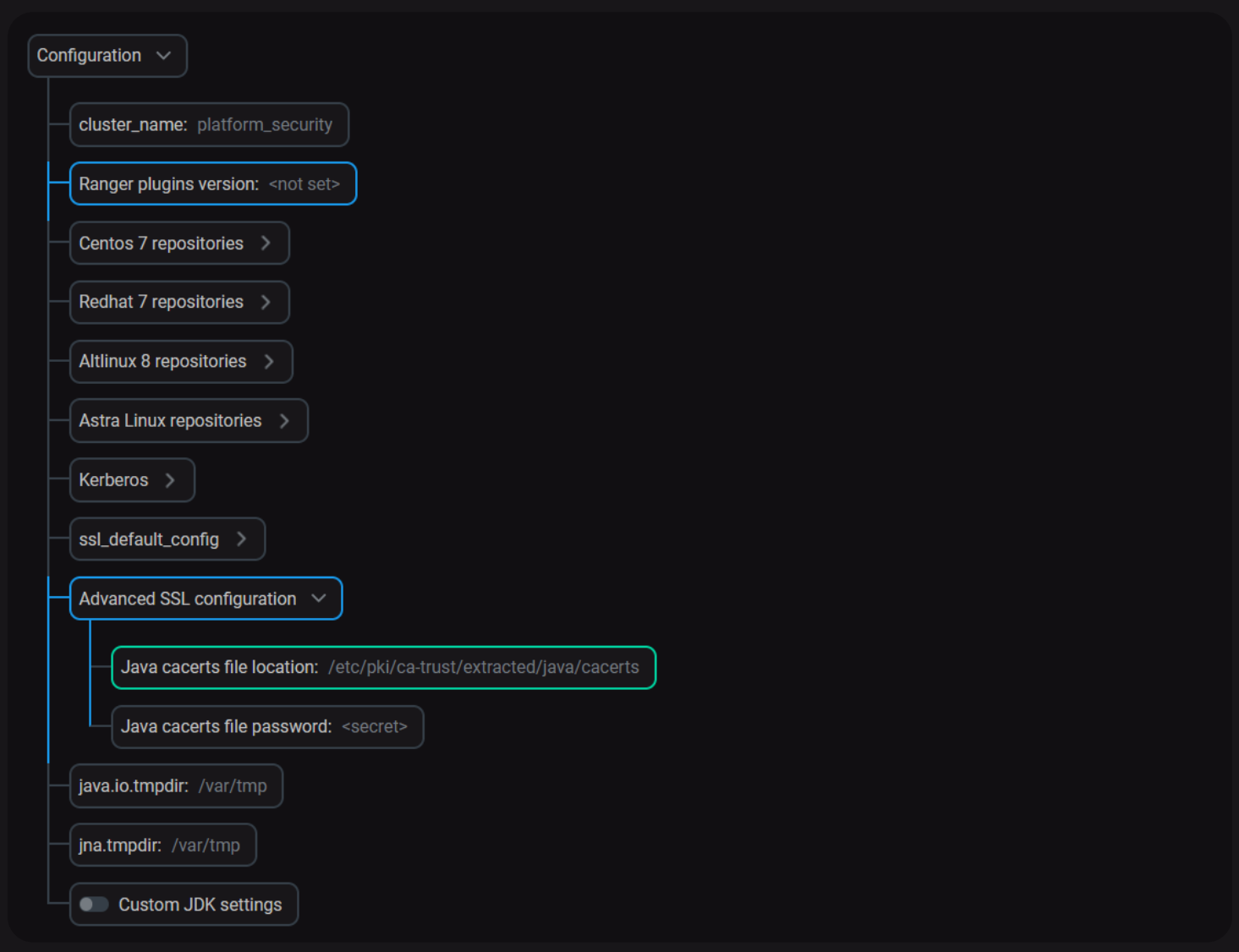
Change the path to cacerts:

NOTE

In order for the ADPS version 1.1.0.b2 components to work correctly with a custom JDK, complete the steps below.

Change the path to cacerts:

1. In ADCM, enter the ADPS cluster configuration and enable the ***Show advanced*** flag.
2. Expand the ***Advanced SSL configuration*** parameter group.



Advanced SSL configuration parameters

3. Change the ***Java cacerts file location*** parameter to the *cacerts* file path that your custom Java uses.

ADPS overview

Sergei Tikhomirov

Contents

To Table of Contents

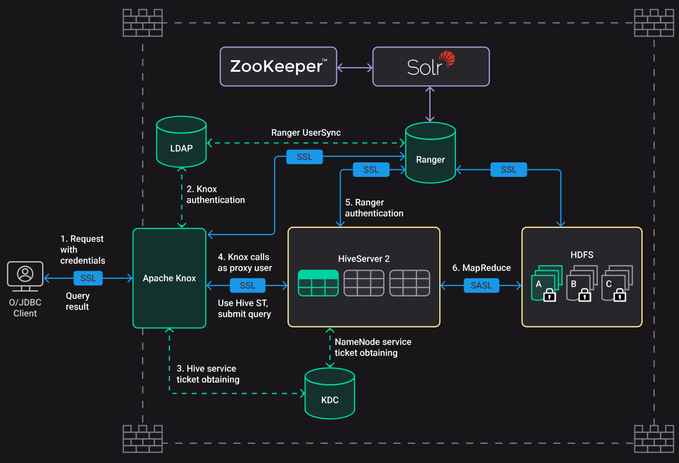
Authentication

Authorization (access control)

Security auditing and monitoring

Data protection

Arenadata Platform Security (ADPS) is a service for centralized management of group security policies in an Apache Hadoop cluster. The diagram below shows how an ADPS cluster can be integrated with an ADH cluster that has Hive.



Possible ADPS integration architecture

ADPS is based on the following services:

- Ranger. An infrastructure for monitoring and managing complex data security on the Hadoop platform. It can use a MariaDB or a PostgreSQL database to store policies and metadata. The Ranger audit logs are stored in Solr and can be viewed in the Ranger Admin UI.
- Knox. ADPS uses Knox as a reverse proxy for a single point of entry and for perimeter security. To enable SSL, you need to use Knox as well.
- **ZooKeeper**. Since Solr doesn't have a master node for node allocation, it needs ZooKeeper for management and service discovery.

Altogether, these services provides a comprehensive approach to safety in the key areas described below.

Authentication

ADPS provides a single authentication point for services and users. The authentication point integrates with the existing enterprise identity and access systems. ADPS supports the following authentication services:

- Kerberos based on MIT/FreeIPA/AD/Samba;
- AD/LDAP;
- local Unix systems.

Authorization (access control)

ADPS provides features that allow system administrators to control access to Hadoop data via the role-based authorization. It is implemented via the Ranger policies that require the Ranger plugins to be used. Ranger policies allow you to create the following authorization models and more:

- fine-grained access control for the data stored in HDFS;
- resource-level access control for YARN;
- service-level access control for the MapReduce operations;
- table/column family-level access control for the HBase data;
- table-level access control for Apache Hive.

Security auditing and monitoring

ADPS allows tracking the system activity using the audit logs. You can also use the perimeter security auditing logs from Knox Gateway. The following processes can be logged:

- access requests;
- data processing operations;
- data changes.

Data protection

ADPS provides mechanisms for the real-time data encryption. ADPS does not require partner solutions for encrypting data at rest, data discovery, and data masking. ADPS supports the following wire encryption methods:

- SSL for ADPS components. This mode is not suitable for all environments and stack services, in particular, there may be problems with the interaction of internal services when using this protocol.
- RPC encryption.
- Data Transfer Protocol encryption.

Knox architecture

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Features

Architecture

- Deployment
- Runtime

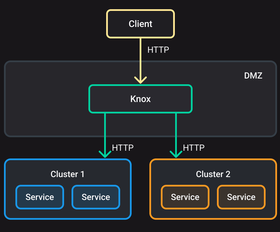
Knox in ADPS

Features

Knox is a gateway service that provides a single point of authentication and access for the Hadoop cluster services.

With Knox, you can achieve the following results:

- Perimeter security for the Hadoop REST APIs:
  - Authentication and token provision (SSO).
  - Ability to integrate the authentication with enterprise and cloud management systems (LDAP).
  - SSL provision for services that do not have that capability.
  - Centralized control via the use of a single gateway that enables auditing and authorization (with Ranger).
- Single URL exposure to aggregate REST APIs of a Hadoop cluster:
  - Control the number of endpoints.
  - Hide the internal Hadoop topology from potential attackers.
- Simplified access due to the encapsulation of services with Kerberos or the use of a single SSL certificate.



Knox placement

Architecture

The gateway is a layer on top of a Jetty JEE server. It has two main extensibility mechanisms: a service and a provider. The service extensibility framework allows you to add support for new HTTP endpoints, while the provider allows adding new features that can be used across various services.

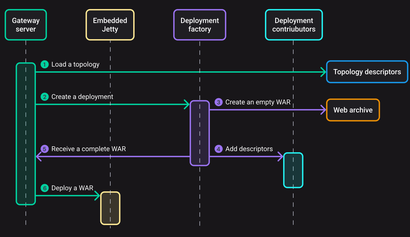
There are two different phases that Knox works in: deployment and runtime. During the deployment, topologies are converted into implementation details that are based on JEE WebArchive (WAR). After that, the runtime phase is responsible for processing the incoming requests using a set of filters that is configured in WAR. More detailed diagrams that go deep into implementation (e.g. differences between the service and provider deployment) can be found in the Knox documentation.

Deployment

The deployment phase exists to convert understandable topology descriptors into optimized executable runtime artifacts. This process can be thought of as compilation of a descriptor into a JEE WAR that gets deployed to an embedded JEE container.

This framework is rather generic, but it has one interesting component called a **contributor** — it's an entity that generates WAR artifacts from topologies. According to its design pattern, each topology is parsed, and an appropriate contributor is selected for each construct within the topology file. After that, the contributor receives the construct and updates the WAR artifacts. The workflow during the deployment phase is presented below:

1. A topology is loaded from the *conf/topologies* directory into an internal structure.
2. A gateway server calls a deployment factory to create a WAR structure.
3. A deployment factory creates a basic WAR structure.
4. Each construct (provider and service) in a topology is visited and the appropriate contributor is invoked. That contributor modifies the WAR structure based on the information passed from a topology file.
5. A complete WAR is returned to the gateway.
6. A WAR is dynamically deployed with an internal container API.

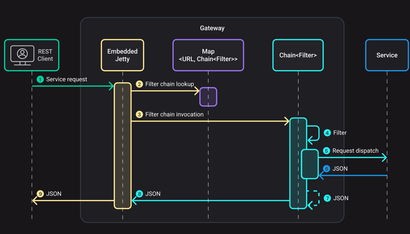


Knox deployment phase

Runtime

The runtime phase is simpler than the deployment phase as it follows well-known JEE models. Below is a high-level diagram that shows how the requests are processed during the runtime. The runtime workflow is described below:

1. A client makes a request for some service. It is received by an embedded JEE container.
2. A filter chain is searched for in a map of URLs and filter chains.
3. A filter chain (which is a filter itself) is invoked.
4. Each filter invokes the filters that follow it in the chain.
5. The last filter in the chain is invoked. Typically this is a special dispatch filter that is responsible for dispatching the request to the ultimate endpoint. Dispatch filters are also responsible for reading the response.
6. A response is received.
7. A response is streamed through various response wrappers added by the applied filters. Such wrappers can edit the response based on their configuration.
8. A response is pulled through the filter response wrapper by the container.
9. A response is returned to the client.

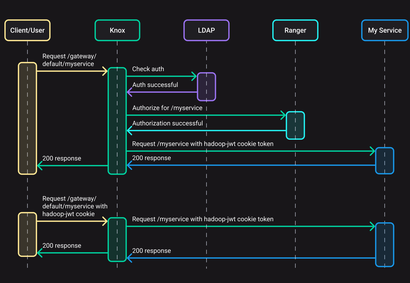


Knox runtime phase

Knox in ADPS

While Kerberos is used for authentication within the Hadoop system, Knox is there to extend the system APIs to new users without having to deal with Kerberos. You can find out the Knox's placement in the ADPS structure in the ADPS overview article.

As Knox can be extended with an LDAP authentication and Ranger authorization, it is possible to create a workflow as demonstrated on the image below. The Ranger part there demonstrates the Knox Ranger plugin.



Knox workflow

Topology descriptors in Knox

Sergei Tikhonievv

AD.PS

Contents

To Table of Contents

Knox topology
- Providers

Topology descriptor

Knox topology

A Knox topology is a cluster-wide configuration file that defines what cluster services are exposed via proxy, how clients can access them, and what security rules are applied to them. All the topologies are stored in the */lib/knox/conf/topologies* directory as XML files.

Commonly, a topology file has the following structure:

```
<topology> 1
    <gateway> 2
        <provider> 3
            <role></role>
            <name></name>
            ...
        </provider>
        ...
    </gateway>
    <service> 4
        <role></role>
        <url></url>
        ...
    </service>
    ...
</topology>
```

1. A root level that contains all the topology settings.
2. A root level for providers that define the Knox Gateway behavior when it receives a request.
3. Provider settings. More information is present in the Providers section.
4. Service configuration. The `<service>` blocks contain information about services that are being proxied through Knox Gateway such as URLs or other service-specific parameters. Knox has many services defined out-of-the-box, but if a service you need is not present, you can define it manually and use it like a predefined one.

Example topology for ADH

Below is an example of a topology file for an ADH cluster that authenticates users via Kerberos and proxies API requests and UI.

```
<topology>
    <generated>true</generated>
    <gateway>
        <provider>
            <role>authentication</role>
            <name>HadoopAuth</name>
            <enabled>true</enabled>
            <param>
                <name>config.prefix</name>
                <value>hadoop.auth.config</value>
            </param>
            <param>
                <name>hadoop.auth.config.cookie.domain</name>
                <value>ru-central1.internal</value>
            </param>
            <param>
                <name>hadoop.auth.config.cookie.path</name>
                <value>/</value>
            </param>
            <param>
                <name>hadoop.auth.config.kerberos.keytab</name>
                <value>/etc/security/keytabs/HTTP.service.keytab</value>
            </param>
            <param>
                <name>hadoop.auth.config.kerberos.name.rules</name>
                <value>DEFAULT</value>
            </param>
            <param>
                <name>hadoop.auth.config.kerberos.principal</name>
                <value>HTTP/stikhomirov-adps.ru-central1.internal@AD.RANGER-TEST</value>
            </param>
            <param>
                <name>hadoop.auth.config.signature.secret</name>
                <value>password</value>
            </param>
            <param>
                <name>hadoop.auth.config.simple.anonymous.allowed</name>
                <value>false</value>
            </param>
            <param>
                <name>hadoop.auth.config.token.validity</name>
                <value>1800</value>
            </param>
            <param>
                <name>hadoop.auth.config.type</name>
                <value>kerberos</value>
            </param>
        </provider>
    </gateway>

    <service>
        <role>HDFSUI</role>
        <url>http://tsn-adh-2.ru-central1.internal:14000/webhdfs/</url>
    </service>

    <service>
        <role>HIVE</role>
        <url>http://tsn-adh-2.ru-central1.internal:10002/</url>
    </service>

    <service>
        <role>NAMENODE</role>
        <url>http://tsn-adh-1.ru-central1.internal:9870/</url>
        <url>http://tsn-adh-3.ru-central1.internal:9870/</url>
    </service>

    <service>
        <role>RESOURCEMANAGER</role>
        <url>http://tsn-adh-2.ru-central1.internal:8088/ws/</url>
    </service>

    <service>
        <role>WEBHBASE</role>
        <url>http://tsn-adh-2.ru-central1.internal:60080/</url>
    </service>

    <service>
        <role>WEBHDFS</role>
        <url>http://tsn-adh-2.ru-central1.internal:14000/</url>
    </service>
</topology>
```

Such topology files can be created manually and pushed to Knox via it's REST API or they can be auto-generated from a topology descriptor that can be configured in the Knox Admin UI.

Providers

Providers define how the requests are processed by Knox Gateway. There are providers available for the following use cases:

- ***Authentication***. Knox supports many authentication methods: LDAP, PAM, Kerberos, SSO, SSO Cookie, JSON Web Tokens, CAS, OAuth, SAML, OpenID Connect, as well as anonymous authentication.
- ***Authorization***. You can restrict certain users, groups, or hosts from accessing services by setting up ACLs.
- ***Identity assertion***. Such providers allow you to impersonate users or adjust their names into an expected format before forwards a request to an actual backend service.
- ***Web application security***. You can opt in to use the following technologies while proxying an incoming request to actual cluster services:
  - Cross-Site Request Forgery
  - Cross-Origin Resource Sharing
  - X-Frame-Options
  - X-Content-Type-Options
  - Strict Transport Security
  - X-XSS-Protection
- ***High availability***. With this provider, you can manually specify the failover options for certain services.
- ***Host mapping***. This provider can be used to dynamically adjust a hostname, port, or scheme of backend service URLs before forwarding requests. This might be useful for the load balancing scenarios.

Topology descriptor

A topology descriptor is a JSON file from which a topology file can be automatically generated. Such files are located in the */lib/knox/conf/descriptors* directory.

A common structure of a descriptor file is presented below.

```
{
  "provider-config-ref": <provider_name>, 1
  "services": [ 2
    {
      "name": <service_name>,
      "urls": [
        <service_urls>
      ],
      ...
    },
    ...
  ]
}
```

1. Name of a provider that is configured in Knox.
2. A list of cluster services to be proxied through Knox Gateway.

Example topology descriptor for ADH

Below is an example of a topology descriptor file for an ADH cluster. A topology from the example above was automatically generated from this provider by Knox.

```
{
  "provider-config-ref": "my-provider",
  "services": [
    {
      "name": "HIVE",
      "urls": [
        "http://tsn-adh-2.ru-central1.internal:10002/"
      ]
    },
    {
      "name": "RESOURCEMANAGER",
      "urls": [
        "http://tsn-adh-2.ru-central1.internal:8088/ws/"
      ]
    },
    {
      "name": "WEBHBASE",
      "urls": [
        "http://tsn-adh-2.ru-central1.internal:60080/"
      ]
    },
    {
      "name": "NAMENODE",
      "urls": [
        "http://tsn-adh-1.ru-central1.internal:9870",
        "http://tsn-adh-3.ru-central1.internal:9870"
      ]
    },
    {
      "name": "HDFSUI",
      "urls": [
        "http://tsn-adh-2.ru-central1.internal:14000/webhdfs/"
      ]
    },
    {
      "name": "WEBHDFS",
      "urls": [
        "http://tsn-adh-2.ru-central1.internal:14000/"
      ]
    }
  ]
}
```

Ranger architecture

Sergei Tikhomirov

Contents

To Table of Contents

Overview

Ranger Admin

Ranger KMS

Ranger User synchronizer

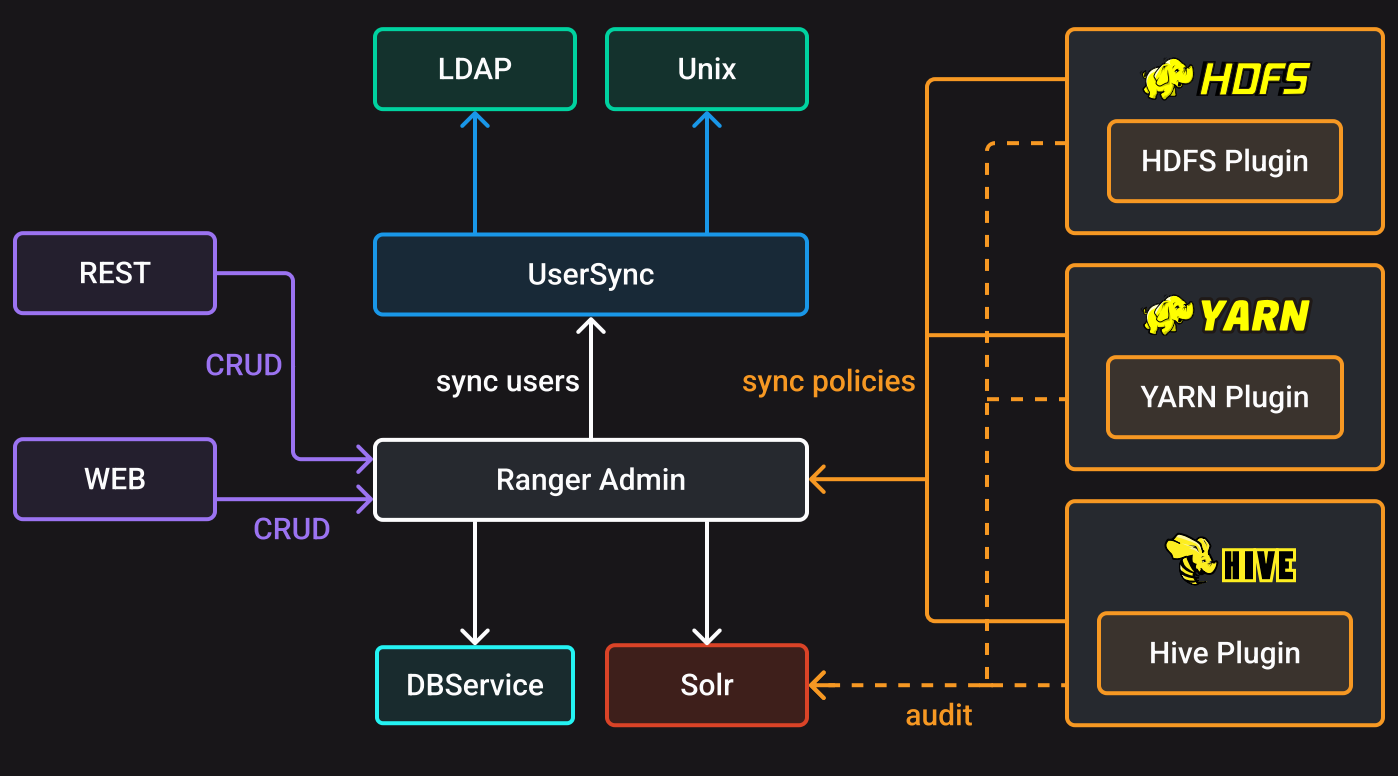
Ranger policy server

Ranger plugins

Overview

Apache Ranger is an open-source security framework that provides centralized policy management for Hadoop and other big data ecosystems.

Ranger is a complex service that consists of three components: Ranger Admin, Ranger KMS, and Ranger User synchronizer.



Ranger architecture

Ranger defines authentication policies for services through plugins and stores them on a policy server.

Ranger Admin

Ranger Admin is a centralizing interface that allows you to manage policies, users, and audit. It has a web interface and also supports REST API. Ranger Admin can work in the high availability mode if there are at least two instances. When a user accesses Ranger Admin's web UI using a browser, a load balancer automatically selects the instance that is the least occupied to provide services. Also, Ranger Admin's metadata can be stored in an external database (PostgreSQL/MySQL).

Ranger Admin supports additional authentication with LDAP/Active Directory.

Ranger KMS

Ranger KMS (Key Management Service) is used for cluster encryption. It is based on Hadoop KMS, but it allows you to store keys in a secure database instead of Java keystore files. The KMS server can be managed through Ranger Admin.

Ranger KMS allows you to create, delete, or update keys using Ranger Admin or via REST API. Ranger KMS is fully compatible with Hadoop KMS's REST API. It is recommended to use Ranger KMS since it provides a secure key storage along with scalability — Ranger KMS supports the high availability mode. Just like Ranger Admin's, Ranger KMS's metadata can be stored in an external database (PostgreSQL/MySQL).

Ranger User synchronizer

Ranger User synchronizer (UserSync) is a feature that allows Ranger to synchronize users/groups from external systems. By default, only users from Unix are imported to Ranger, but you can configure an additional LDAP source.

By default, the incremental synchronization mode is used. In each synchronization period, UserSync updates only new or modified users and user groups. When a user or user group is deleted, UserSync does not synchronize the change to Ranger Admin. To improve performance, UserSync does not synchronize empty user groups. The synchronization is one-way; i.e., if users are deleted from Ranger Admin, they would still remain in the original system.

The synchronization happens periodically, and you can set how often the cycle should happen in the Ranger configuration. The parameter responsible for it is `ranger.usersync.sleeptimeinmillisbetweensynccycle` in the ***ranger-ugsync-site.xml*** parameter group. If the value is unset, the default value is used: `300000` for Unix and `21600000` for LDAP.

Ranger policy server

Ranger policy server allows defining authorization policies for Hadoop applications.

Ranger plugins

Ranger provides access control plugins that replace the original authentication plugins of the Hadoop applications. If a Ranger plugin is enabled for a service, you can set authentication policies for it in Ranger Admin. Ranger plugins periodically update policies from Ranger Admin and cache them in local storage. It is also possible to setup audit logging for plugins.

Policy model in Ranger

Sergei Tikhomirov

AD.PS

Contents

To Table of Contents

Basic concepts

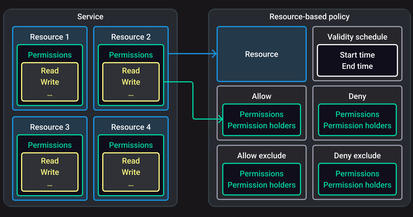
Data manipulation

- Data masking
- Row filter

Access control

- Resource-based access control
- Tag-based access control

Ranger is a convenient framework that manages accesses and permissions for Hadoop and other systems. The Ranger's effectiveness and usability comes from its most prominent feature — policies.



A Ranger policy model

Basic concepts

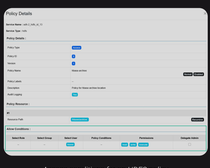
A Ranger policy consists of the following blocks:

- **Resources**. In the context of Ranger, a resource is an entity that requires authorization to be accessed (e.g. a file, database, table, column, etc.). For example, Solr has only one resource — a collection. Ranger allows you to use wildcards, macros, and variables in the resource names, which makes it possible to use a smaller number of policies for a large number of resources.
- **Permissions**. A permission represents an action that can be performed on a resource. For example, reading a file, querying a table, etc.

> **NOTE**
>
> Do not confuse these permissions with the Ranger module permissions.

- **Permission holders**. There are three types of permission holders:
  - User. An individual identity that can be a part of a group. For example, `hdfs` is a user that was created for the HDFS service.
  - Group. A set of users that share a common trait. For example, the `hadoop` group combines several users that were created for the Hadoop services.
  - Role. An authorization level that can be assigned to a user. There are three internal system roles in Ranger Admin: `User` (basic permissions), `Admin` (privileges for all the Ranger modules expect for KMS), and `Auditor` (privileges for all the Ranger modules). You can create a custom role that would combine users, groups, and even other roles.

  Users and groups are typically obtained by UserSync from LDAP/AD/OS. Within a policy, you can set up allow/deny permissions to a resource for several holders at once.
- **Access conditions**. While authorization policies grant access to certain resources, they can also be set up deny access to users/groups/roles on a resource, exclude certain users from the denied/allowed accesses, or deny all accesses to a specific resource other than the ones allowed in the policy. For example, the policy below allows the `hbase` user to read, write, and execute in the */hbase/archive* directory.



Access conditions for an HDFS policy

- **Policy validity schedule**. You can make Ranger policies effective only for a specific duration. You might want to use this feature to issue a delayed access at a certain time or to issue a temporary access for some users/groups/roles.
- **Security zone**. Ranger's security zones allow you to separate resource policies into various zones for convenience. Such separation simplifies administration of policies and also lowers the amount of policies that need to be checked during authorization, since only policies under a particular zone that contains the requested resource are loaded and checked. Also, it allows administrators to set up different policies based on the zones they have admin rights for.
- **Delegated admin**. Additionally to Ranger administrators, certain users/groups/roles can also manage the authorization policies for a certain resource. To achieve that, these permission holders need to be marked as delegated admins when creating a policy for the said resource. For example, the `hdfs` user is a delegated admin for all the HDFS service resources, as shown in the policy below, while the `zeppelin` and `impala` just have the read, write, and execute rights.

An HDFS policy with a delegated admin setting

Data manipulation

Data masking

Ranger allows you to mask certain information for some resources. For example, you can mask a column in Hive and instead of a person's age users would see their age range (e.g. `30-40` instead of `35`). This is done by issuing data masking policies, where you can add users who will see raw information and those who will see the masked version.

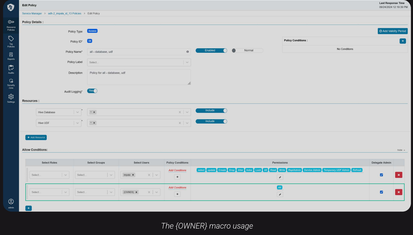
Row filter

While data masking would transform the information, row filter completely removes the information that the user doesn't have access to. This can be used to hide sensitive information or to limit the information by some parameter, e.g. department.

> **NOTE**
>
> Data masking and row filter policies do not grant access to resources. If a user is listed in such a policy but doesn't have access to a referenced resource, the policy will not be applied to them.

Access control

The access control mechanism strictly defines entities that can access certain data. In Ranger, such mechanism is implemented through access conditions that are set during policy creation. The type of access control that Ranger helps you achieve is known as attribute-based access control (ABAC). It is a paradigm, according to which a subject's ability to perform a set of operations is determined by evaluating attributes associated with the subject (user, group), object (resource), requested operations (create, read, etc.), and sometimes environment attributes (time, location, device, etc.). ABAC makes it possible to design authorization policies without prior knowledge of specific resources or users, which helps avoid the need for new policies as new resources or users are introduced. For example, you can allow each user to work with resources they own by using the `{OWNER}` macro in the ***Allow conditions*** section.



The {OWNER} macro usage

There are several subtypes of ABAC that are listed below.

Resource-based access control

The idea of the resource-based control centralizes a single resources and attempts to create a comprehensive access rule around it, allowing it to certain permission holders, while denying it to others. In order to implement the resource-based access control, you need to set the allow/deny conditions for resources (e.g. permission to select a column in some table in a Hive database). This type of access control can get out of hand rather quickly as more and more policies will need to be created as your system grows, but for smaller systems it can be rather clean. See an example of a resource-based policy above.

Tag-based access control

A tag is a special label assigned to a resource, so that special policies would be applied to it. This enables admins to separate some resources based on responsibility (private health information, identity information, credit card information, etc.). To safely implement the resource access, it's worth designing proper data classification and creating tools that scan for sensitive information.

A great benefit of creating authorization policies for classifications instead of resources is that such policies will be applied as classifications are added, removed, or updated. Also, you can create just one tag-based policy to authorize access to a resource for several services like Hive, HBase, Kafka. This feature helps you to reduce the complexity of the authorization policies management.

To achieve the best scalability, it is advised to use the tag-based approach with role-based authorization where possible.

Ranger plugins

Sergei Tikhomirov

Contents

To Table of Contents

- Structure
- Server-side component
  - Service definition file
  - Class implementation
- Application-side component
- Available plugins

A Ranger plugin is a software that enables integrating a service, for which a plugin is available, into Ranger. Such plugins allow Ranger Admin to control access for the service resources via policies. This article describes the Ranger plugin components and how they interact with Ranger. See the schema in the Ranger architecture article which depicts their role in the Ranger workflow.

Structure

A Ranger plugin consists of two components:

- **Server-side component** — a component that exists on the Ranger side.
- **Application-side component** — a component that exists on the service side and invokes the Ranger service to check if the end user has permissions for the requested resource.

You can find a detailed description for each component below.

Server-side component

The server-side component consists of a service configuration that should be defined in a JSON file and an implementation of a class that inherits the `RangerBaseService` class.

Service definition file

The service configuration file has some meta information, but there are also several crucial blocks:

- **Resources**. This block contains information about the service resources (e.g. resource name, type, label, etc.).
- **Access types**. This block's items contain information about which actions are possible to perform on the service resources.
- **Configs**. The information here represents the user input fields.

A typical configuration file structure looks like this:

```
{
    "id": "<service_id>",
    "name": "<service_name>",
    "implClass": "<service_class>",
    "label": "<service_label>",
    "description":"<service_description>",
    "resources": [
        {
            "itemId": 1,
            "name": "<resource_name>",
            "type": "<resource_type>"
        }
    ],
    "accessTypes": [
        {
            "itemId": 1,
            "name": "<action_name>",
            "label": "<action_label>"
        }
    ],
    "configs": [
        {
            "itemId": 1,
            "name": "<field_name>",
        }
    ],
    "enums": [ ],
    "contextEnrichers":[ ],
    "policyConditions":[ ]
}
```

NOTE

Keep in mind, that the purpose of the example above is just to demonstrate the file structure, and many essential fields are omitted.

Class implementation

Consider a child of the `RangerBaseService` class. It expects two functions to be overridden: `validateConfig` and `lookupResource`. The former validates values that are passed to the `configs` section of the service definition file, while the latter enables the autofill of the resource when you are creating a policy in the Ranger Admin UI.

To let the configuration file know which class implements the necessary functionality, the `implClass` field is used.

Application-side component

The application-side component consists of only one thing — a Ranger authorizer. It's a class that communicates with the server-side component and checks if a user has access to the requested resource.

This class's typical workflow is as follows:

1. A requested resource is created.
2. A Ranger request is created using the `RangerAccessRequest` class. Among other things, it includes the resource being requested, the user making the request, and the action that needs to be performed.
3. The request is sent using the `isAccessAllowed` method of the `RangerBasePlugin` class.
4. The response is checked to determine whether the user has access to perform the action of the said resource.

The `RangerBasePlugin` class requires two files to be present in the class path: *ranger-<service-name>-security.xml* and *ranger-<service-name>-audit.xml*.

Available plugins

Currently, there are Ranger plugins available for several ADH, ADPS, and ADS services.

ADH

- HDFS
- HBase
- Hive
- Impala
- Ozone
- Solr
- Spark3
- Trino
- YARN

ADS

- Kafka
- NiFi

ADPS

- Knox
- Solr

OpenBao architecture

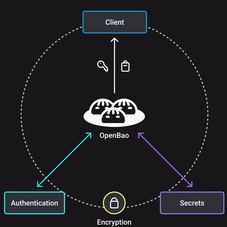
Elena Kostyuchenko

Contents

To Table of Contents

- Main features
- High-level workflow
- Components
- Main concepts
  - Sealing and unsealing
  - Authentication
  - Policy enforcement
  - Token management
  - Leases and expiry
  - Audit logging

OpenBao is a secrets and encryption management system. It provides capabilities for storing, generating, encrypting, revoking, auditing secrets (such as API keys, certificates, passwords) and allows administrators to audit and manage access policies. OpenBao is also able to generate secrets on demand, credentials leasing/renewal and revocation, and encryption-as-a-service.



OpenBao overview

This article describes the internal architecture of OpenBao, its core components, workflows, and how it ensures secrets management.

Main features

OpenBao provides the following features for managing sensitive data:

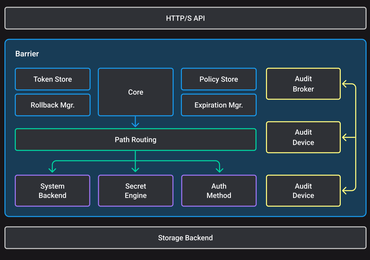
- **Centralized secrets management**. OpenBao acts as a single, secure system of record for secrets such as API keys, certificates, passwords, and tokens.
- **Dynamic secrets**. OpenBao can generate secrets on demand. For example, it can issue temporary database credentials with a defined lease time.
- **Leasing and automatic revocation**. Every secret and token in OpenBao is associated with a lease. When the lease expires, the secret is automatically revoked. Clients may renew leases if policies allow, but once the maximum lifetime is reached, secrets are invalidated. This mechanism enforces credential rotation and minimizes risk.
- **Encryption-as-a-service**. OpenBao provides an interface for performing cryptographic operations such as encryption, decryption, and key derivation without exposing the encryption keys themselves. Applications can securely delegate cryptographic functions to OpenBao instead of handling keys directly.
- **Fine-grained access control**. Access to secrets and operations is controlled by policies. Policies are path-based and follow a deny-by-default model, ensuring that only explicitly authorized actions are permitted.
- **Auditing and logging**. Every request and response passing through OpenBao can be captured by audit devices. Logs contain detailed metadata about client identity, requested paths, and operations.

High-level workflow

OpenBao's request lifecycle typically involves the following stages:

1. **Authentication**
   A client submits credentials or uses an authentication method (e.g. username/password, token, GitHub, LDAP, or public/private key). OpenBao verifies the client's identity.
2. **Validation**
   Optional external verification using trusted sources or identity providers. For example, GitHub, LDAP, AppRole.
3. **Authorization**
   Applying policy rules (ACL-style), which define paths or operations the client is permitted to perform.
4. **Access**
   Once authorized, a client can request secrets, generate dynamic secrets, encrypt/decrypt data, etc. Secrets may have leases (expiry). After lease expiration, secrets are revoked.

Components



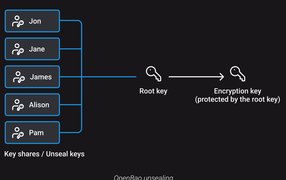
OpenBao architecture

- **Barrier**
  The barrier is the cryptographic encryption layer that protects all sensitive data. Every value written to the underlying storage backend is encrypted before being stored.
- **Storage Backend**
  The storage backend is the physical persistence layer, such as a file system or a cloud storage service. The backend only stores encrypted blobs from the barrier. It stores secrets, tokens, policies, leases, and audit logs but never has access to plaintext data.
- **Server**
  The OpenBao server is the central daemon that coordinates all operations. It handles client requests, applies authentication and authorization logic, interacts with the barrier for encryption/decryption, manages the lifecycle of leases, and routes to the correct Secrets Engine or audit device.
- **Token Store**
  The token store is an internal component that manages tokens issued to clients after successful authentication. Tokens represent a client identity, carry associated policies, and may have leases attached. The token store ensures tokens can be validated and revoked.
- **Policy Store**
  The policy store holds access control policies. Policies are written in ACL style and mapped to paths. They define exactly which operations a client can perform. The server enforces these rules during authorization.
- **Secrets Engine**
  Secrets Engine are pluggable components mounted at paths inside OpenBao. They can generate dynamic secrets (e.g. database credentials), provide encryption services, or store static key/value secrets. Each engine runs inside the server and communicates with the barrier for secure persistence.
- **Audit devices**
  Audit devices record every request and response passing through OpenBao. Multiple audit devices can be configured (e.g. local files, syslog, external log services). Audit logs are themselves protected and must be accessible only to administrators.
- **Expiration Manager**
  The expiration maniger tracks leases associated with tokens and secrets. It ensures that expired secrets are revoked automatically and that renewable secrets can be extended when permitted.
- **Rollback Manager**
  The rollback manager ensures consistency in case of failures. If a Secrets Engine request fails, rollback entries ensure partial state is cleaned up.

Main concepts

Sealing and unsealing

When the server starts, it is in the **sealed** state. The root key must be reconstructed before OpenBao can start working.



OpenBao unsealing

By default, OpenBao uses Shamir's Secret Sharing to split the unseal key into a number of shards. A threshold number of shards is needed to reconstruct the key. This supports distributed key recovery and prevents single-point compromise.

Optionally, external key management systems (KMS) or hardware security modules (HSM) may be used for auto-unsealing or root key protection.

Authentication

Clients authenticate using supported methods (tokens, public/private keys, LDAP, GitHub, etc.). The server validates credentials through the chosen authentication method. After successful authentication and authorization, an OpenBao token is issued and associated with the applicable policies.

Policy enforcement

When a client makes a request, the server checks the policies associated with its token against the path and operation. Policies follow a deny-by-default model, which means that the access is granted only if explicitly allowed.

Token management

Tokens are the main client ID once the client is verified. Tokens may carry leases, be renewable, or be child tokens of another. The token store validates tokens on each request, enforces TTLs, and supports revocation.

Leases and expiry

Secrets and tokens may be issued with leases. Once the lease expires, the expiration manager triggers revocation. Clients may renew their leases before expiry, if permitted by policy. This ensures that stale credentials are rotated automatically.

Audit logging

Every request and response is logged by one or more audit devices. Audit logs include the client identity, operation, path, and metadata, but not credentials.

ADFS releases

Contents

4.0.0
3.0.0
2.0.0
1.2.2
1.2.1
1.2.0
1.1.2
1.1.1
1.1.0
1.0.9
1.0.8
1.0.7
1.0.6
1.0.5
1.0.4
1.0.3
1.0.2
1.0.1
1.0.0

4.0.0

Details

Date: 30 Jul 2021

New services

- Add the Deployment CLI service.

Upgrade

Backend Plugin API 4.0 (unchanged)

- Added capabilities:
  - ADFS.CLI: Implementation supported for ADFS auth-specific key retrieval
  - ADFS.CLI: Implementation for acknowledging login via ADFS
  - ADFS.CLI: Implementation that acknowledges the full token

Port Status

- Added support for Android Linux 11 (Increased)
- Added support for Windows 10 IoT (Increased)

Improvements

- Added the Request Origin header change to the Request, which allows mapping incoming and outgoing requests to their relevant internal http (SQL and Storage Service) requests.

Vulnerabilities

- Added support for Request Origin header validation, ensuring that the client is not sending requests to the wrong server.
- Upgrade users and OS.
- Upgrade packages and OS.
- Upgrade packages and OS.

Bug Fixes

- The **Add Resource** components crashed when it was used by a client.
- Fixed a crash caused by the **Add Resource** component when it was used by a client.

Miscellaneous

- Due to the Base64 service deprecation, it's necessary to remove it from a server with an upgrade.
- The Windows ADFS service version is 4.0.

3.0.0

Details

Date: 30 Jul 2021

Upgrade

Backend Plugin API 3.1 (unchanged)

- Added capabilities:
  - ADFS.CLI: Implementation supported for ADFS auth-specific key retrieval
  - ADFS.CLI: Implementation for acknowledging login via ADFS
  - ADFS.CLI: Implementation that acknowledges the full token

Port Status

- Added support for Android Linux 11 (Increased)
- Added support for Windows 10 IoT (Increased)

Improvements

- Added the Request Origin header change to the Request, which allows mapping incoming and outgoing requests to their relevant internal http (SQL and Storage Service) requests.

Vulnerabilities

- Added support for Request Origin header validation, ensuring that the client is not sending requests to the wrong server.
- Upgrade users and OS.
- Upgrade packages and OS.
- Upgrade packages and OS.

Bug Fixes

- The **Add Resource** components crashed when it was used by a client.
- Fixed a crash caused by the **Add Resource** component when it was used by a client.

Miscellaneous

- Due to the Base64 service deprecation, it's necessary to remove it from a server with an upgrade.
- The Windows ADFS service version is 3.0.

2.0.0

Details

Date: 30 Jul 2021

Upgrade

Backend Plugin API 2.1 (unchanged)

- Added capabilities:
  - ADFS.CLI: Implementation supported for auth-specific key retrieval
  - ADFS.CLI: Implementation for acknowledging login via ADFS
  - ADFS.CLI: Implementation that acknowledges the full token
  - ADFS.CLI: Implementation for acknowledging login via ADFS
  - ADFS.CLI: Implementation that acknowledges the full token

Port Status

- Added support for Android Linux 11 (Increased)
- Added support for Windows 10 IoT (Increased)

Improvements

- Added the Request Origin header change to the Request, which allows mapping incoming and outgoing requests to their relevant internal http (SQL and Storage Service) requests.

Vulnerabilities

- Added support for Request Origin header validation, ensuring that the client is not sending requests to the wrong server.
- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- The **Add Resource** components crashed when it was used by a client.
- Fixed a crash caused by the **Add Resource** component when it was used by a client.

Miscellaneous

- Due to the Base64 service deprecation, it's necessary to remove it from a server with an upgrade.
- The Windows ADFS service version is 2.0.

1.2.2

Details

Date: 30 Jul 2021

Port Status

- Added support for Windows 10 IoT (Increased)

Improvements

- Added the Request Origin header change to the Request, which allows mapping incoming and outgoing requests to their relevant internal http (SQL and Storage Service) requests.

Vulnerabilities

- Added support for Request Origin header validation, ensuring that the client is not sending requests to the wrong server.
- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- The **Add Resource** components crashed when it was used by a client.
- Fixed a crash caused by the **Add Resource** component when it was used by a client.

Miscellaneous

- Due to the Base64 service deprecation, it's necessary to remove it from a server with an upgrade.
- The Windows ADFS service version is 1.2.2.

1.2.1

Details

Date: 30 Jul 2021

Upgrade

- Added a dependency on the ADFS Storage Service.

Port Status

- Added support for Windows 10 IoT (Increased)

Improvements

- Added the Request Origin header change to the Request, which allows mapping incoming and outgoing requests to their relevant internal http (SQL and Storage Service) requests.

Vulnerabilities

- Added support for Request Origin header validation, ensuring that the client is not sending requests to the wrong server.
- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- The **Add Resource** components crashed when it was used by a client.
- Fixed a crash caused by the **Add Resource** component when it was used by a client.

Miscellaneous

- Due to the Base64 service deprecation, it's necessary to remove it from a server with an upgrade.
- The Windows ADFS service version is 1.2.1.

1.2.0

Details

Date: 30 Jul 2021

Upgrade

Backend Plugin API 2.1 (unchanged)

- Added capabilities:
  - ADFS.CLI: Implementation supported for auth-specific key retrieval
  - ADFS.CLI: Implementation for acknowledging login via ADFS
  - ADFS.CLI: Implementation that acknowledges the full token
  - ADFS.CLI: Implementation for acknowledging login via ADFS
  - ADFS.CLI: Implementation that acknowledges the full token

Port Status

- Added support for Windows 10 IoT (Increased)

Improvements

- Added the Request Origin header change to the Request, which allows mapping incoming and outgoing requests to their relevant internal http (SQL and Storage Service) requests.

Vulnerabilities

- Added support for Request Origin header validation, ensuring that the client is not sending requests to the wrong server.
- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- The **Add Resource** components crashed when it was used by a client.
- Fixed a crash caused by the **Add Resource** component when it was used by a client.

Miscellaneous

- Due to the Base64 service deprecation, it's necessary to remove it from a server with an upgrade.
- The Windows ADFS service version is 1.2.0.

1.1.2

Details

Date: 30 Jul 2021

Port Status

- Added support for the Nvidia Jetson nx64 kit

Improvements

- Added support for the **ADFS Storage Client**.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **Add Resource** component when it was used by a client.

Miscellaneous

- Due to the **Base64** service deprecation, it's necessary to remove it from a server with an upgrade.
- The Windows ADFS service version is 1.1.2.

1.1.1

Details

Date: 27 Jul 2021

Port Status

- Added support for the Nvidia Jetson nx64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.
- Added a dependency on the new Storage SDK service.
- Added a dependency on the new Storage SDK service.

1.1.0

Details

Date: 27 Jul 2021

Port Status

- Added support for the Nvidia Jetson nx64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.
- Added a dependency on the new Storage SDK service.
- Added a dependency on the new Storage SDK service.

1.0.9

Details

Date: 27 Jul 2021

Port Status

- Added support for the Nvidia Jetson nx64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.
- Added a dependency on the new Storage SDK service.

1.0.8

Details

Date: 27 Jul 2021

Port Status

- Added support for the Nvidia Jetson nx64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

1.0.7

Details

Date: 27 Jul 2021

Port Status

- Added support for the Nvidia Jetson nx64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

1.0.6

Details

Date: 27 Jul 2021

Port Status

- Added support for the Nvidia Jetson nx64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

1.0.5

Details

Date: 27 Jul 2021

Port Status

- Added support for the Nvidia Jetson nx64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

1.0.4

Details

Date: 27 Jul 2021

Port Status

- Added support for Windows 11 IoT Enterprise x64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

1.0.3

Details

Date: 26 Jul 2021

Port Status

- Added support for Windows 11 IoT Enterprise x64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

1.0.2

Details

Date: 26 Jul 2021

Port Status

- Added support for Windows 11 IoT Enterprise x64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

1.0.1

Details

Date: 26 Jul 2021

Port Status

- Added support for Windows 11 IoT Enterprise x64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

1.0.0

Details

Date: 26 Jul 2021

Port Status

- Added support for Windows 11 IoT Enterprise x64 kit

Improvements

- Added the ability to send a system request to the **ADFS_NODE** service.

Vulnerabilities

- Upgrade users and OS.
- Upgrade packages and OS.

Bug Fixes

- Fixed a crash caused by the **ADFS_NODE** service when it was used by a client.

Miscellaneous

- Due to the **Storage** SDK service's Docker container **Oracle Private Cloud**, which allows you to move a production workload to the SDK, and please add services to the SDK, and please add services to the SDK.

Supported services

Sergei Tikhomirov

To Table of Contents

This article lists services included into the current version of ADPS. The ***Version*** column contains references to the release notes of each component.

Service	Component	Version
Apache Knox	Knox Gateway	2.0.0_arenadata1
Apache Ranger	Ranger Admin	2.6.0_arenadata2
	Ranger KMS	2.6.0_arenadata2
	Ranger UserSync	2.6.0_arenadata2
Apache Solr	Server	8.11.3_arenadata1
Apache ZooKeeper	Server	3.8.4_arenadata1
Arenadata Monitoring	Monitoring	3.2
MariaDB	MariaDB	5.5.68
OpenBao	Server	2.3.2