Arenadata™ Streaming

Версия - v1.3-ENG

Installing Arenadata Streaming

2021 Arenadata™

Оглавление

1 Download distribution platform **3**

2 Installing ADS cluster with ADCM **7**
- 2.1 Benefits . . . 7
- 2.2 Preliminary actions . . . 7
- 2.3 Installation steps . . . 8
  - 2.3.1 Upload ADS bundle . . . 8
  - 2.3.2 Create ADS cluster . . . 8
  - 2.3.3 Manage tools . . . 21

3 Installing ADS cluster with Ambari **37**
- 3.1 Installing Arenadata Streaming with Ambari 2.7.0 . . . 37
  - 3.1.1 Setting up a remote YUM repository . . . 37
  - 3.1.2 Install Ambari Server . . . 37
  - 3.1.3 Ambari Server setup . . . 37
  - 3.1.4 Start Ambari Server . . . 38
- 3.2 Preparing to install Arenadata Streaming MPack . . . 38
  - 3.2.1 Download Arenadata Streaming MPack . . . 38
  - 3.2.2 Import MPack to Ambari stack . . . 39

Currently, there are several ways to install a cluster **ADS**: using **Ambari 2.7.0** as well as **Arenadata Cluster Manager** – ADCM.

Documentation includes:

- The principle of installing the Arenadata Streaming platform using Ambari 2.7.0 and the procedure for preparing to install Arenadata Streaming MPack;
- the principle of installing the Arenadata Streaming platform using ADCM and the procedure for preparing to download and install the Arenadata Streaming bundle.

The instruction can be useful to administrators, programmers, developers and employees of information technology departments who implement and maintain the system.

Important: Contact information support service – e-mail: info@arenadata.io

Глава 1

Download distribution platform

To download the platform distribution **Arenadata Streaming** you need to select the tab “Download” on the site arenadata.io. This takes you to the *App Store* (store.arenadata.io, Pic.1.3.), and opens the on-screen application form to download the product (Pic.1.1.).

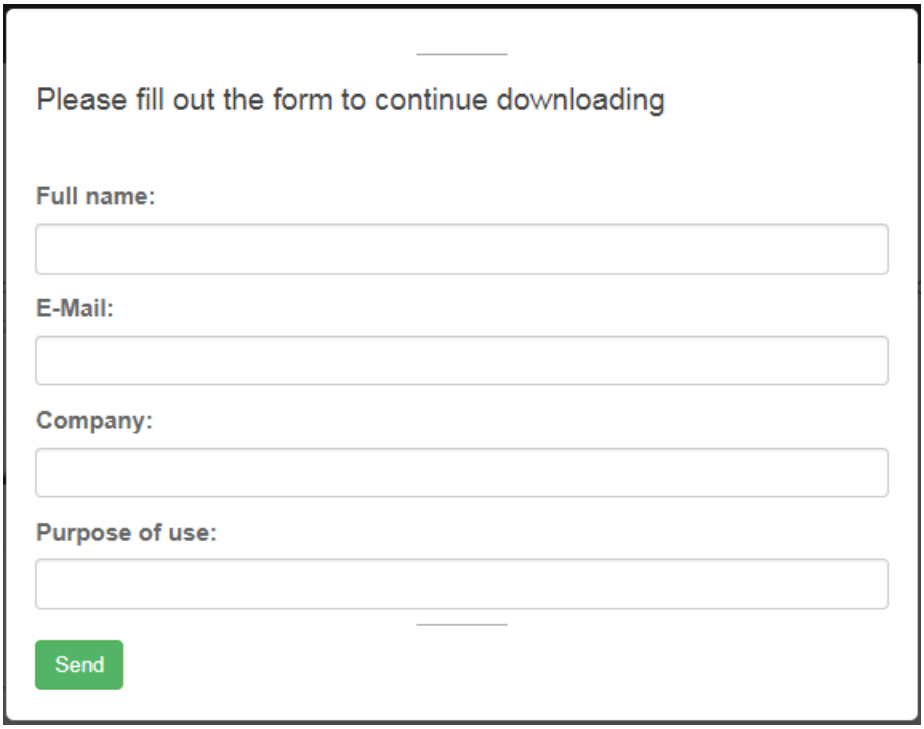


Рис.1.1.: Application for product download

In the screen form of the application should fill in the following fields:

- *Full name* – full name of user;
- *E-Mail* – user email address;
- *Company* – user company name;
- *Purpose of use* – description of the purpose of using the platform.

All fields are required. After entering the data, press the the *Send* button. If the fields are filled in correctly, a corresponding message is displayed (Pic.1.2.).

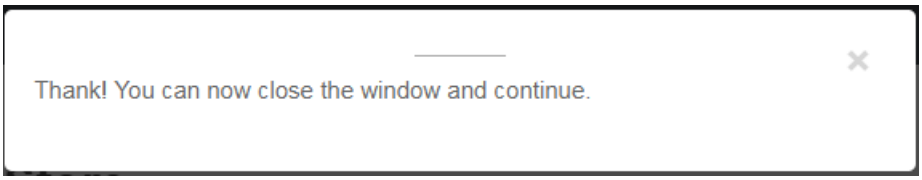


Рис.1.2.: Application accepted

The message box should be closed. This takes you to the page *App Store* with a list of products **Arenadata** (Pic.1.3.).

When hovering the cursor over the product **ARENADATA STREAMING**, a choice of actions is provided:

- *Download* – go to the download page of the platform distributive;
- *Documentation* – go to the online documentation page of the platform.

To download the platform distribution package, select the *Download* item, this will lead to the corresponding page with a brief description of the product, links to distribution components and release information (Pic.1.4.).

In the section *Choose version* you should indicate the version of the platform **ADS** you are interested in, and in the field *Choose OS* – the operating system to be used. At the same time, the release date and release type of the selected version, its number, links to the license of use and Release Notes are displayed in the section *Release information*.

In the section *Links to distribution components* there are links to download components of the selected platform version. After downloading the components, install them, acting in accordance with the subsequent sections of this document.

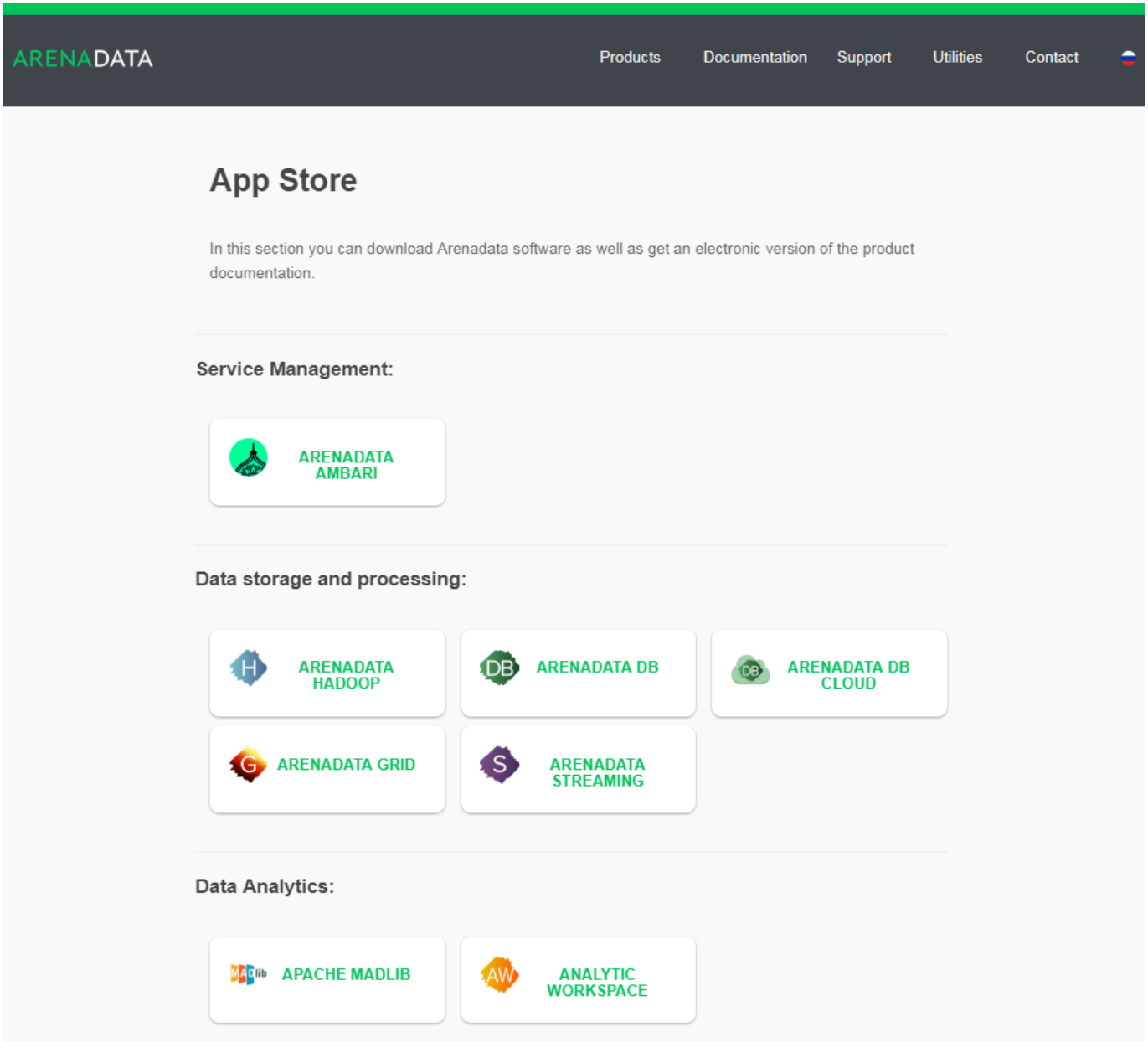


Рис.1.3.: App Store

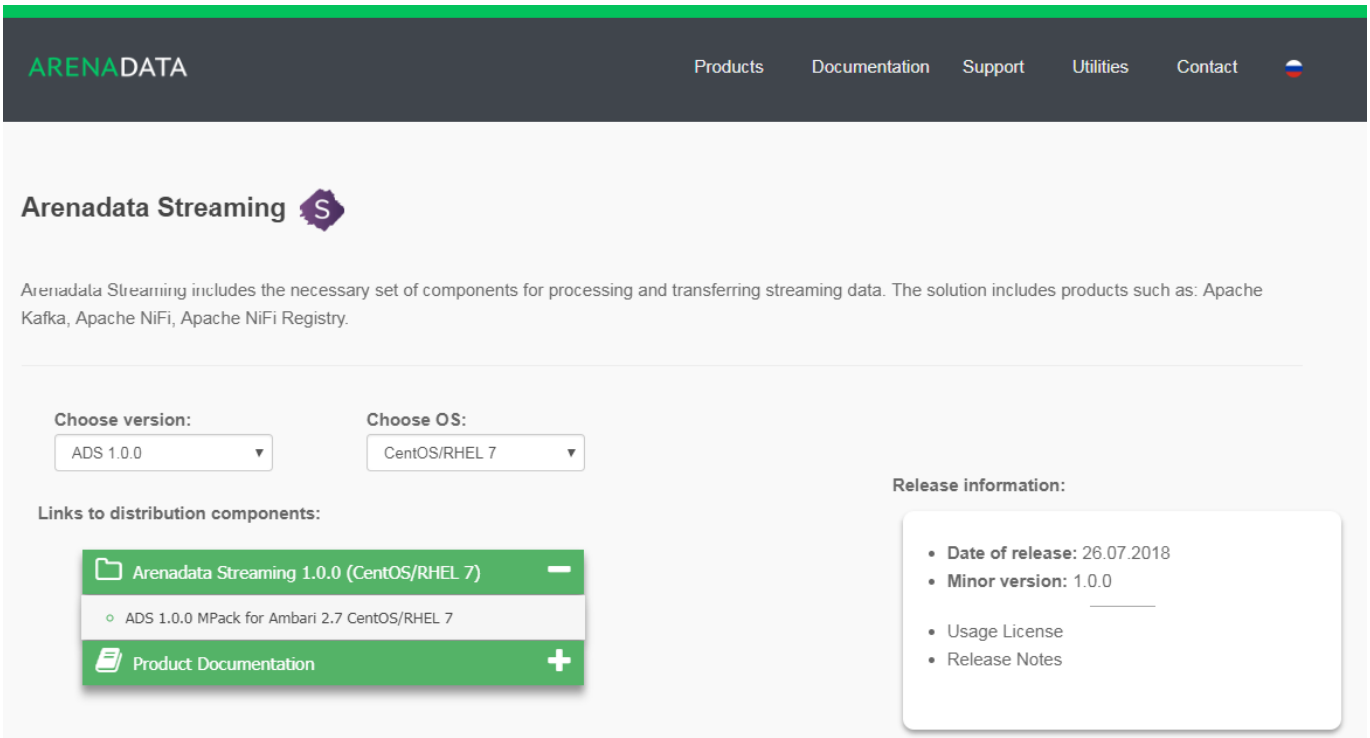


Рис.1.4.: Download Page

Глава 2

Installing ADS cluster with ADCM

The simplest way to install an **ADS** cluster is to use an **Arenadata Cluster Manager** – ADCM.

2.1 Benefits

The advantages of installing the **ADS** cluster through the **ADCM** are:

1. All installation and configuration logic is encapsulated in the ADS bundle:
- User doesn't have to enter a large number of console commands;
- All important settings of OS and database are configured.
2. Single interface for access and configuration management:
- Ability to manage multiple services through a single interface
- Ability to manage multiple clusters through a single interface
3. There is a possibility to use both cloud and physical infrastructure.
4. All ADS rpm packets are taken from the Arenadata repository:
- All rpms have been tested;
- A fixed set of packages is used which simplifies user assistance and compiling bug reports.

2.2 Preliminary actions

To install the **ADS** cluster using **ADCM** you must complete the following preliminary steps:

1. Install ADCM;
2. Create the hosts for the ADS cluster:
- Load required host bundle. The current example uses the *Datafort* bundle;
- Initialize the required number of hosts (in the current example, this step is skipped because the hosts are ready, not the cloud provider);
- Add hosts to ADCM (the current example uses 4 hosts): for Zookeeper (*zk*), for Kafka brokers (*kafka 1* and *kafka2*), for Nifi (*nifi1*);
3. (Optional) Create the monitoring cluster:
- Load monitoring bundle;

- Instantiate and install monitoring cluster.

2.3 Installation steps

2.3.1 Upload ADS bundle

Loading the **ADS** bundle is necessary to create a cluster prototype in the **ADCM** from which it is possible to generate the cluster instances later.

To load the bundle:

1. Open "BUNDLES" tab in ADCM (Pic.2.1.).

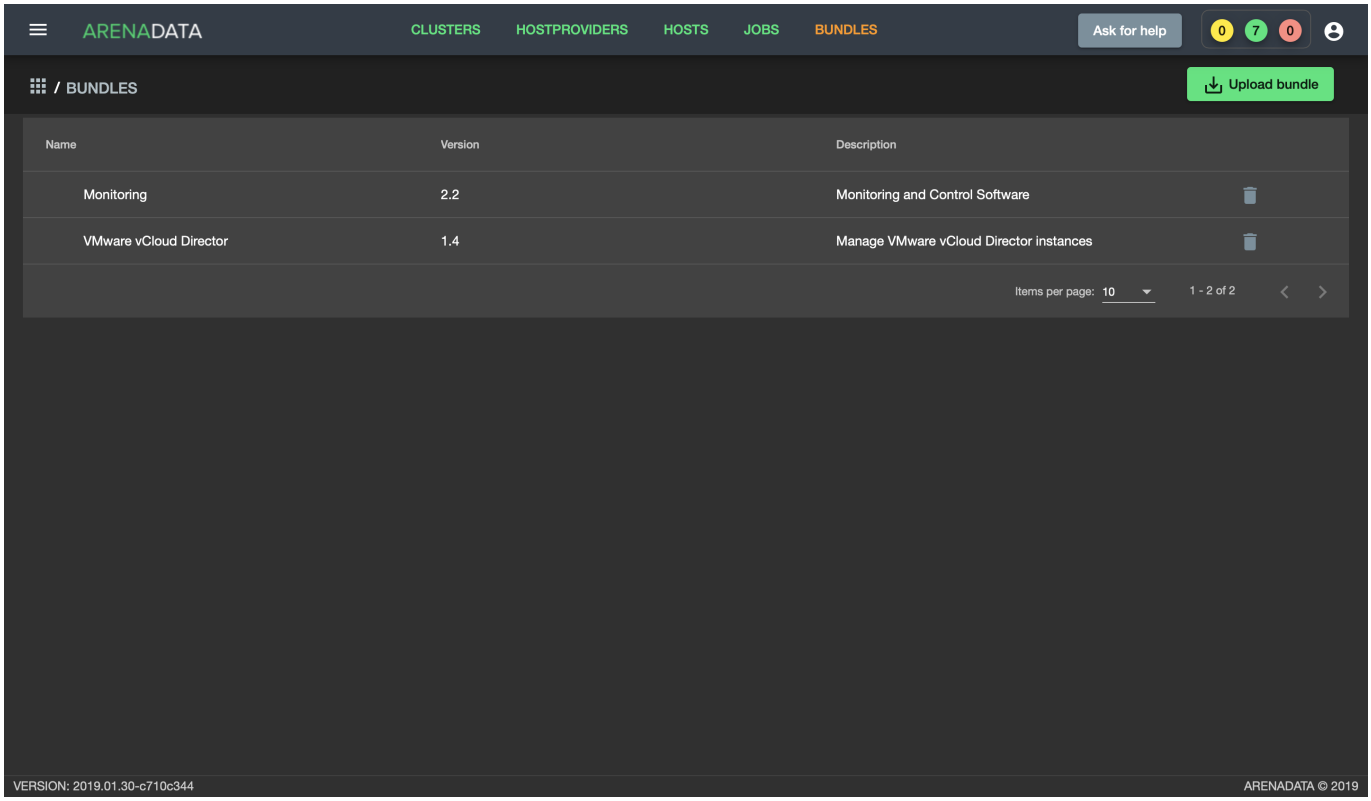


Рис.2.1.: "BUNDLES" tab

2. Press the "Upload bundle" button and select the ADS bundle file in the opened form (Pic.2.2.).
3. As a result the fact of successful loading is displayed in the list of bundles on the "BUNDLES" tab (Pic.2.3.).

2.3.2 Create ADS cluster

After you complete *preliminary actions* and upload ADB bundle, the **ADCM** cluster Manager contains the following list of objects (Pic.2.4.):

- A monitoring cluster prototype and instantiated cluster (a real monitoring program running on a host, ADCM records its results and settings);
- An *ads* cluster prototype for instantiation;
- Pre-prepared hosts, the ADCM database stores records of them and their credentials - ssh keys or passwords.

The following functionality is available:

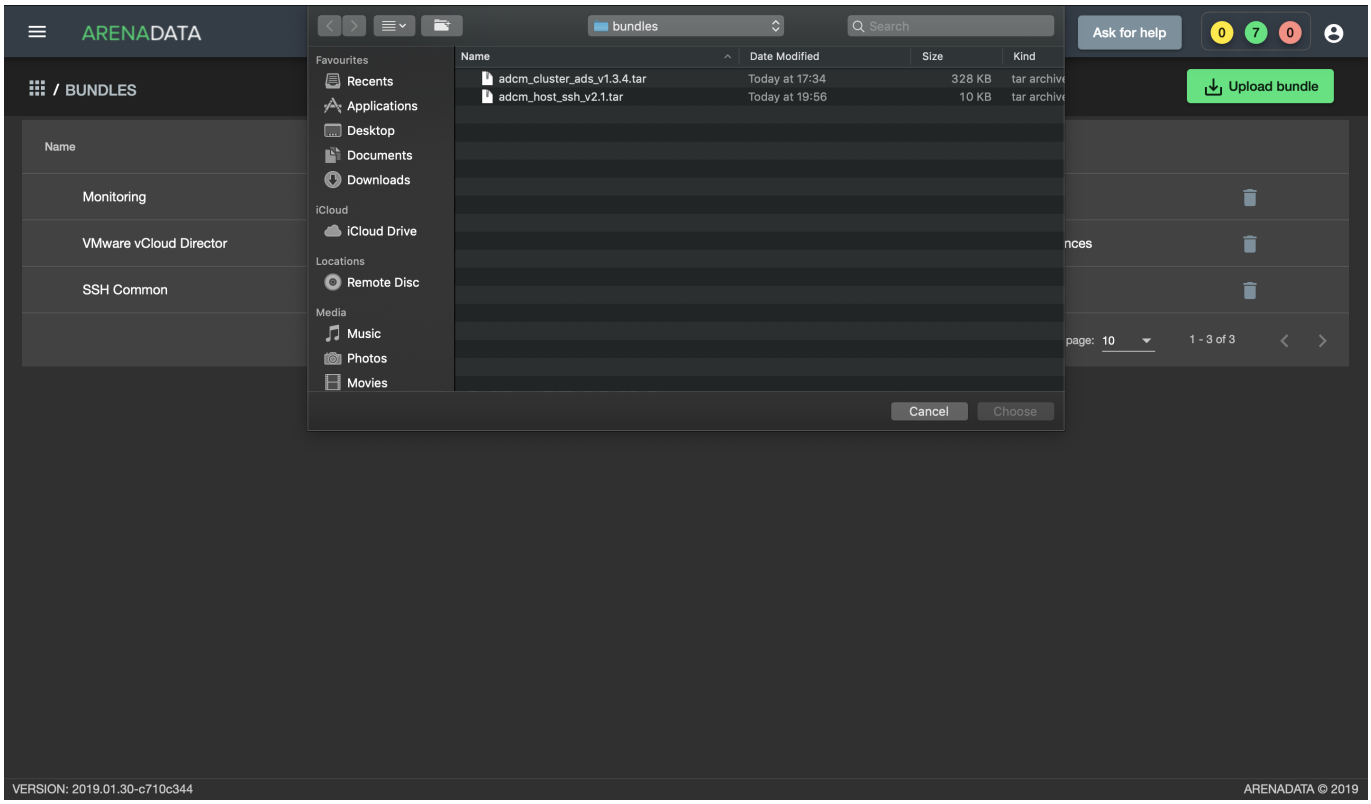


Рис.2.2.: Choose a bundle

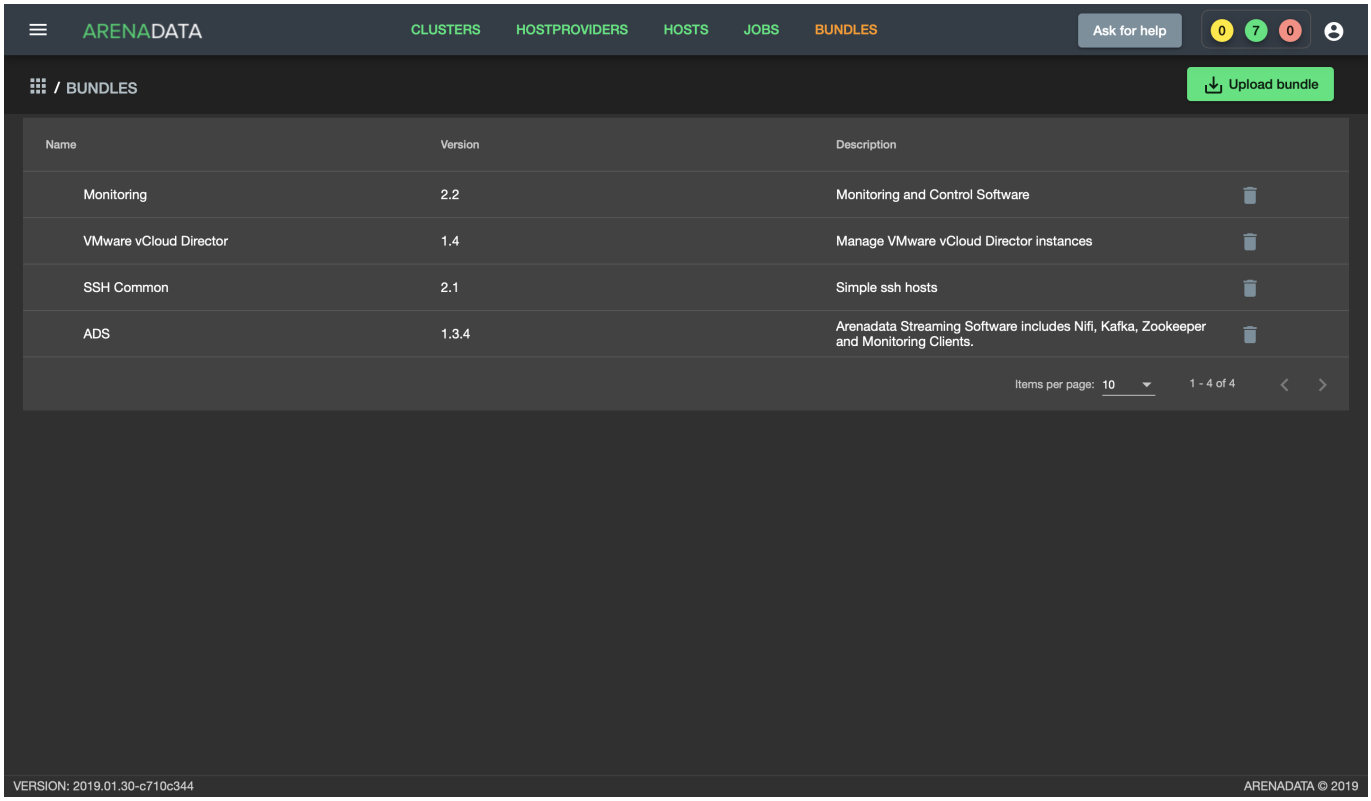


Рис.2.3.: Bundle load success

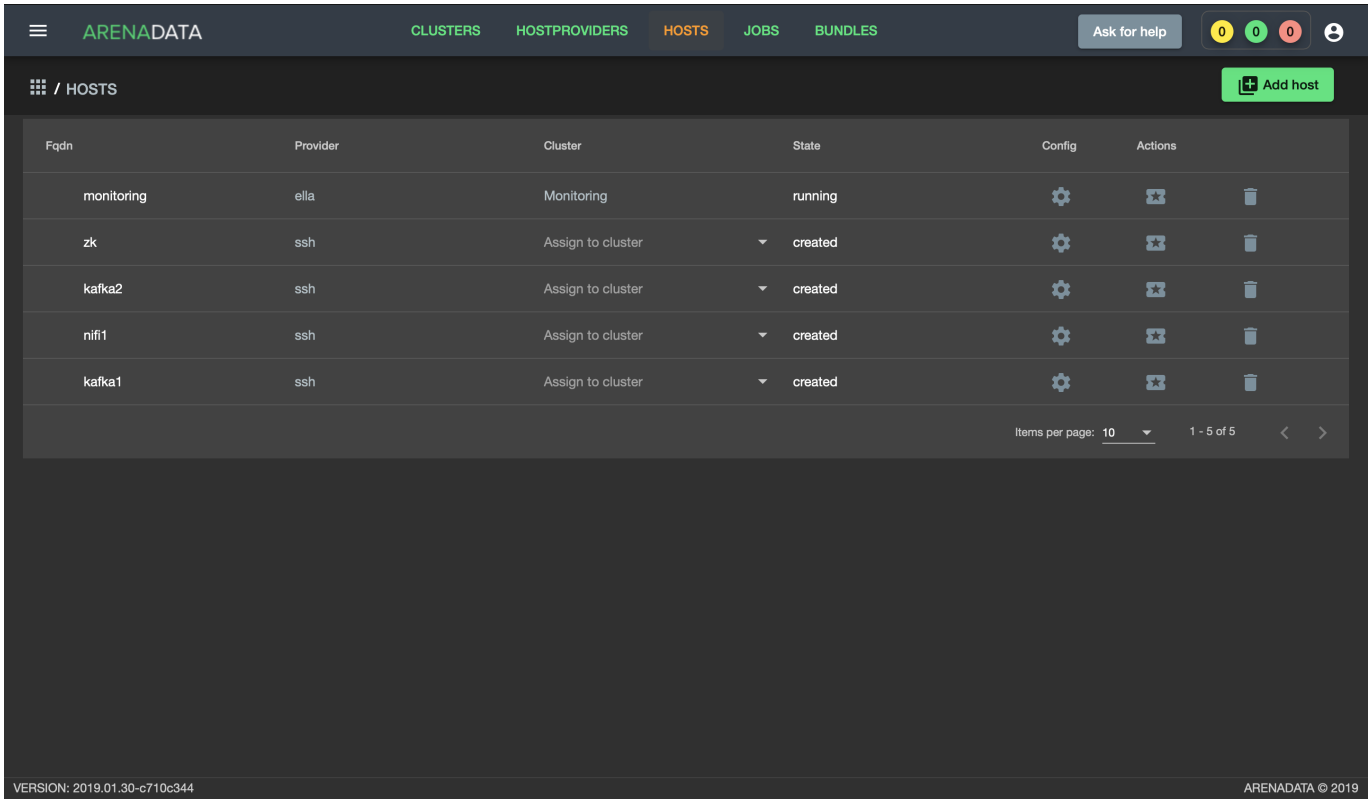


Рис.2.4.: List of objects of ADCM

- *Instantiate cluster*;
- *Configure cluster*;
- *Add services*;
- *Add hosts*;
- *Map service components to hosts*;
- *Ways to install services*;
- *Install Zookeeper service*;
- *Install Kafka service*;
- *Install Nifi service*;
- *Install Monitoring Clients service*.

Instantiate cluster

When cluster instance *ads* is created in web interface of **ADCM** it means only adding data about it to the database **ADCM**, this step doesn't install *ads* on the hosts.

To instantiate cluster:

1. Open the "CLUSTERS" tab in ADCM (Pic.2.5.).
2. Press the "Add cluster" button and in the opened form create an instance of the cluster from the prototype *ads* obtained from the bundle (Pic.2.6.).

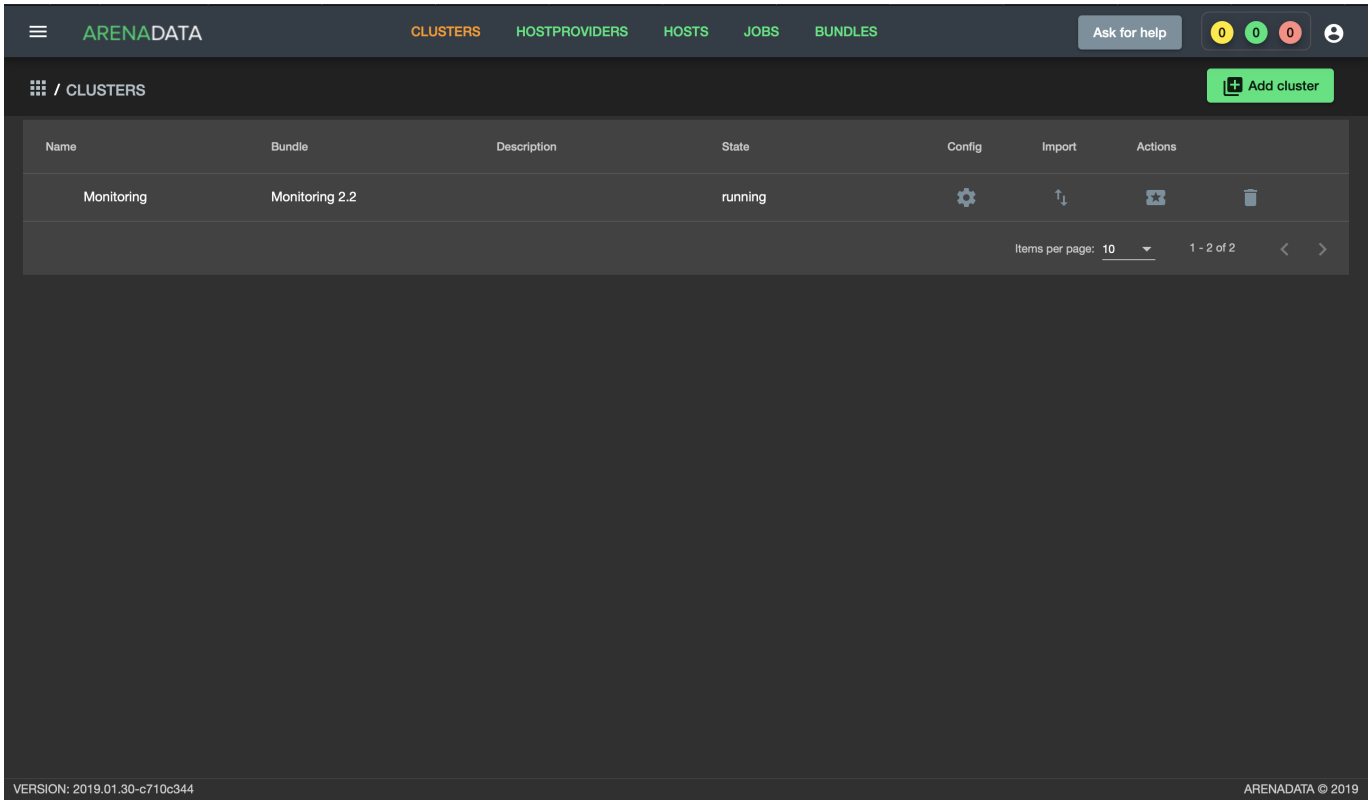


Рис.2.5.: “CLUSTERS” tab

3. As a result of the actions performed, the creation of the cluster instance is displayed in the ADCM database on the “CLUSTERS” tab (Pic.2.7.).

Configure cluster

To go to the settings of the cluster instance *ads*, press the button with the gear icon in the necessary row of the “CLUSTERS” tab (Pic.2.7.) and go to the “Configuration” menu section. This opens the configuration window for the selected instance (Pic.2.8.).

In the block “ADCM” it is possible to configure addresses:

- *Host* (or fqdn and IP address);
- *Port*.

In the “repos” settings block, you specify the packages required for *ads* installation from various yum repositories, and in each of the parameters you can change the default url to the required one:

- *ads*;
- *monitoring*;
- *epel*.

Any cluster’s service component is able to send keep alive status information to the **ADCM** process in the docker container. Sometimes **ADCM** can be behind NAT and there is no evident way to automatically tell the address of **ADCM** to the service components located on cluster’s hosts. That is why it should be configured manually. During the **ADB** installation the **ADCM** address would be used to fill component’s keep alive status configuration.

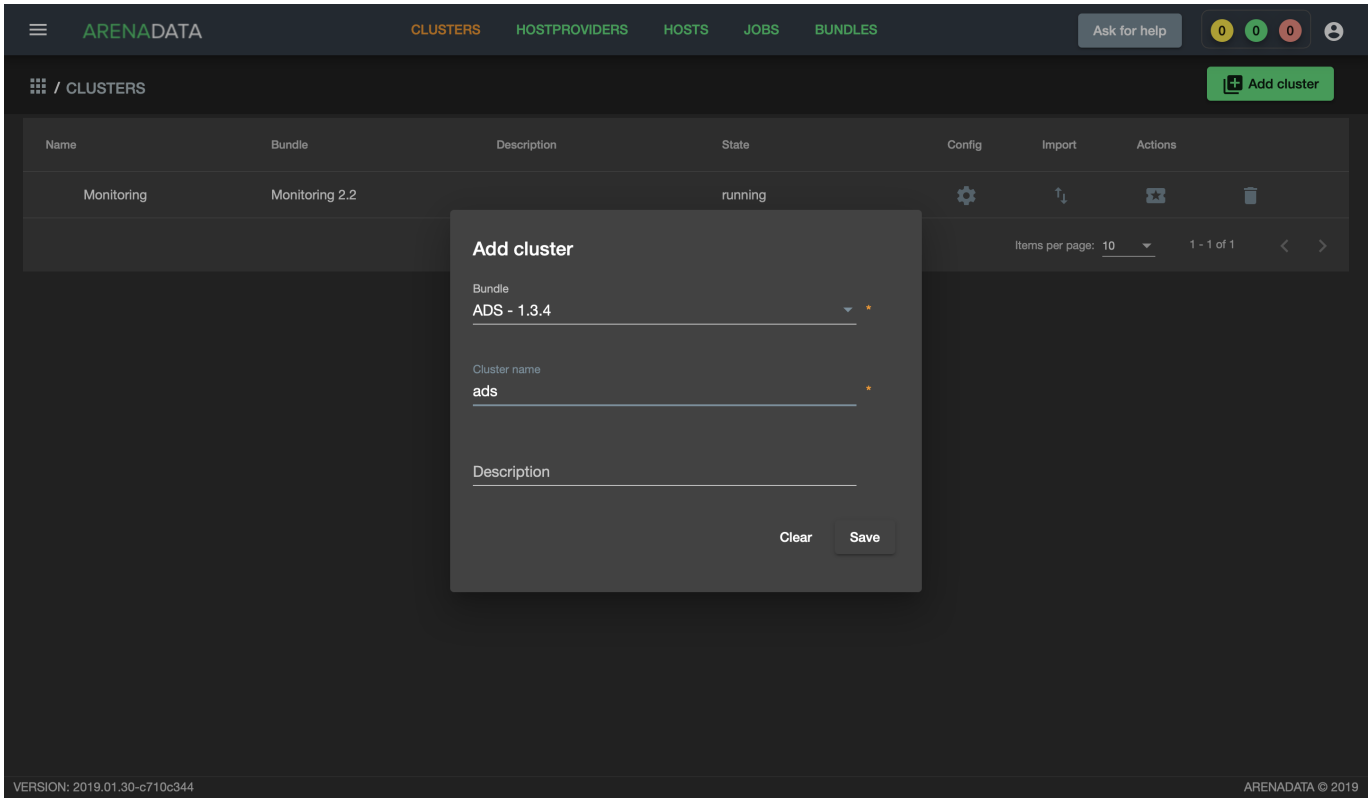


Рис.2.6.: Creating a cluster instance

Add services

Cluster **ADS** contains the following services:

- *Zookeeper* – service for storing configurations, performing distributed process synchronization;
- *Kafka* – distributed platform for streaming operations and data;
- *Nifi* – distributed platform for building and automating data flows between different systems;
- *Monitoring Clients* – agents that send host and ADS information to monitoring.

Not all services are required to be installed. For example if you do not plan to use **Nifi** then there is no need to add this service. If you have your own monitoring server (not **Graphite** based) there is no need for agents from the *monitoring clients* service. However, if you plan to use **Kafka**, the service of the same name and **Zookeeper** are required, the same can be said about the service **Nifi**. At the same time, the service may consist of mandatory and optional components. For example, the *Kafka* service consists of the mandatory component *broker* and the optional components: *manager* and *schema-registry*.

In this example, all services are added to the cluster:

- *Setting up the Zookeeper service*;
- *Setting up the Kafka service*;
- *Setting up the Nifi service*;
- *Setting up the Monitoring Clients service*.

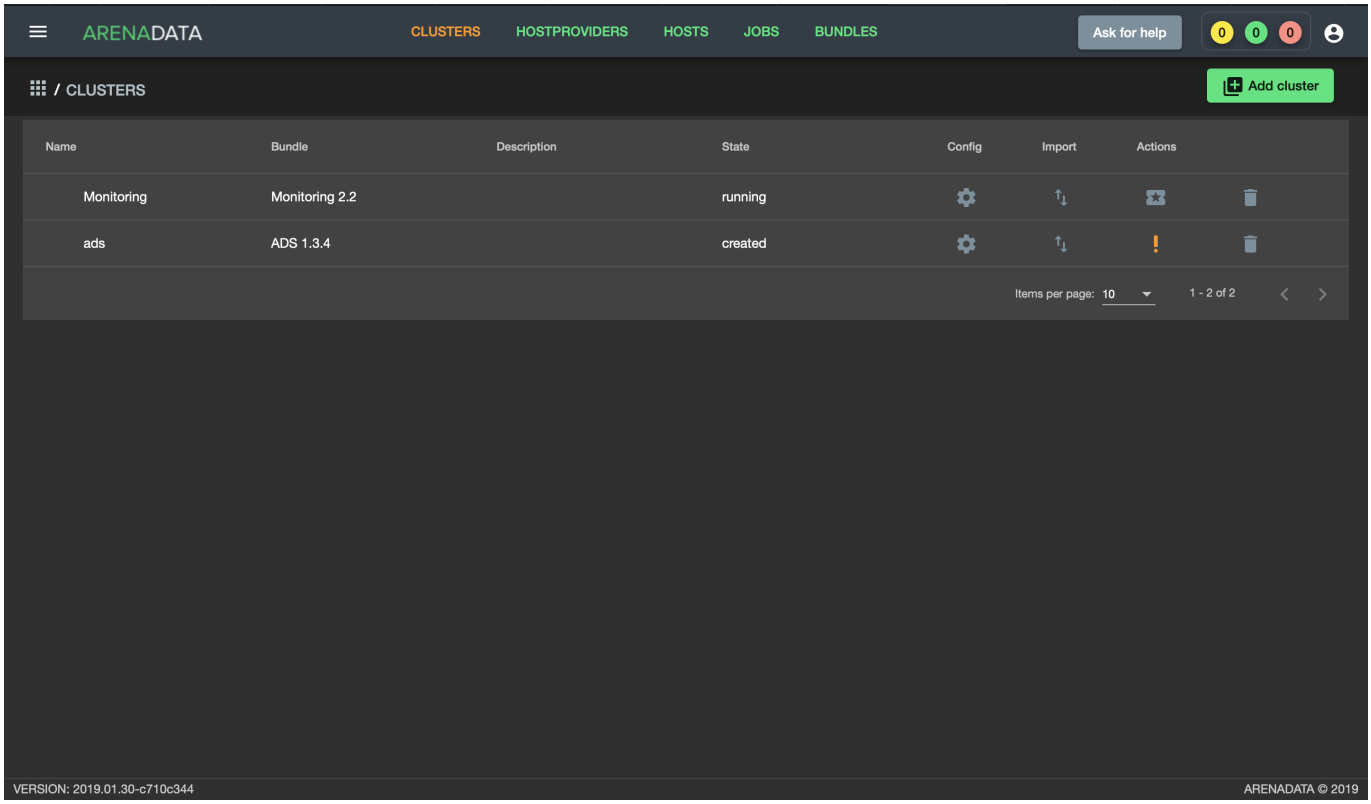


Рис.2.7.: Cluster instance creation success

Important: Currently it is not possible to remove already added service from the cluster

Setting up the Zookeeper service

To go to the settings of the service *Zookeeper*, press the button with the gear icon in the necessary row of the "SERVICES" tab in the **ADCM** and go to the "Configuration" menu section. This opens the configuration window for the selected service (Pic.2.9.).

The settings block "Main" contains the following parameters:

- *connect* – the Znode connection string, in which Zookeeper stores the configuration of the current cluster, is used by the *Kafka* service. In the current implementation, this parameter is not editable and is automatically generated on the ADCM side;
- *data_dirs* – directory for storing transaction logs Zookeeper. This parameter is specified as *dataDir* in the configuration file *zoo.cfg*;
- *client_port* – the port on which Zookeeper listens to client connections.

The settings block "Advanced" contains the following parameters:

- *zoo_cfg_content* – the contents of the file *zoo.cfg*, which is further templateable. This parameter can be used to make additional settings;
- *zookeeper_env_content* – the contents of the file *zookeeper-env.sh*, which is further template. This parameter can be used to make environment variables.

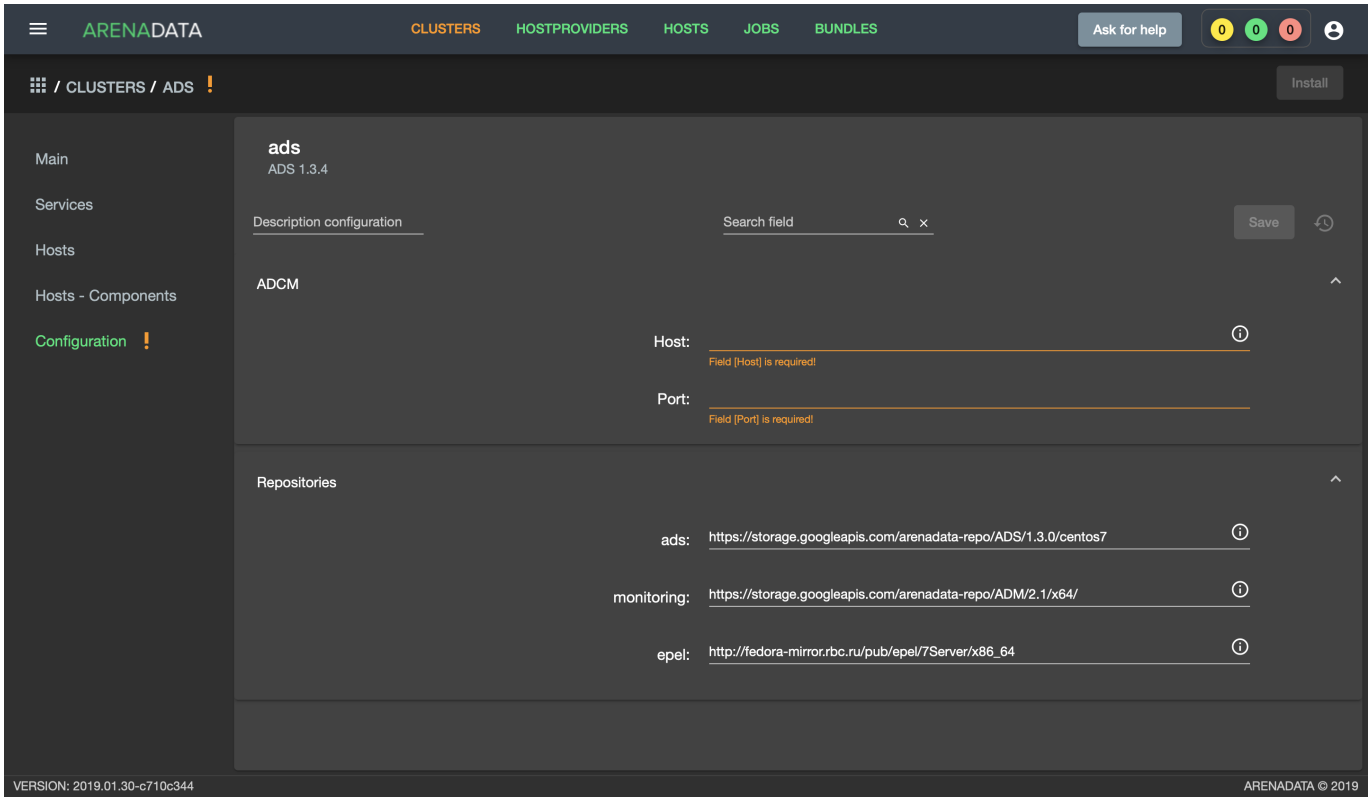


Рис.2.8.: Cluster configuration window

Setting up the Kafka service

To go to the settings of the service *Kafka*, press the button with the gear icon in the necessary row of the "SERVICES" tab in the **ADCM** and go to the "Configuration" menu section. This opens the configuration window for the selected service (Pic.2.10.).

The settings block "Main" contains the following parameters:

- *data_dirs* – directory for storing data in *Kafka*. Indicated to the quality of the parameter *log.dirs* in the configuration file server.properties;
- *listeners* – a list of URIs (protocol, host, and port on which the broker is run), separated by commas. If you are not using the *PLAINTEXT* protocol, then you must also specify *listener.security.protocol.map*. To bind to all interfaces, specify the host name as *0.0.0.0*. Leave the host name blank for binding to the default interface. It is specified as the *listeners* parameter in the configuration file server.properties;
- *default_replication_factor* – replication factor, with which topics are created and stored by default. It is specified as the parameter *efault.replication.factor* in the configuration file server.properties;
- *delete_topic_enable* – this option allows you to delete topics. If the parameter is disabled, deleting the topic through the administration tools does not lead to the actual deletion. It is specified as the parameter *default.replication.factor* in the configuration file server.properties;
- *log_retention_hours* – the number of hours that topics are stored in *Kafka*. It is specified as the *log.retention.hours* parameter in the configuration file server.properties;
- *log_roll_hours* – the maximum time after which a new segment log appears, even if the old log is not full. It is specified as the parameter *log.roll.hours* in the configuration file server.properties;

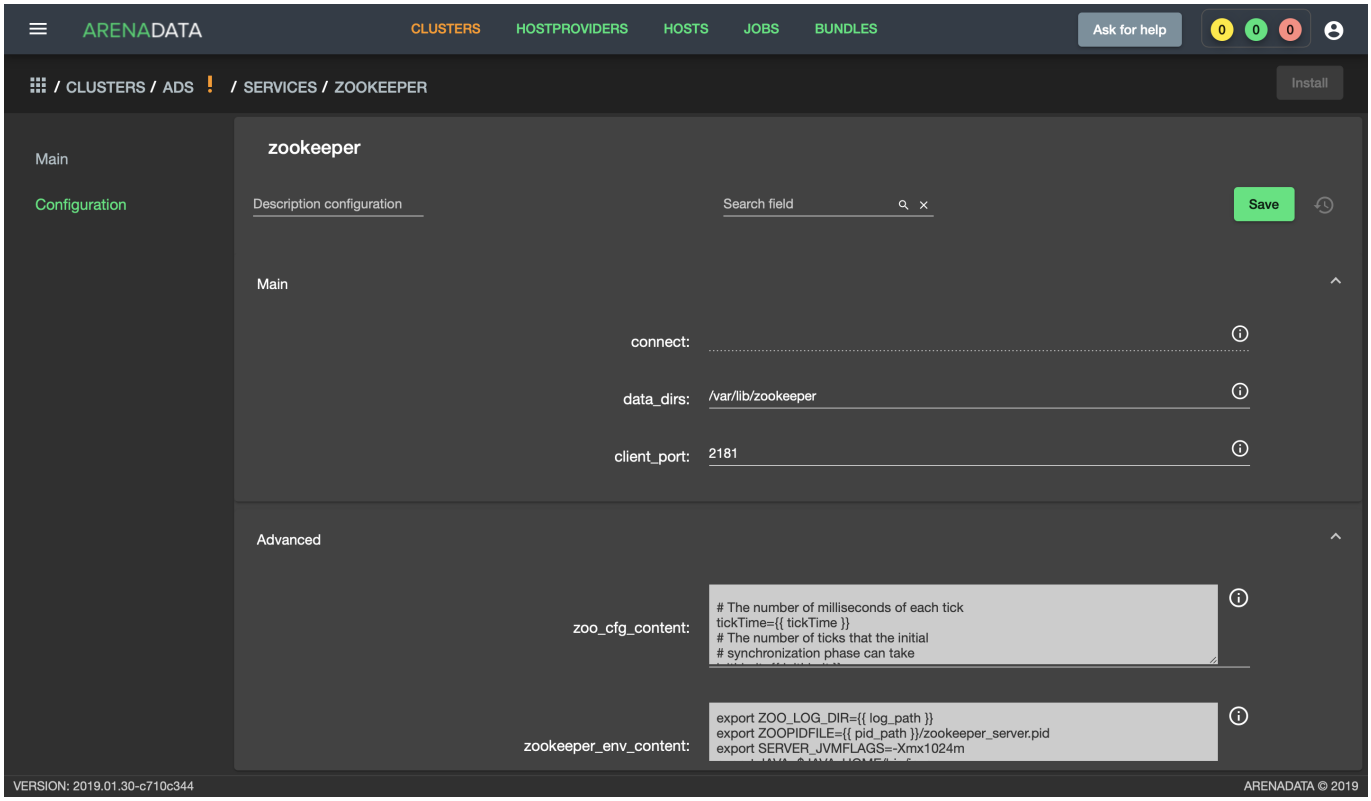


Рис.2.9.: Configuration window of Zookeeper service

- *broker_jmx_port* – port on which *Kafka* broker gives jmx-metrics. It is specified as the parameter *JMX_PORT* in the file *kafka-env.sh*;
- *manager_port* – port on which *Kafka-Manager* runs. Indicated in the file *kafka-manager-env.sh*;
- *schema_registry_heap_opts* – heap size allocated to the *schema-registry* process. Specified as the *SCHEMA_REGISTRY_HEAP_OPTS* parameter in *schema-registry-env.sh*;
- *schema_registry_listener_port* – port that is listening on *schema-registry*. Specified as the parameter *listeners* in the configuration file *schema-registry.properties*;

The settings block “Advanced” contains the following parameters:

- *server_properties_content* – the contents of the *server.properties* file, which is further templateable. This parameter can be used to make additional settings;
- *kafka_env_content* – the contents of the file *kafka-env.sh*, which is further template. This parameter can be used to make environment variables.

Setting up the Nifi service

To go to the settings of the service *Nifi*, press the button with the gear icon in the necessary row of the “SERVICES” tab in the **ADCM** and go to the “Configuration” menu section. This opens the configuration window for the selected service (Pic.2.11.).

The settings block “Main” contains the following parameters:

- *nifi_ui_port* – http-port on which the web interface of the service *Nifi* runs. It is specified as the parameter *nifi.web.http.port* in the configuration file *nifi.properties*;

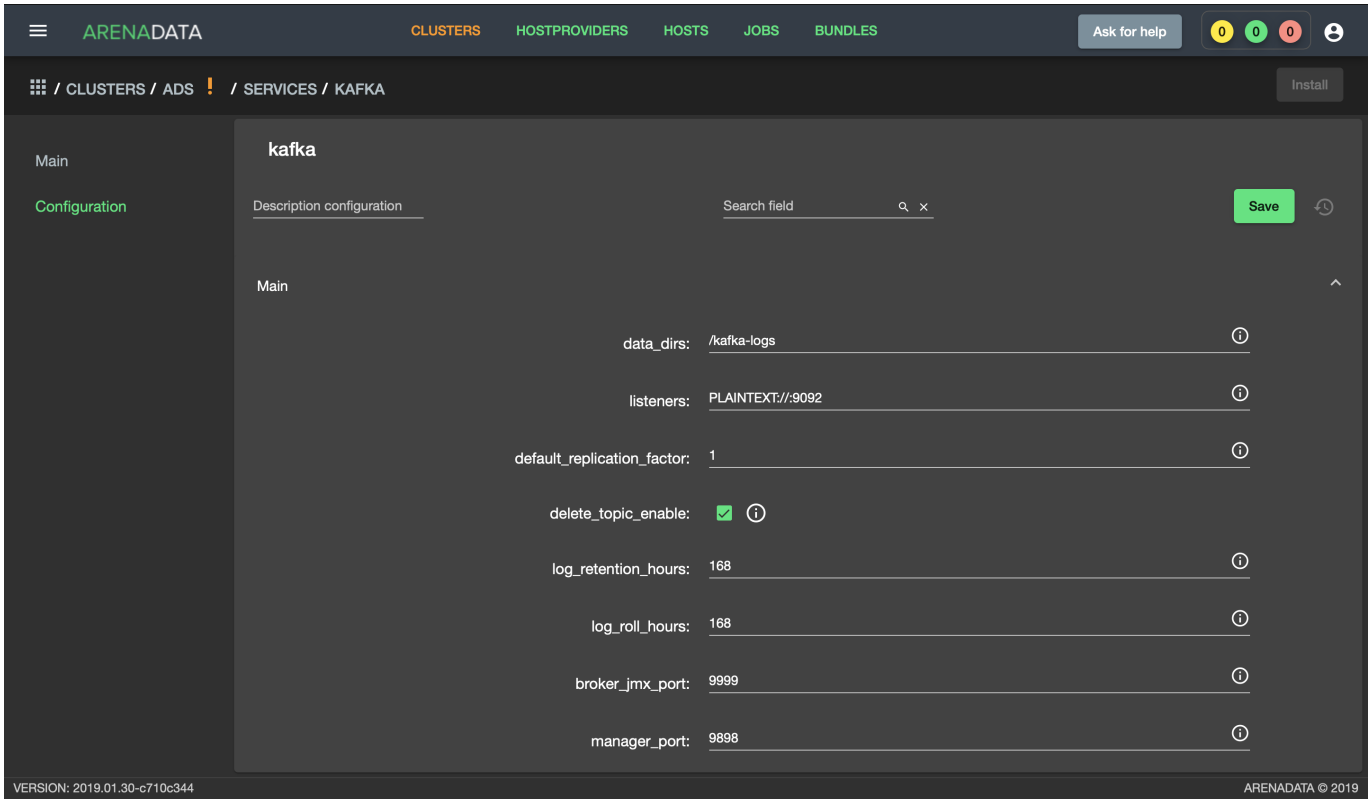


Рис.2.10.: Configuration window of Kafka service

- *nifi_node_jvm_memory* – heap size allocated to the service process *Nifi*. It is specified in the configuration file *bootstrap.conf*.

The following optional parameters are set in the "Custom" settings block:

- *nifi_custom_nars* – parameter should be used when adding *custom nars*; comma separated. Specified as *nifi.nar.library.directory.lib...* in the configuration file *nifi.properties*;

The settings block "Advanced" contains the following parameters:

- *nifi_properties_content* – the contents of the file *nifi.properties*, which will be further templateable. This parameter can be used to make additional settings;
- *nifi_env_content* – the contents of the file *nifi-env.sh*, which is further templateable. This parameter can be used to make environment variables;
- *bootstrap_content* – the contents of the file *bootstrap.conf*, which is further templateable. This field can be used to make settings related to the start of the service;
- *logback_content* – the contents of the file *logback.xml*, which is further templateable. This field can be used to make settings related to logging;
- *state_management_content* – the contents of the file *state_management.xml*, which is further templateable. This field can be used to make settings related to storing the state of the service *Nifi*;
- *authorizers_content* – the contents of the file *authorizers.xml*, which is further template. This field can be used to make authorization settings to the *Nifi* service if security policies are configured;
- *login_identity_providers_content* – the contents of the file *login_identity_providers.xml*, which is further template. This field can be used to make authorization settings used by the *state* provider, if security policies

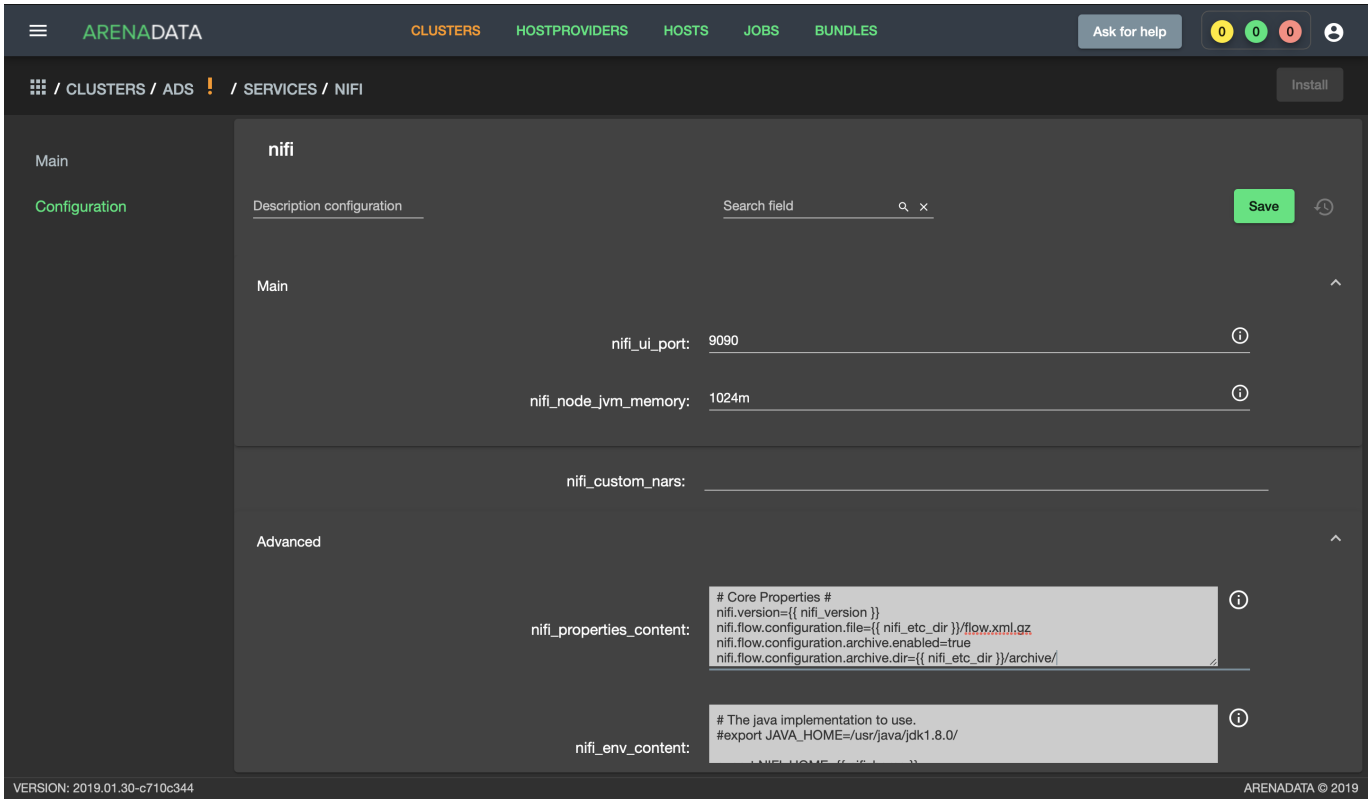


Рис.2.11.: Configuration window of Nifi service

are configured.

Setting up the Monitoring Clients service

To go to the settings of the service *monitoring clients*, press the button with the gear icon in the necessary row of the "SERVICES" tab in the **ADCM** and go to the "Configuration" menu section. This opens the configuration window for the selected service (Pic.2.12.).

The settings block "Advanced" contains the following parameters:

- *kafka_dashboard* – a file in the format *json*, which is further template and sent to *Grafana*;
- *kafka_metrics* – a file in the format *yaml*, which will be further template. Includes *jmx* broker metrics *Kafka*.

Add hosts

To add hosts to the cluster *ads*:

1. Open the "Hosts" tab in the *ads* cluster menu (Pic.2.13.).
2. Press the "Add hosts" button and in the opened form select the required hosts (Pic.2.14.).
3. As a result, the fact of adding hosts is displayed in the *ads* cluster in the list of the "Hosts" tab (Pic.2.15.).

Map service components to hosts

Each service consists of components that should be mapped to some hosts in the cluster. To do this, a component should be selected on the "Hosts - Components" tab by clicking on it in the "Components" column and mapped with a click to the required host in the "Hosts" column (Pic.2.16.).

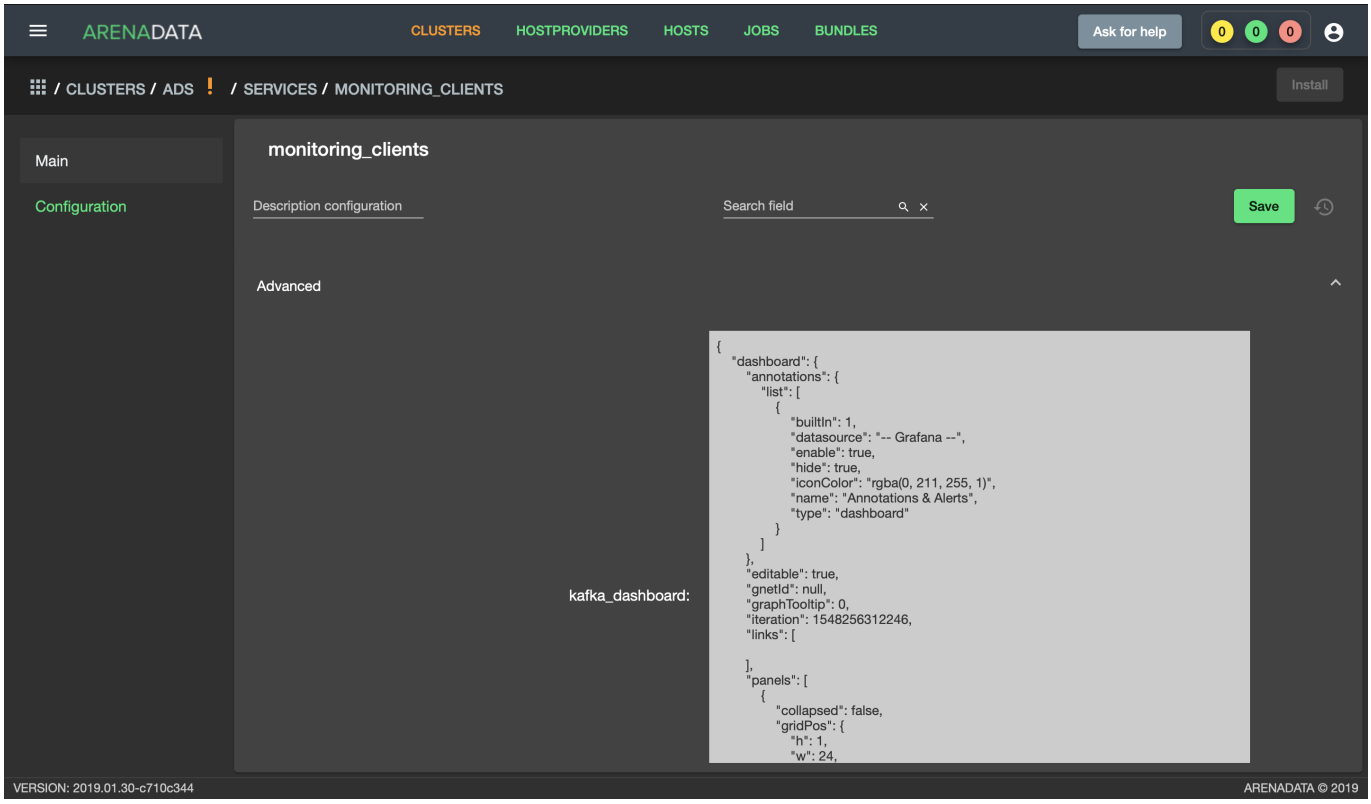


Рис.2.12.: Configuration window of Monitoring Clients service

Because the a*Zookeeper*, *Kafka*, *Nifi* and *Monitoring Clients* services added to the **ADS** cluster, but are not mapped yet, initially there are no components on any of the hosts:

1. Components of the *Zookeeper* service are (Pic.2.17.):

- *Zookeeper.SERVER* – should be added to one or an odd number of hosts (*zk1*); for large clusters no more than *5* are recommended.

2. Components of the *Kafka* service are (Pic.2.18.):

- *kafka.BROKER* – should be added to one or more broker hosts (*kafka1*, *kafka2*);
- *kafka.MANAGER* – optionally could be added to any one host (*kafka1*);
- *kafka.SCHEMA_REGISTRY* – optionally could be added to any one host (*kafka2*).

3. Components of the *Nifi* service are (Pic.2.19.):

- *nifi.SERVER* – should be added to one or more hosts (*df-mdw*);

4. Components of the *monitoring clients* service are (Pic.2.20.):

- *monitoring_clients.diamond* – must be added to all hosts (*zk1*, *kafka1*, *kafka2*, *nifi2*). Collects system metrics and sends them to **ADCM**;
- *monitoring_clients.jmxtrans* – must be added to one host (*zk1*). Collects service-specific metrics and sends them to **ADCM**;

Ways to install services

There are two ways to install services:

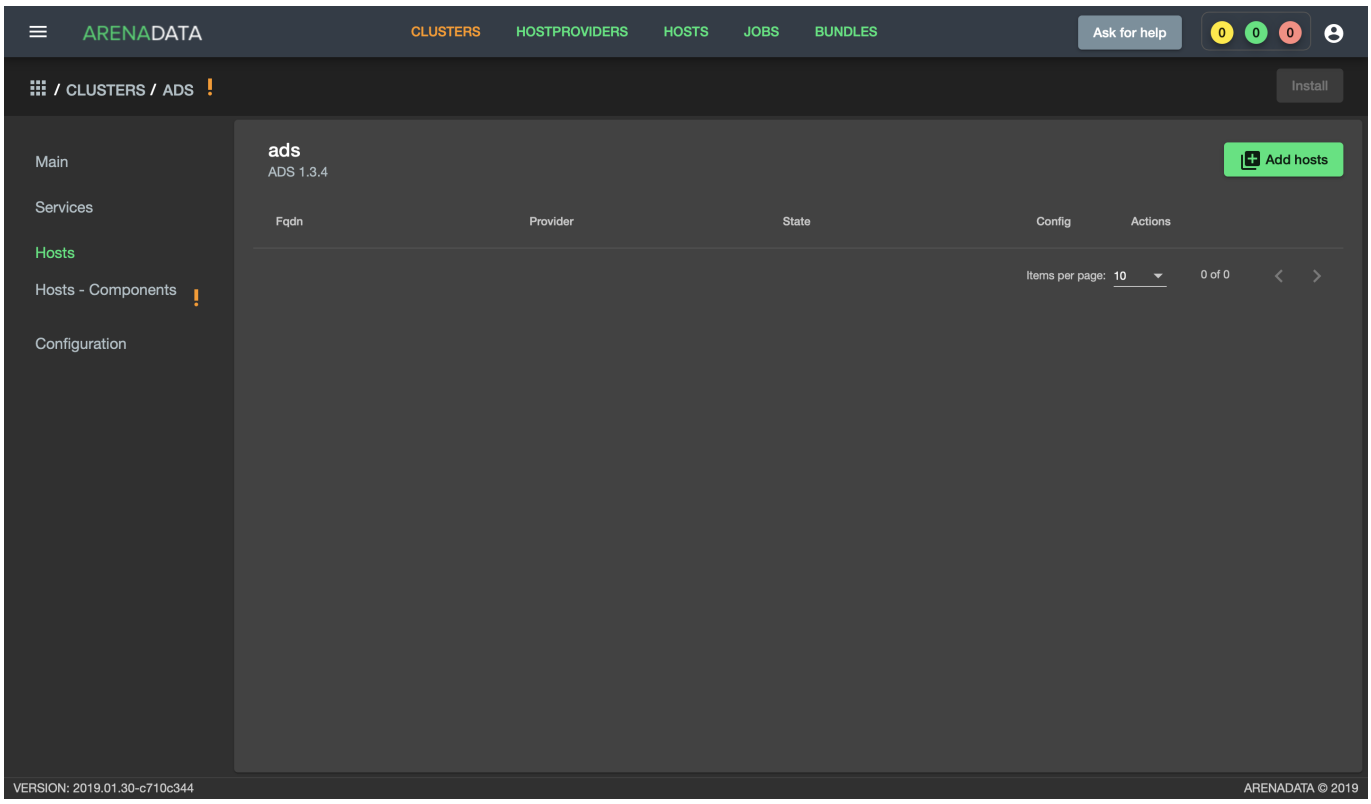


Рис.2.13.: “Hosts” tab of the ads cluster

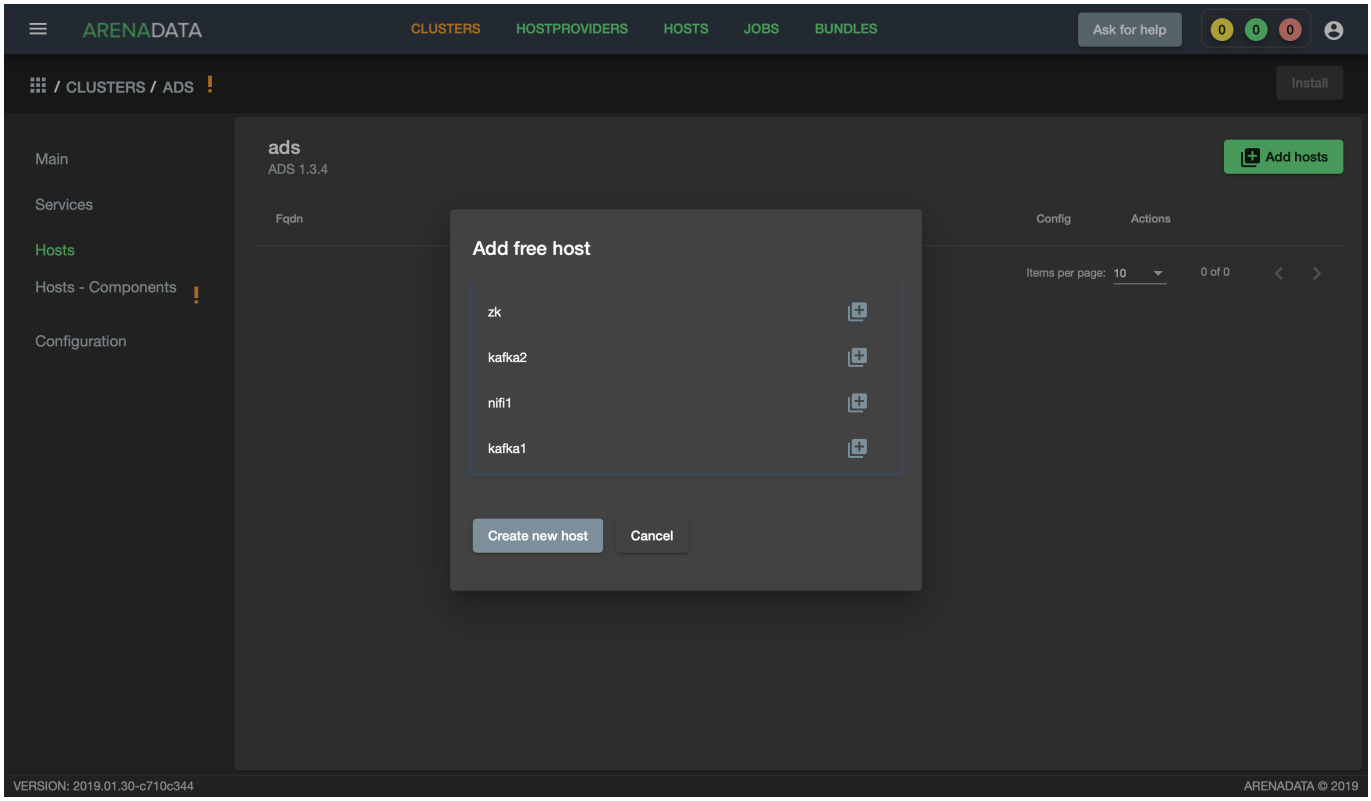


Рис.2.14.: Choose hosts

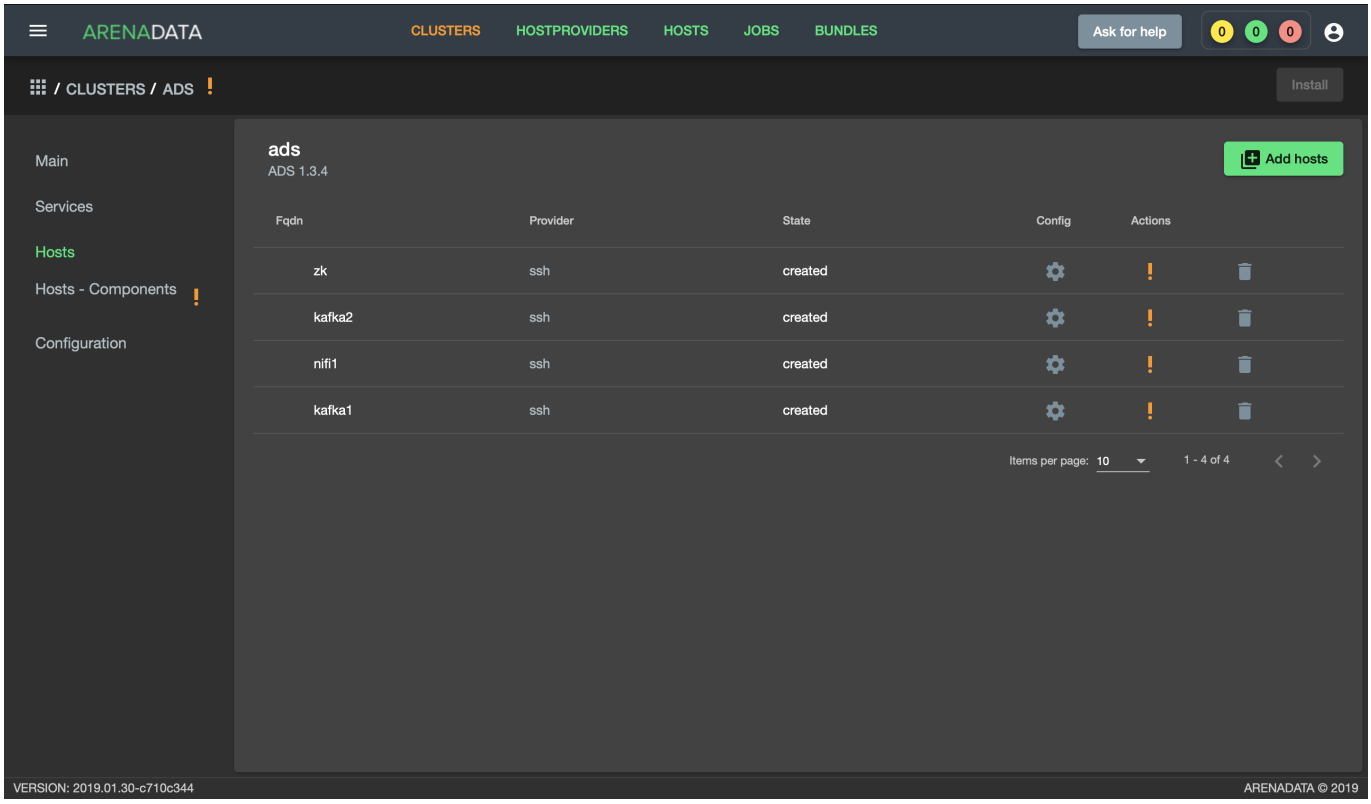


Рис.2.15.: Result of successfully adding hosts

- Install all cluster services. The operation at the cluster level is a sequential call to the installation and configuration of all added services;
- Custom installation service. The operation at the service level is the installation of a single service.

Important: Regardless of the type of installation selected, you initially need to import the settings from the *Monitoring* cluster in case the *Monitoring Clients* service is added

Install all cluster services

To install all added services in a cluster *ads*, select the corresponding cluster in **ADCM** and click the *Install* button in the upper right corner of the screen form (Pic.2.21.).

As a result of the installation all the added services change the state from *created* to *installed* (Pic.2.22.).

Install Zookeeper service

To install the *Zookeeper* service, you need to perform a number of actions on the cluster tab “Services”:

- In the “Actions” field click on the icon in the *Zookeeper* service row and select the action *Install* (Pic.2.23.). At this point, the hosts are configured, the necessary packages are installed.
- As a result of the installation, the service *Zookeeper* changes state from *created* to *installed*.

Install Kafka service

To install the *Kafka* service, you need to perform a number of actions on the cluster tab “Services”:

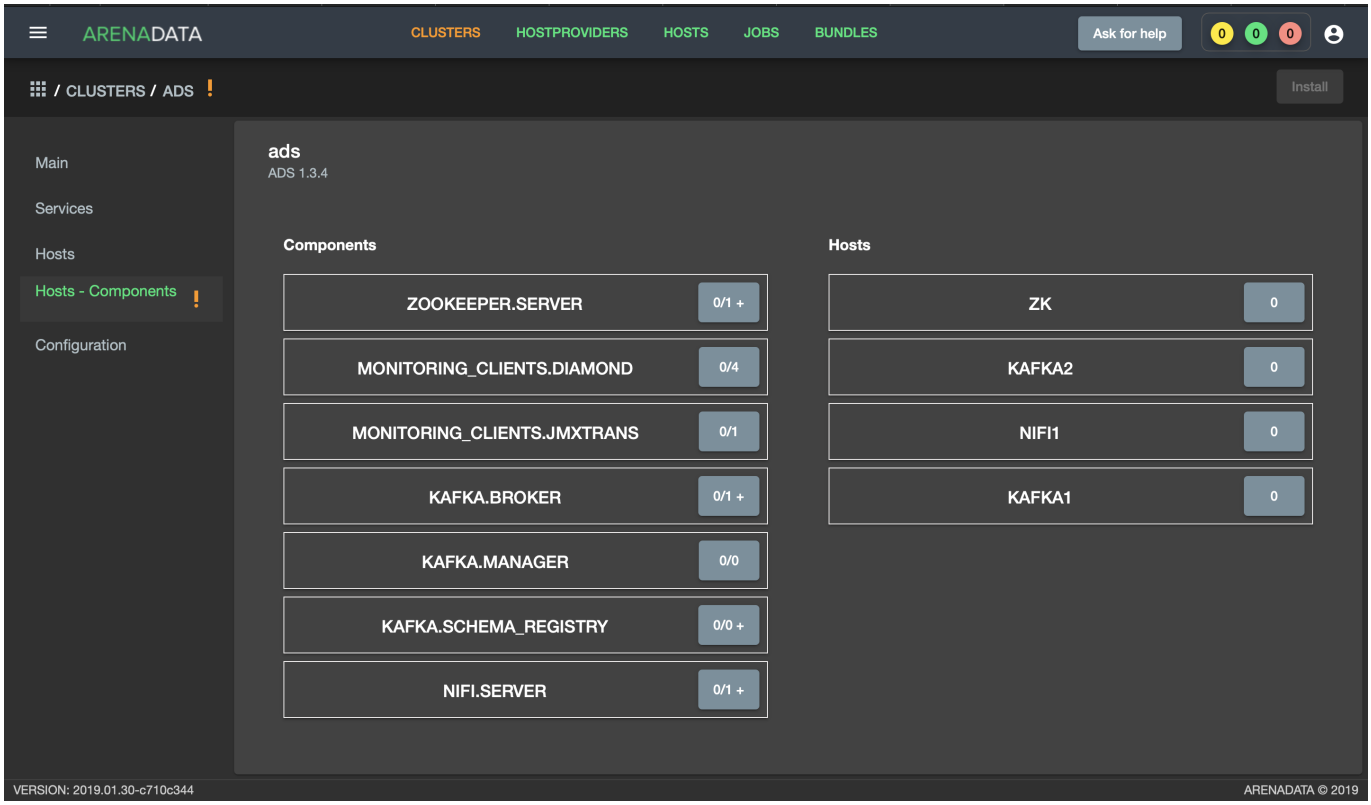


Рис.2.16.: Map service components to hosts

- In the “Actions” field click on the icon in the *Kafka* service row and select the action *Install* (Pic.2.24.).
- As a result of the installation, the service *Kafka* changes state from *created* to *installed*.

Install Nifi service

To install the *Nifi* service, you need to perform a number of actions on the cluster tab “Services”:

- In the “Actions” field click on the icon in the *Nifi* service row and select the action *Install* (Pic.2.25.).
- As a result of the installation, the service *Nifi* changes state from *created* to *installed*.

Install Monitoring Clients service

The *monitoring clients* service is optional and requires importing the configuration parameters of the monitoring cluster (addresses, login/password) to the cluster *ads*:

1. To import the monitoring configuration to the *ads* cluster, open the “CLUSTERS” tab in **ADCM**, select the *Import* option and mark the imported service settings using the checkboxes in the opened form (Pic.2.26.).
2. To install monitoring clients in the cluster *ads*:

- Return to the cluster tab “Services”. Press in the “Actions” field on the icon in the *monitoring clients* service row and select the action *Install* (Pic.2.27.).
- As a result of the installation, the service *monitoring clients* changes state from *created* to *monitored*.

2.3.3 Manage tools

ADS provides for 2 types of operations:

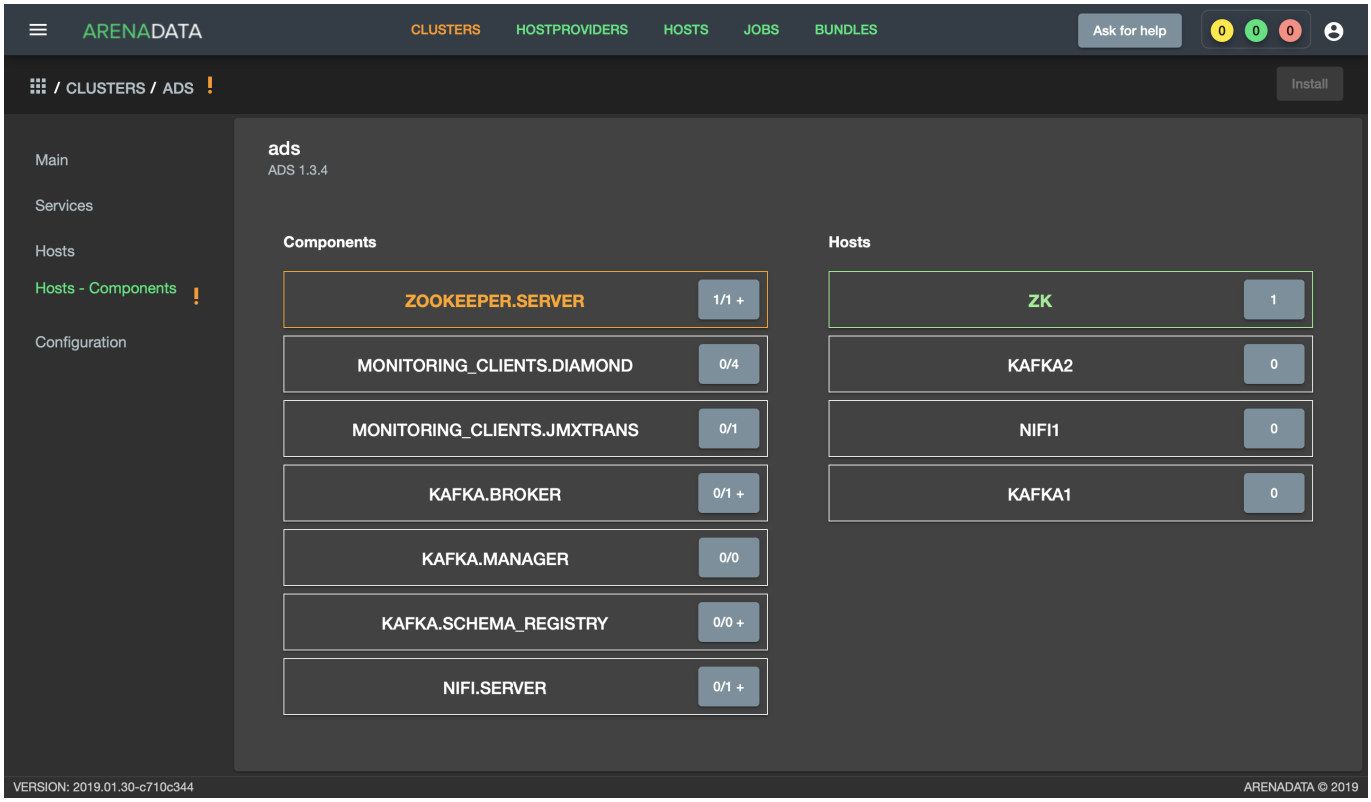


Рис.2.17.: Components of the *Zookeeper* service

- *Cluster level operations* – performing operations sequentially on all services;
- *Service level operations* – performing operations on a particular service.

Cluster level operations

Start and stop ADS – there is a possibility of sequential start and stop of all cluster services *ads* through **ADCM**. To do this, open in **ADCM** cluster *ads* and click the *Start*, *Stop* or *Restart* depending on the required work with the cluster (Pic.2.28).

As a result of a particular action, the cluster changes its state to the corresponding – *running* or *stopped*, in the event that the cluster is already installed and is not in the *created* state.

Service level operations

For each of the services there is an opportunity to check its working capacity, as well as to manage it independently of the others.

Health check Zookeeper service

Health check of the *Zookeeper* service is a connection to the *Zookeeper* quorum, creating test *znode* in it and checking the availability of the created *znode* to each of the quorum hosts.

To check the results of actions on the service *Zookeeper* you need:

1. Go to the cluster *ads* in ADCM, on the "Services" tab for the *Zookeeper* service in the "Actions" field click on the icon and select the *Check* action (Pic.2.29).
2. Open the "JOBS" tab (Pic.2.30).

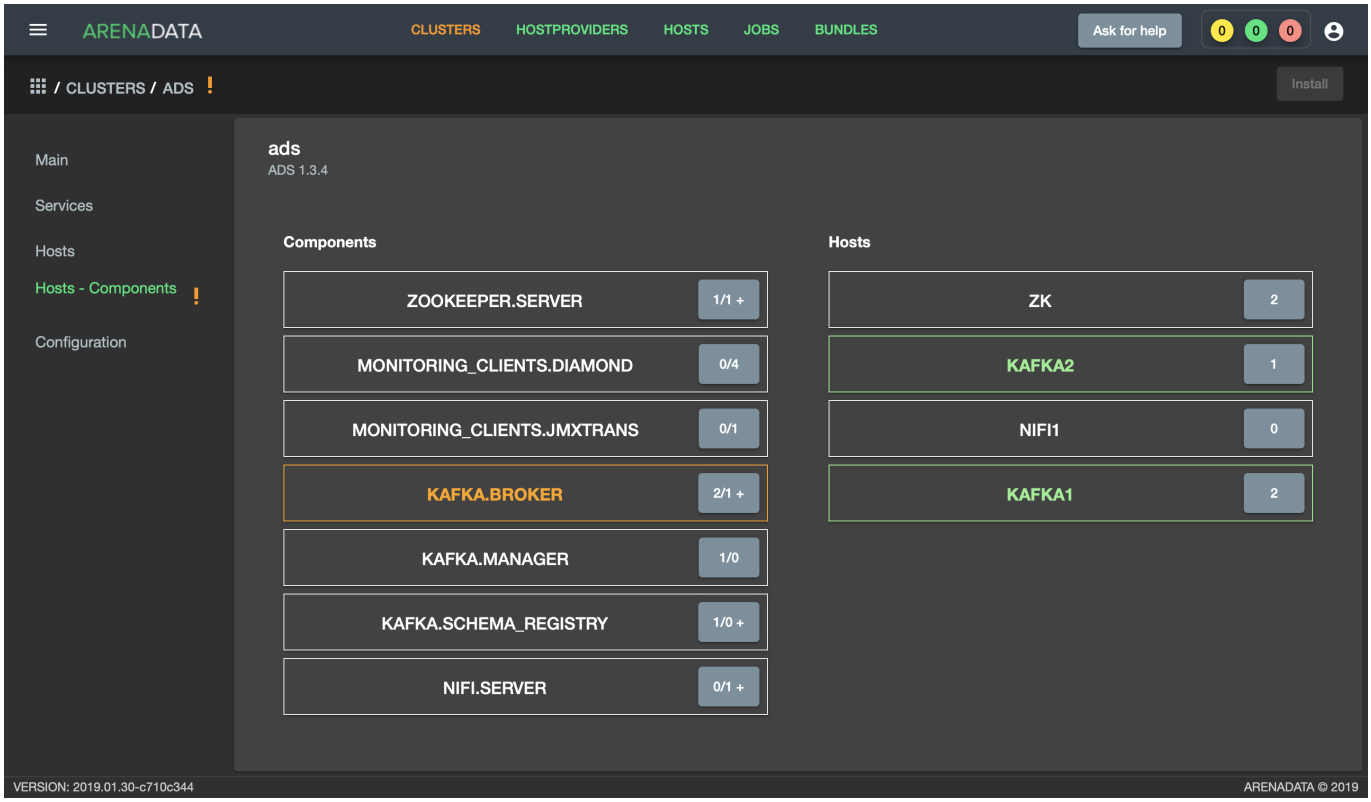


Рис.2.18.: Components of the *Kafka* service

3. Select the last check action over *ads* cluster and get results (Pic.2.31).

Health check Kafka service

Health check of the *Kafka* service is the creation of test topics on each of the *BROKER* hosts and checking the availability of the created topics.

To check the results of actions on the service *Kafka* you need:

1. Go to the cluster *ads* in ADCM, on the "Services" tab for the *Kafka* service in the "Actions" field click on the icon and select the *Check* action (Pic.2.32).
2. Open the "JOBS" tab (Pic.2.33).
3. Select the last check action over *ads* cluster and get results (Pic.2.34).

Start and stop services

For each of the services it is possible to control it independently of the others, to perform such operations as *Stop*, *Start*, *Restart*.

For example, in order to restart all the components of the service *Kafka* you need to go to the cluster *ads* in **ADCM**, on the "Services" tab for the *Kafka* service in the "Actions" field click on the icon and select the *Restart* action (see Pic.2.32).

As a result, the service status changes to *running* or *stopped*, if the service has already been installed and is not in the *created* state.

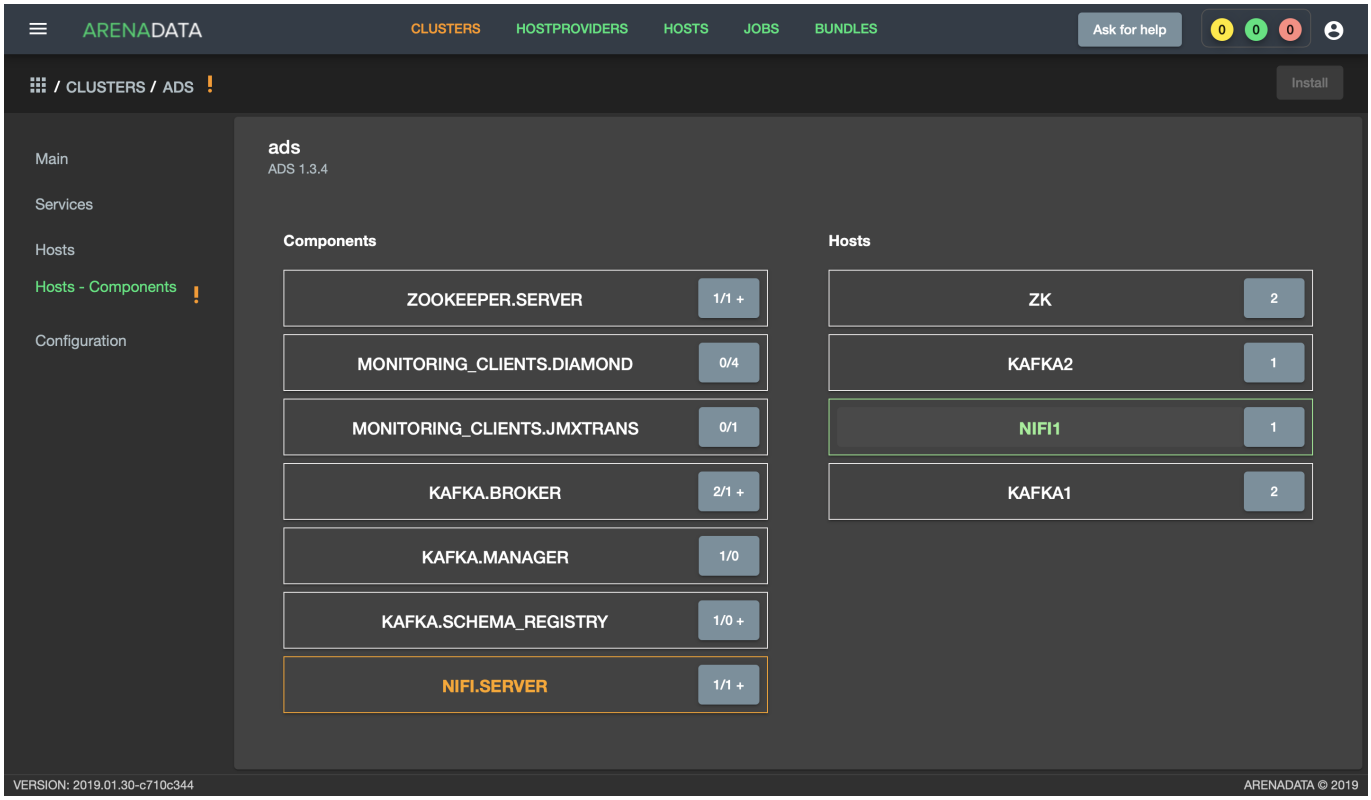


Рис.2.19.: Components of the *Nifi* service

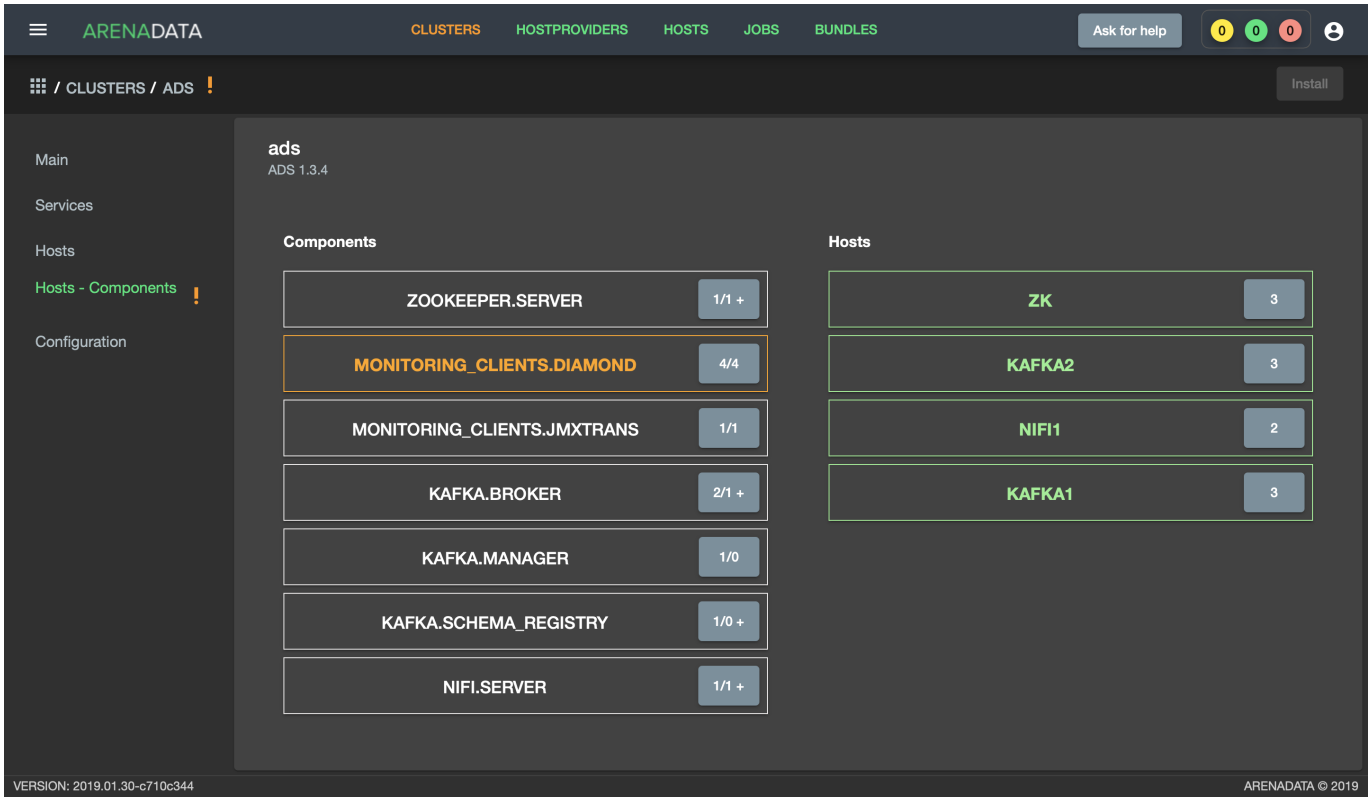


Рис.2.20.: Components of the *Monitoring Clients* service

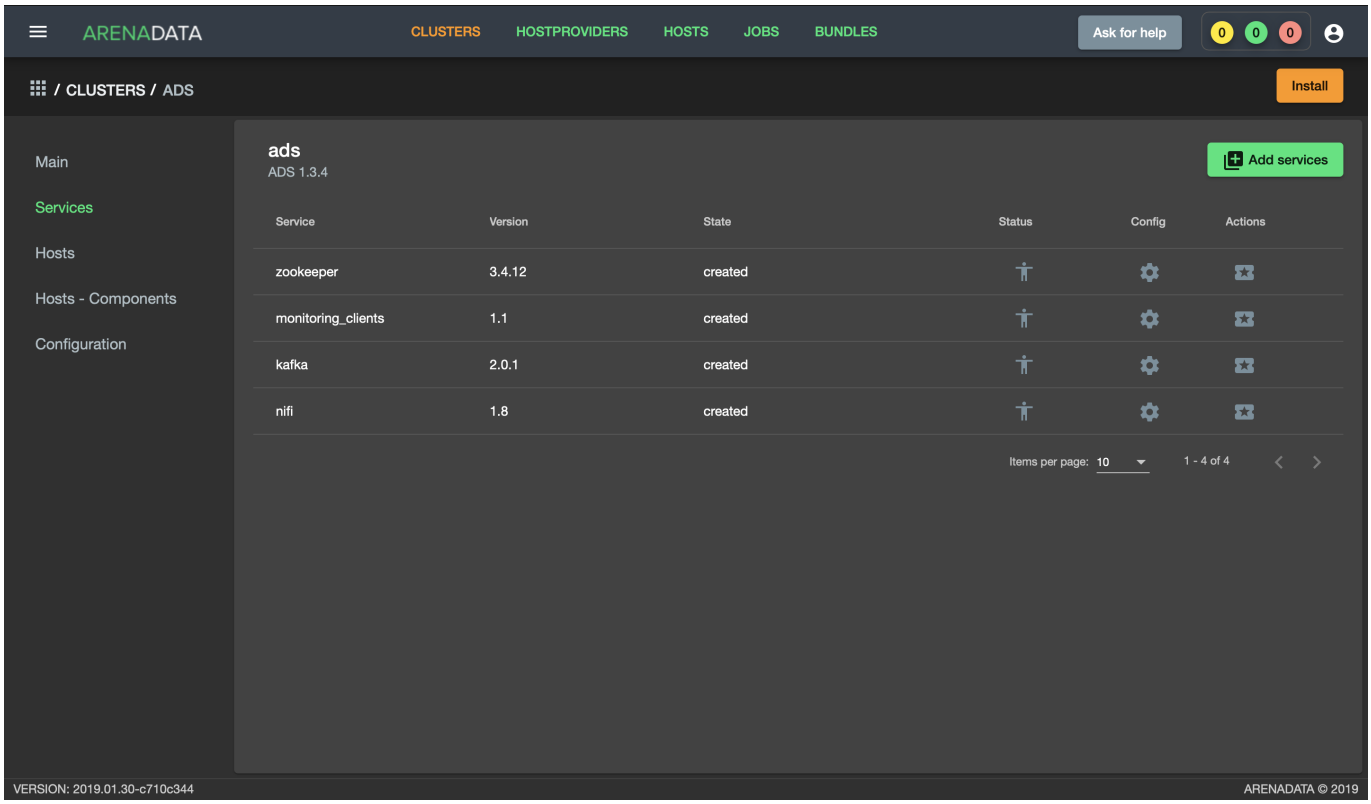


Рис.2.21.: Install all cluster services

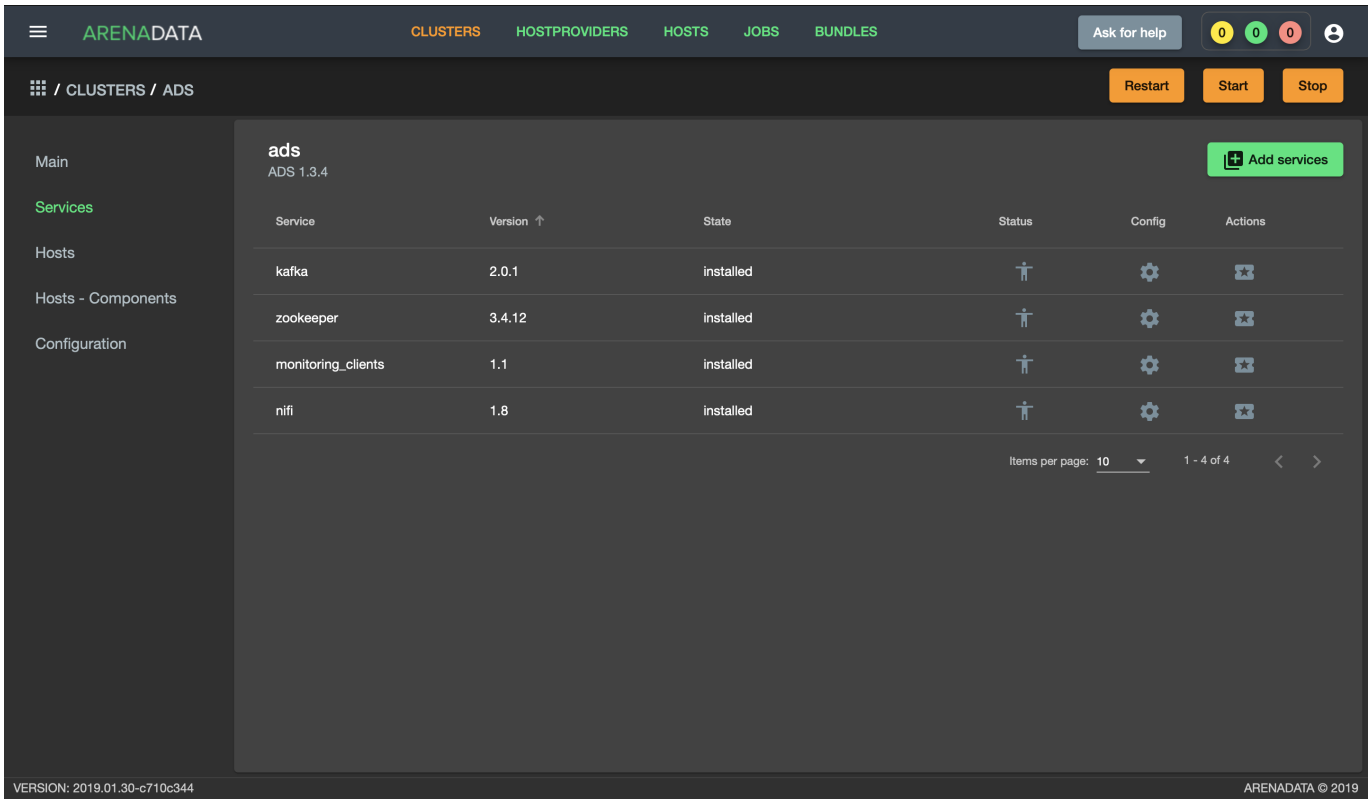


Рис.2.22.: Result of the installation

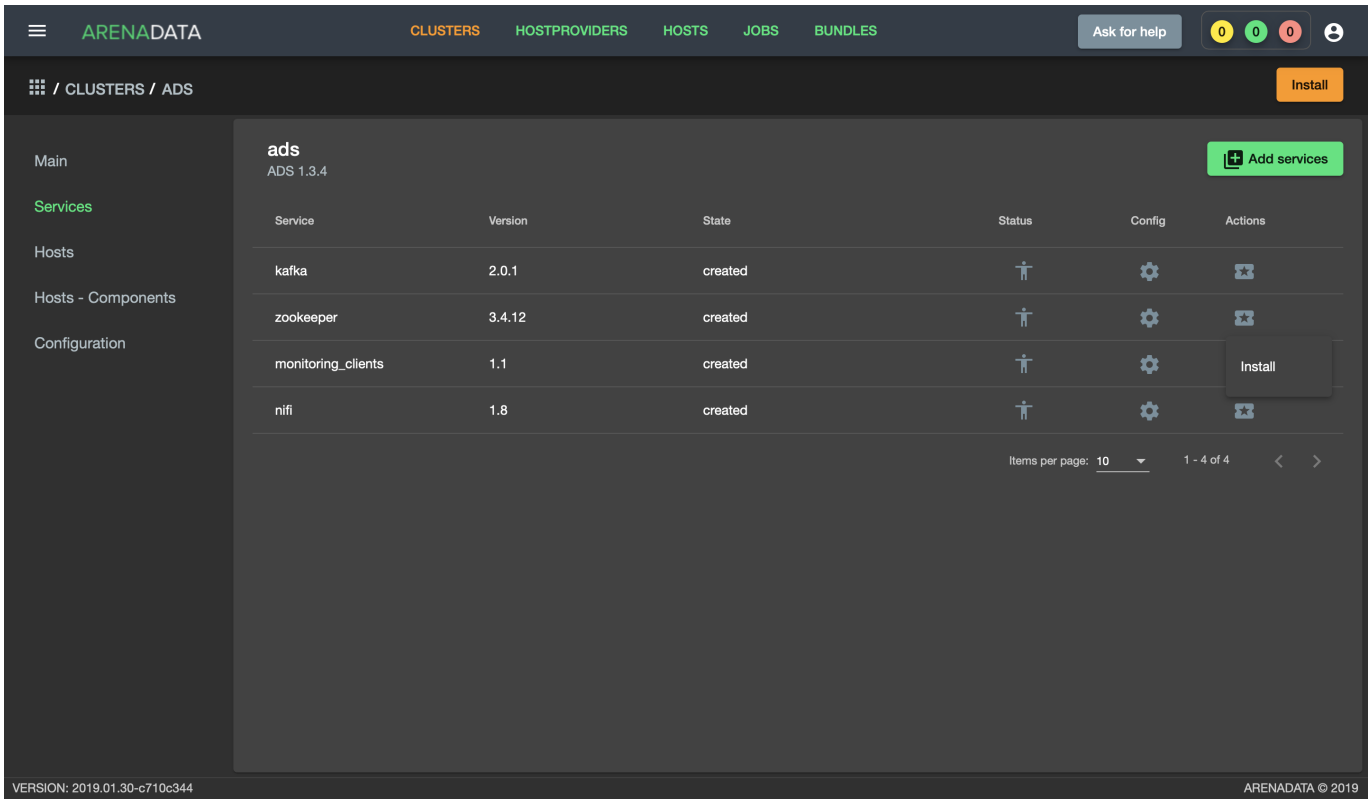


Рис.2.23.: Install Zookeeper

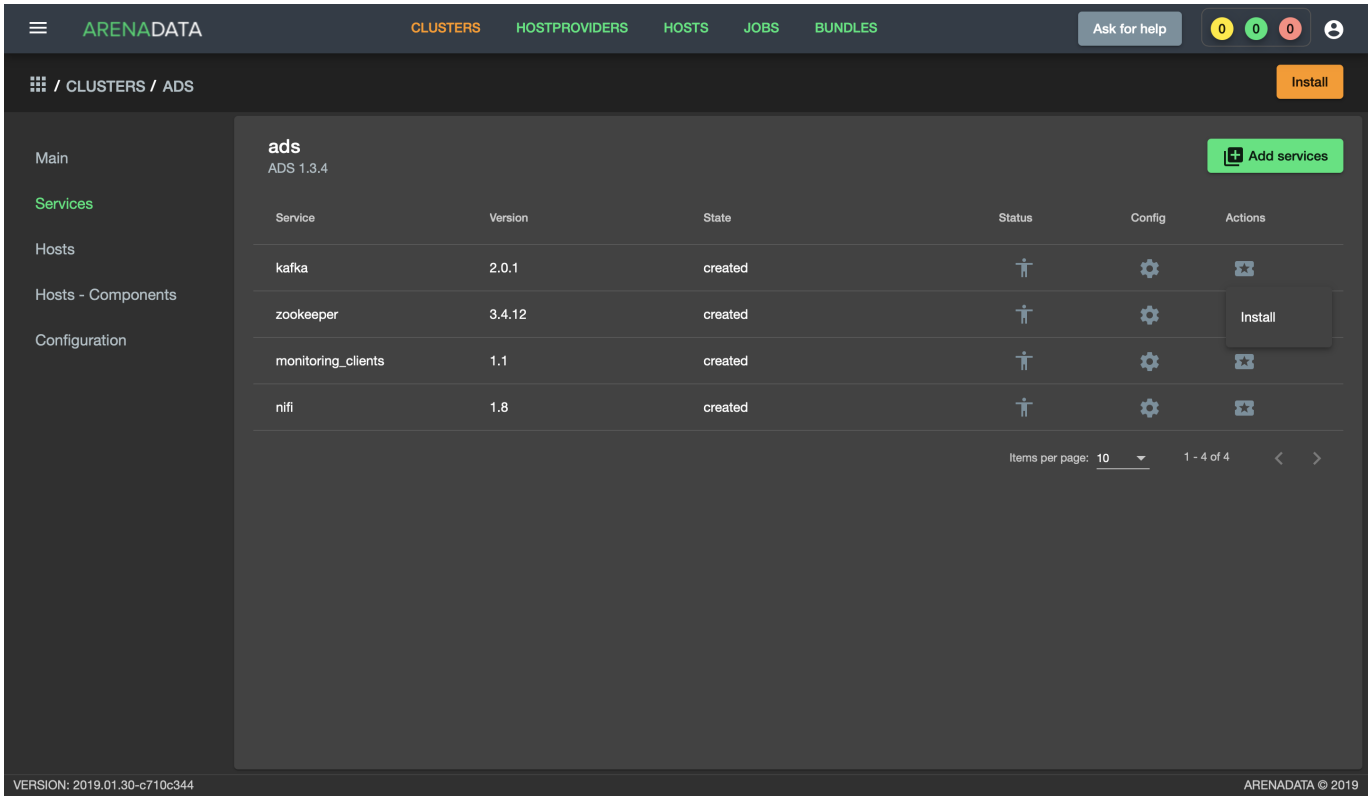


Рис.2.24.: Install Kafka

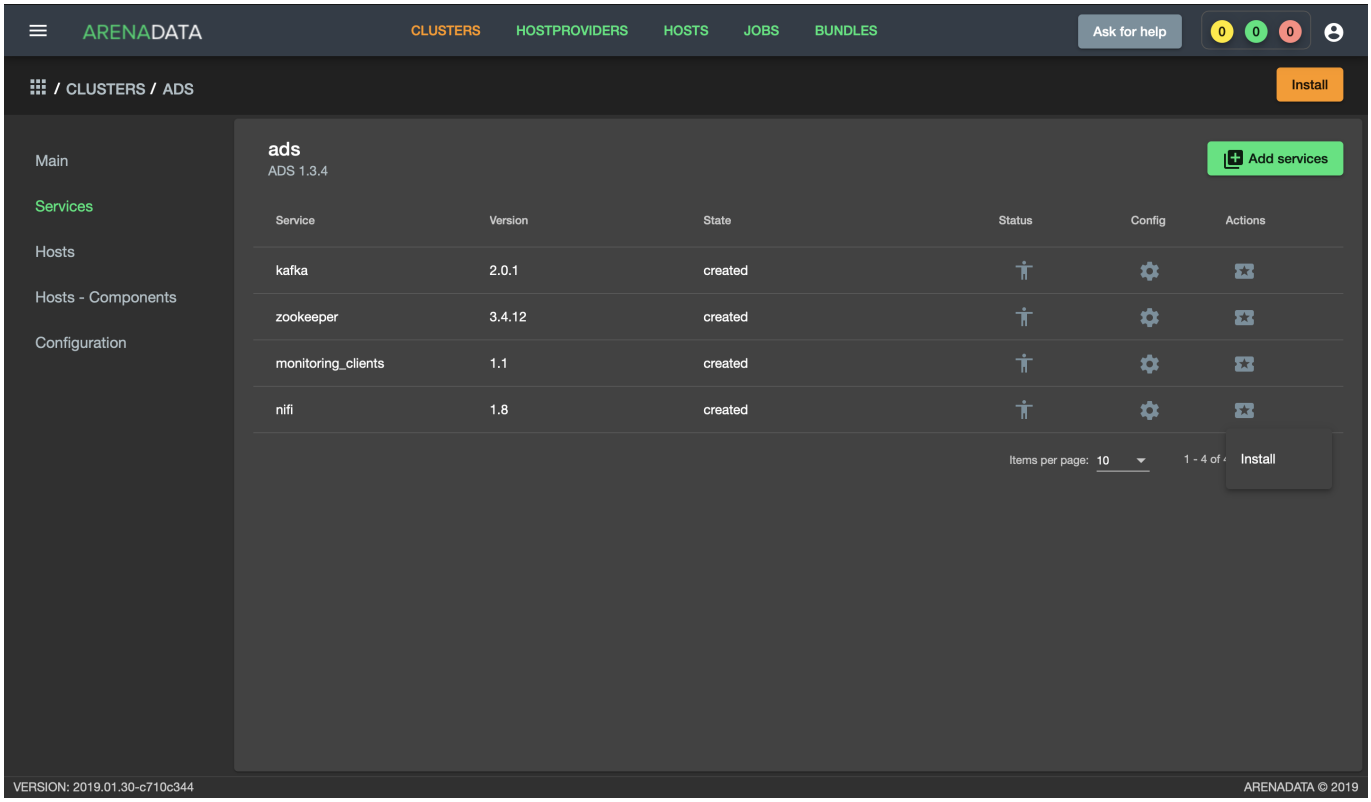


Рис.2.25.: Install Nifi

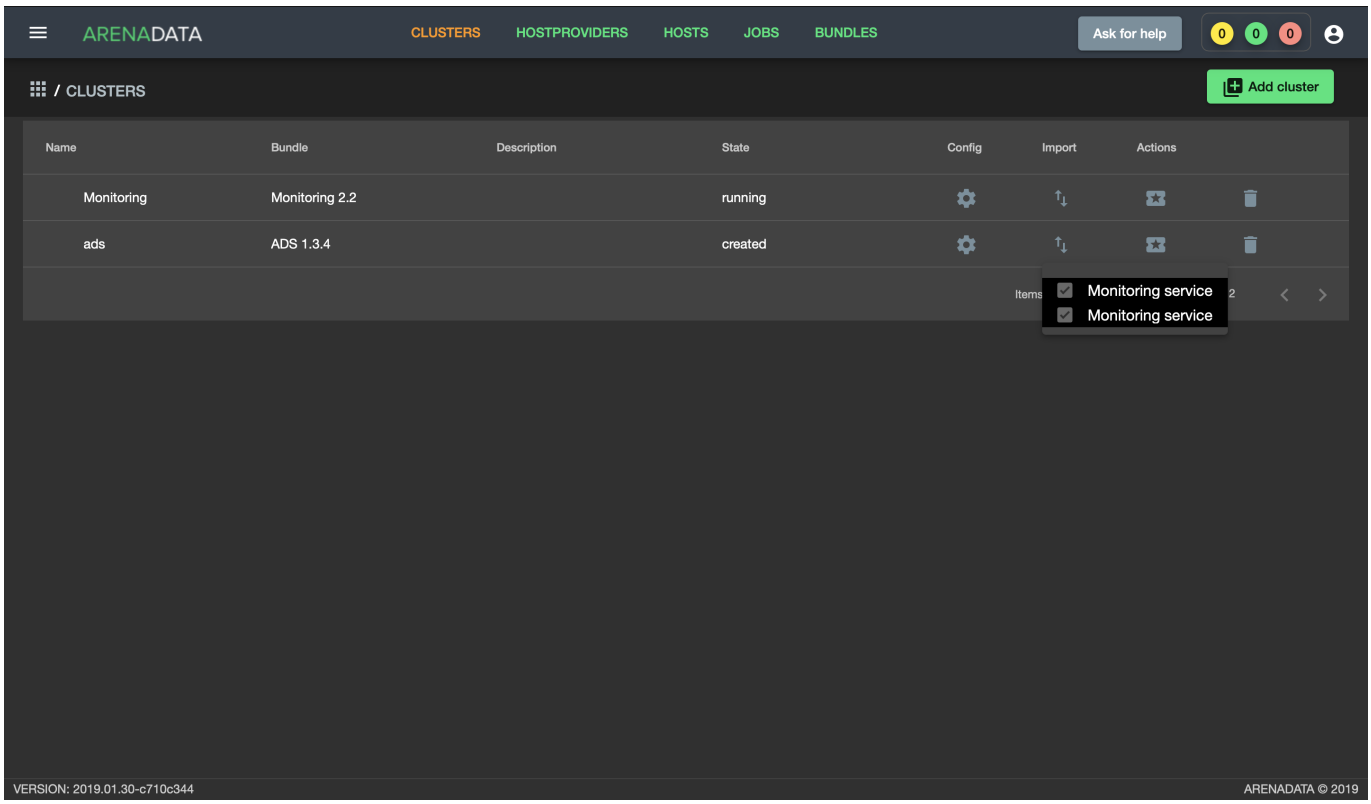


Рис.2.26.: Import monitoring configuration

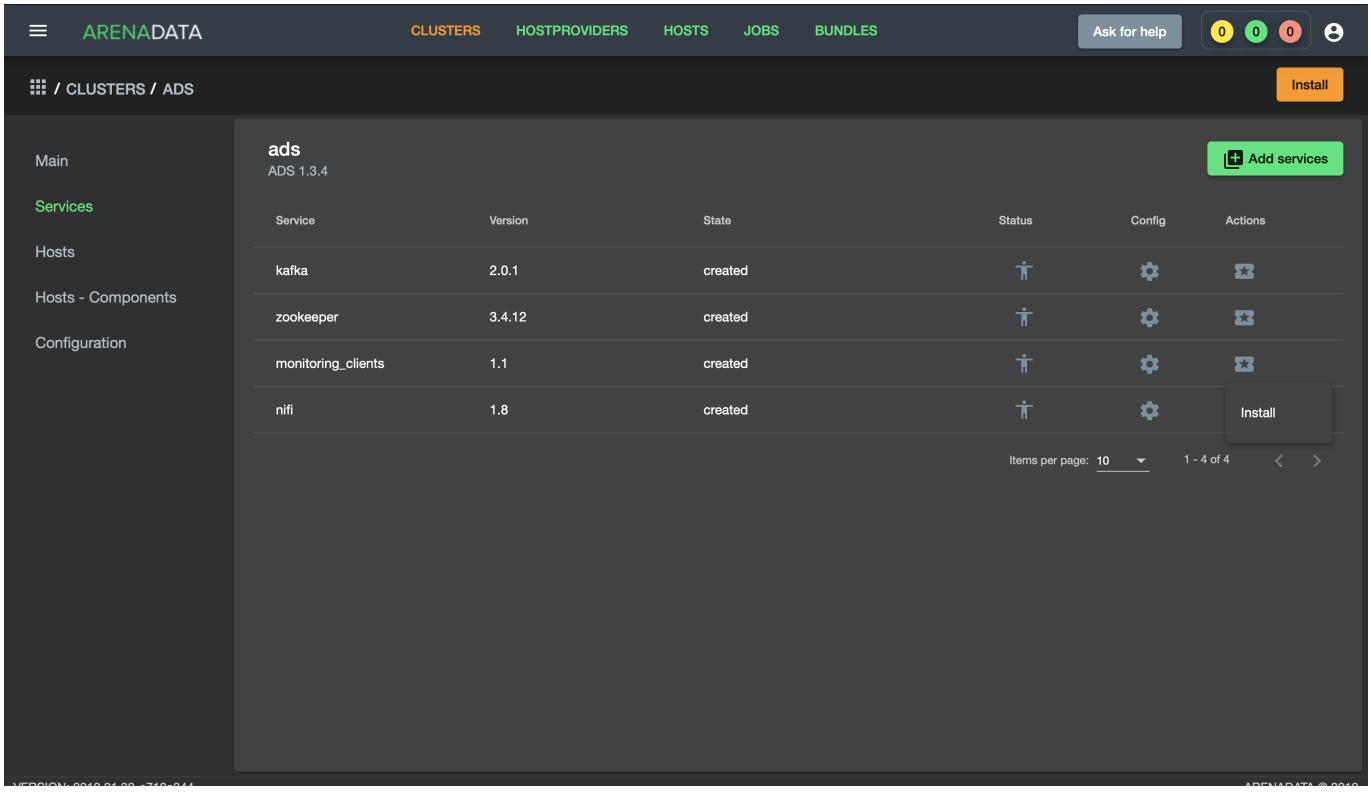


Рис.2.27.: Install monitoring clients

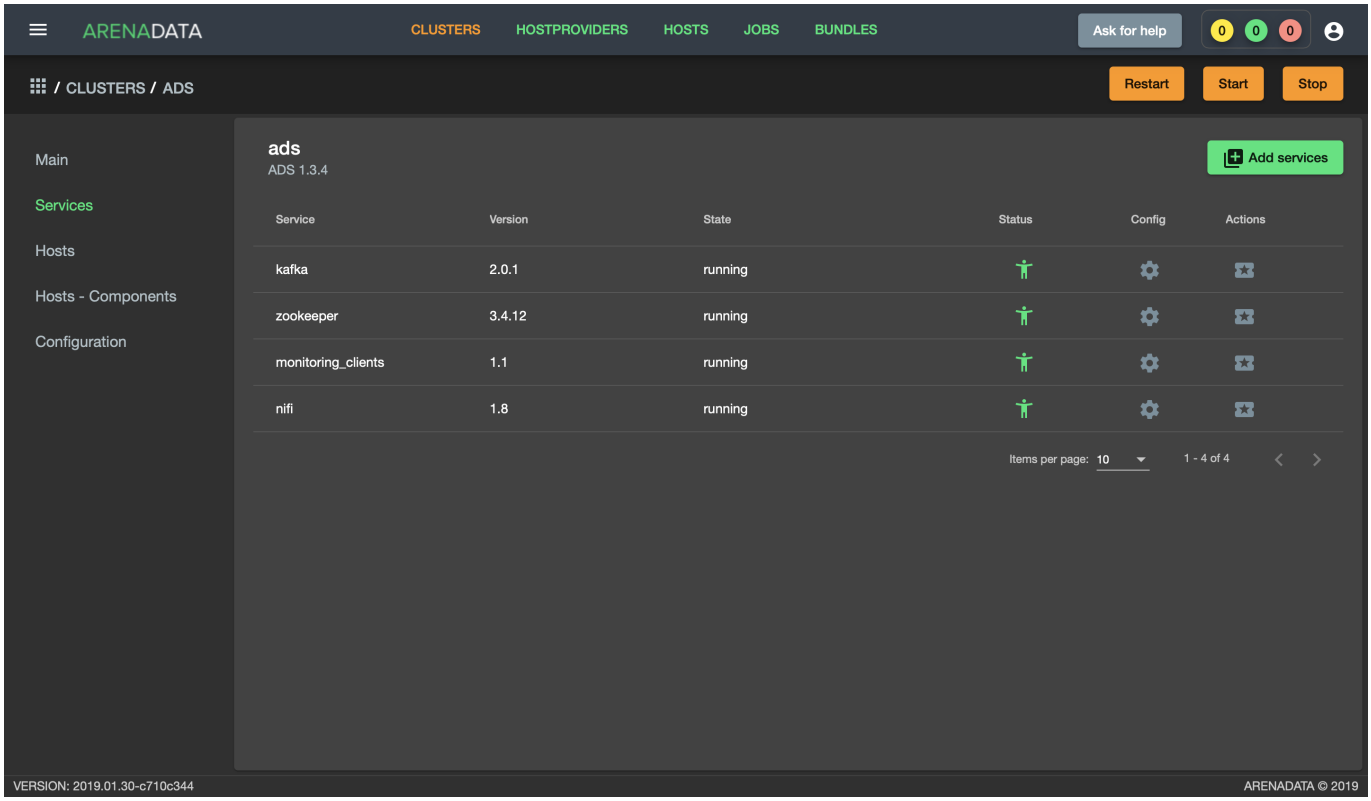


Рис.2.28.: Cluster level operations

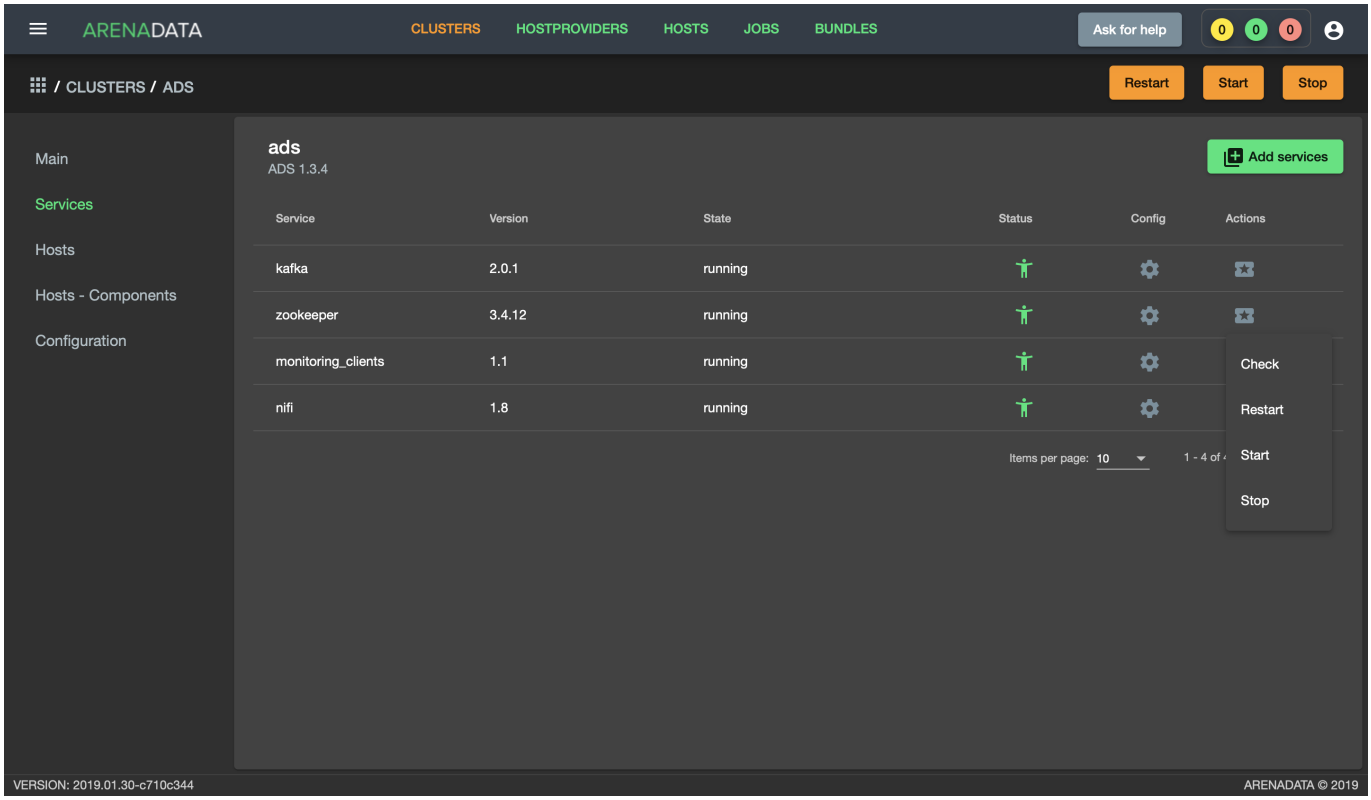


Рис.2.29.: Run the *Zookeeper* service health check

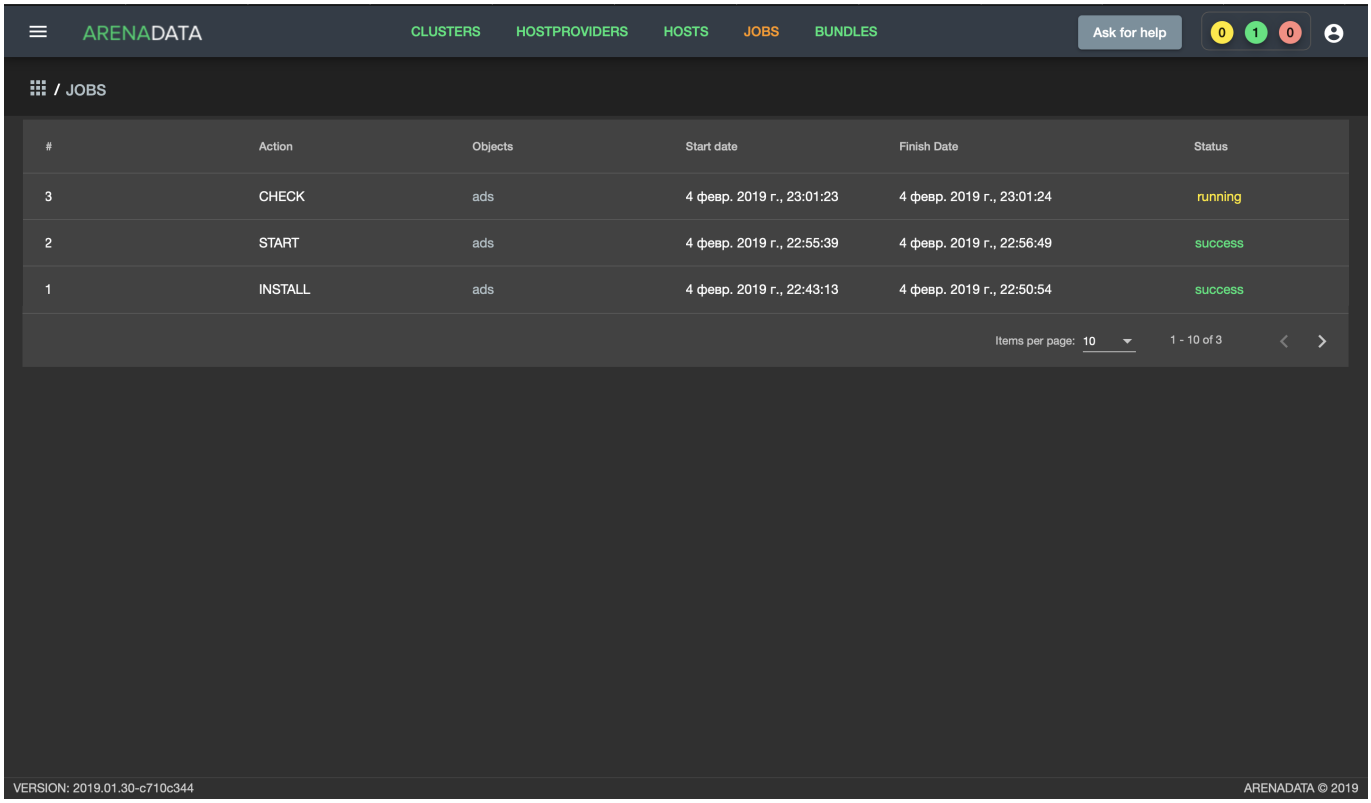


Рис.2.30.: “JOBS” tab

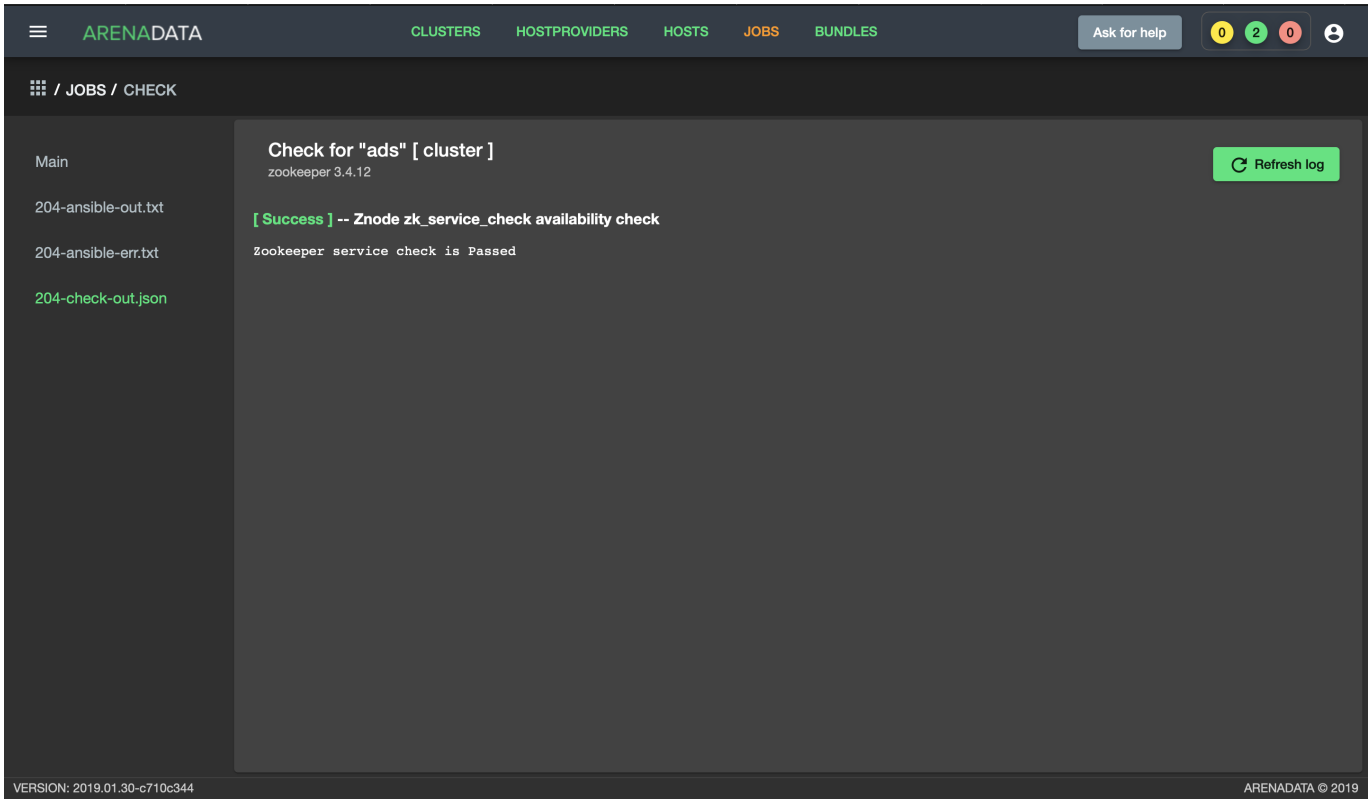


Рис.2.31.: Health check *Zookeeper* service

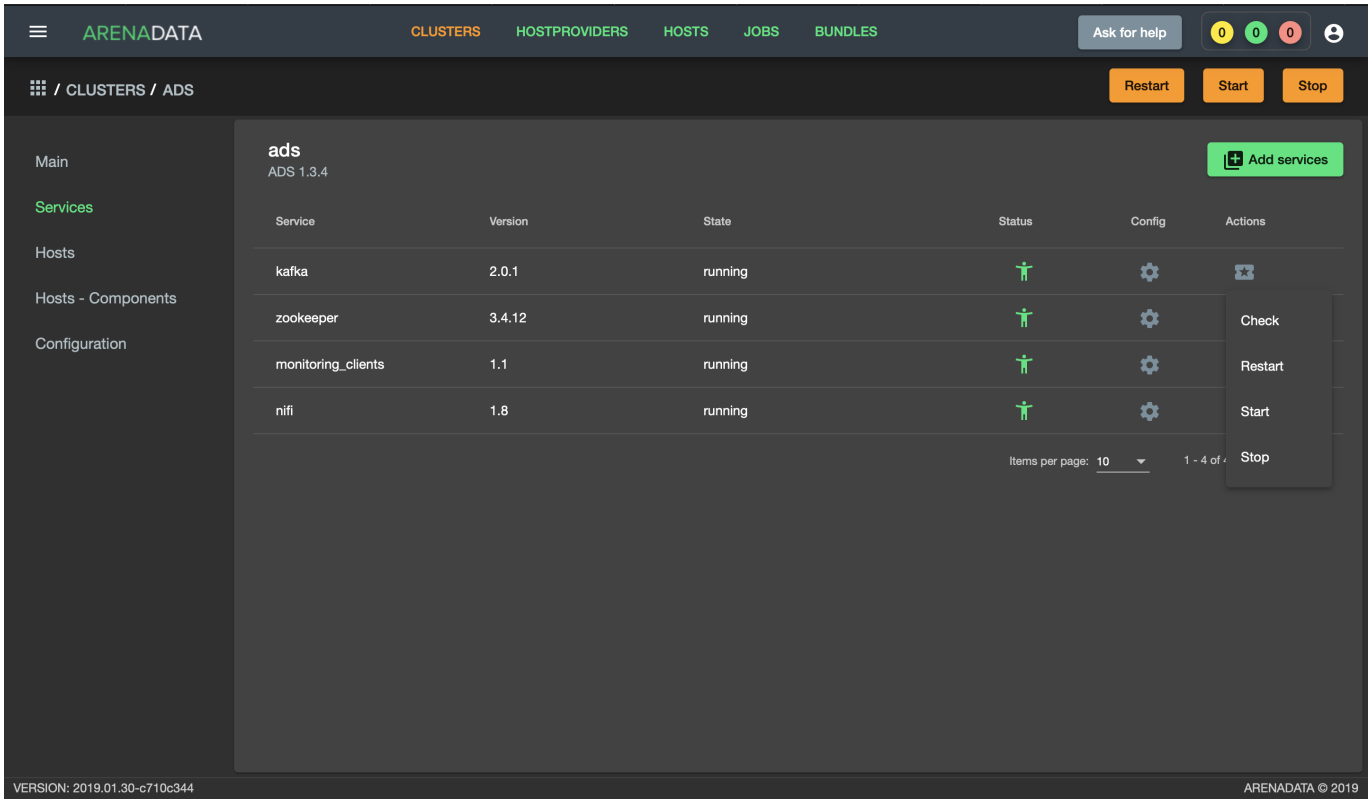


Рис.2.32.: Run the *Kafka* service health check

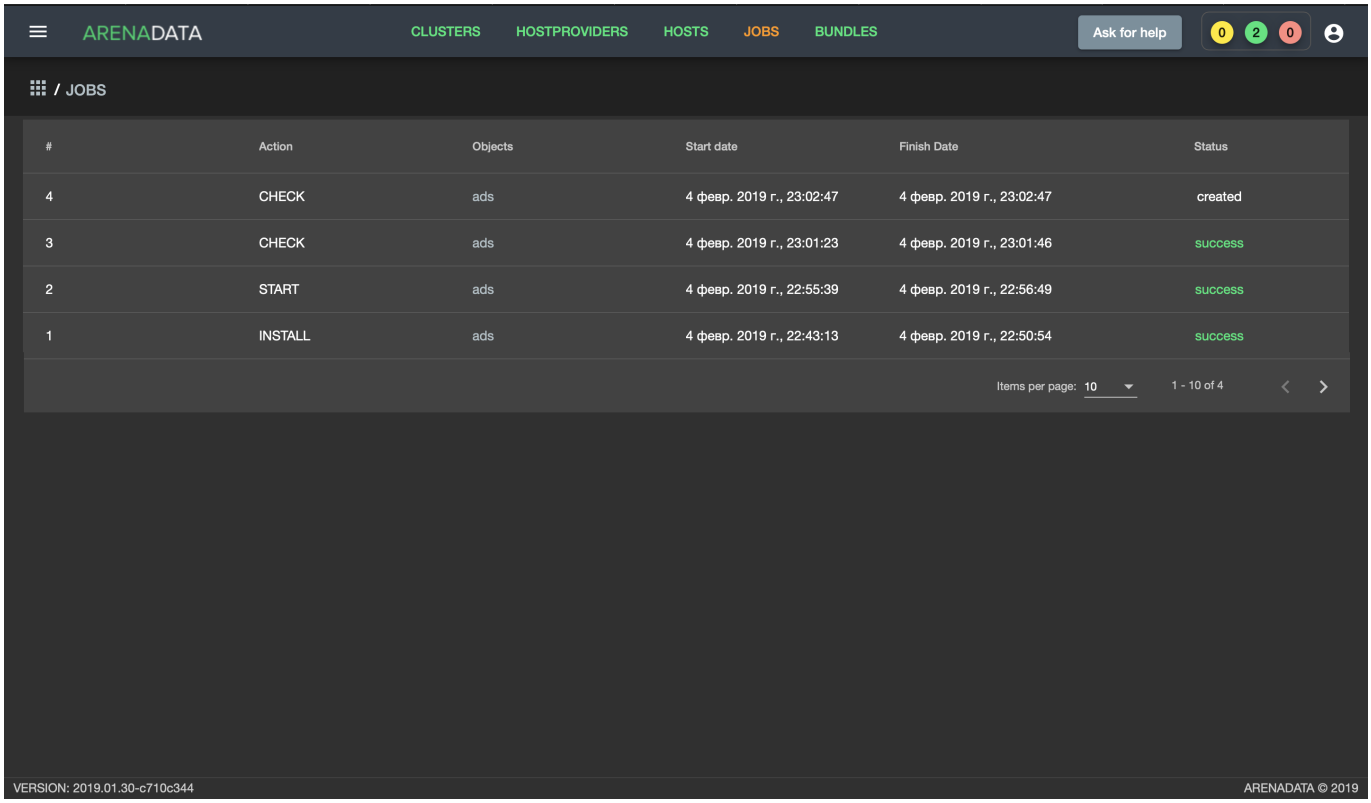


Рис.2.33.: “JOBS” tab

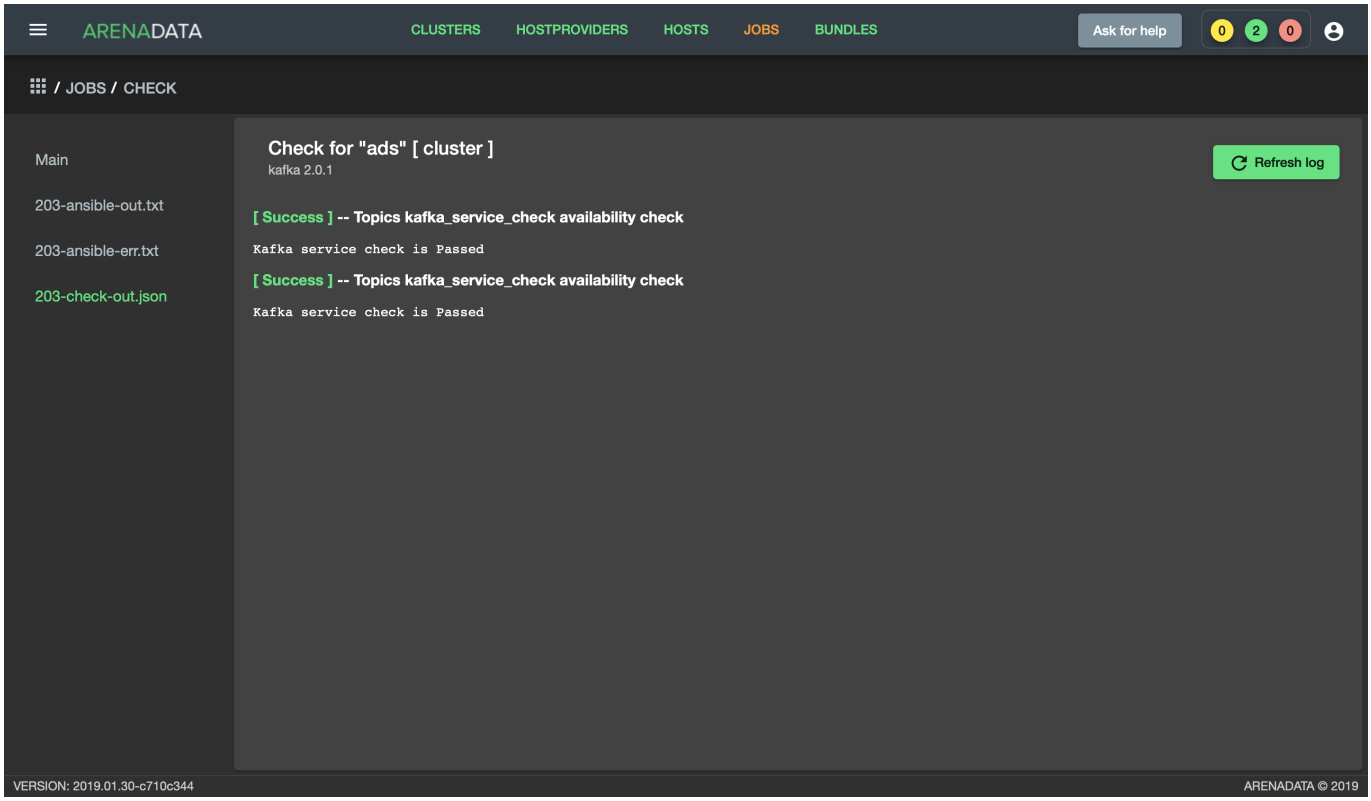


Рис.2.34.: Health check *Kafka* service

Глава 3

Installing ADS cluster with Ambari

Further, the documentation describes the principle of installing the Arenadata Streaming platform using Ambari 2.7.0 and the procedure for preparing to install Arenadata Streaming MPack.

Important: Before proceeding to the following points, you need to download ADS 1.0.0

3.1 Installing Arenadata Streaming with Ambari 2.7.0

Installing the **Ambari 2.7.0** server goes through several steps:

- *Setting up a remote YUM repository*
- *Install Ambari Server*
- *Ambari Server setup*
- *Start Ambari Server*

3.1.1 Setting up a remote YUM repository

Setting up a remote repository is no different from setting up any additional repository *YUM*. To add a repository, you must run the following command as *root*:

```
yum-config-manager --add-repo http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.7.0.0/a
```

3.1.2 Install Ambari Server

The **Ambari** server is installed from the RPM package using the *YUM* command:

```
yum install ambari-server
```

The command sets the **Ambari** server which is the web application server to the port *8080*. Also installed is the server instance **PostgreSQL** on port *5432*.

3.1.3 Ambari Server setup

Ambari server must be configured to work correctly. If the instance **PostgreSQL** is configured on the default port you should run the following command:

```
ambari-server setup
```

During the setup process you must specify or accept the default settings:

- *User account* – you can choose any account to start the Ambari server (it is not necessary to log in from *root*). If the user does not exist, it is automatically created;
- *Java JDK* – to download **Oracle JDK 1.8**, you must enter the value *1* and accept the license **Oracle JDK** to download files from **Oracle**. The installation of **JDK** is done automatically;
- *Database* – select database:

  `Enter advanced database configuration`

  At the command line you must answer *n* or *y*:

  - *n* – for use with Ambari standard built-in database PostgreSQL. By default, the database name PostgreSQL is set to *ambari* and the login/password is *ambari/bigdata*;
  - *y* – if you want to use with Ambari an existing PostgreSQL database, MySQL or Oracle instead of the default one. Next, for the selected database you must specify the connection parameters.

3.1.4 Start Ambari Server

After installing the **Ambari** server start is performed by the command:

```
ambari-server start
```

To check the server status use the command:

```
ambari-server status
```

To stop the server use the command:

```
ambari-server stop
```

Ambari server is available on port *8080*. By default it is set to the following account:

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

Important: It is recommended to change the password after the first login

To enter the web interface **Ambari** you need to specify the server address in the address bar of the browser:

```
http://<адрес сервера>:8080
```

This requests a login and password. After authorization opens the web interface **Ambari** (Pic.3.1.).

3.2 Preparing to install Arenadata Streaming MPack

The main components **ADS** are installed using **Ambari Management Pack** (**MPack**):

- *Download Arenadata Streaming MPack*
- *Import MPack to Ambari stack*

3.2.1 Download Arenadata Streaming MPack

To install **Arenadata Streaming MPack**, you need to download and unpack the following archives in a dedicated place (you should avoid using the */tmp* directory):

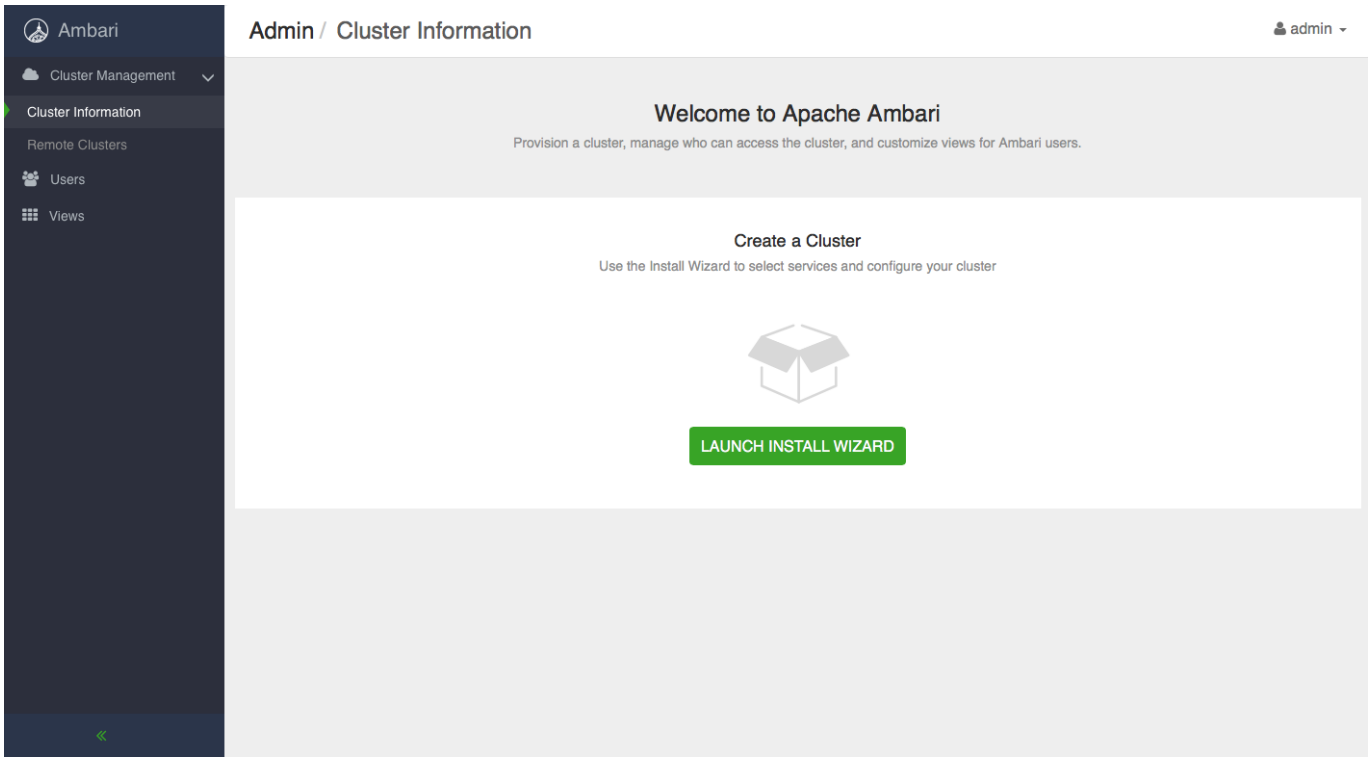


Рис.3.1.: Ambari web interface before cluster setup

- *ads-ambari-mpack-{version}.tar.gz* – a set of service and script configurations for deploying and configuring the platform via **Ambari**.

If archives are loaded into the */tmp* directory then to unpack them in a directory for example */staging* you need to run the following command:

```
tar -xzvf /tmp/ads-ambari-mpack-{version}.tar.gz -C /staging/
```

3.2.2 Import MPack to Ambari stack

To import the **ADS** stack into **Ambari** you need to run the following commands:

1. Save the current folder *Ambari Resources* for possible system recovery:

   ```
   cp -r /var/lib/ambari-server/resources /var/lib/ambari-server/resources.backup
   ```

2. Run the stack import command:

   ```
   ambari-server install-mpack --mpack=/staging/ads-ambari-mpack-{version}.tar.gz --purge
   --verbose
   ```

3. Restart **Ambari**:

   ```
   ambari-server restart
   ```