Installation guide

This guide describes the installation and configuration of Kaa components on a single Linux node and in a cluster environment. This guide does not cover installation and configuration of third party systems (databases, web servers, etc). If this is the first time you use Kaa, we recommend that you start the evaluation using Kaa Sandbox instead of attempting manual installation.

Installing Kaa from stable release

This section describes how to install Kaa by downloading an installation package, as opposed to building Kaa from source.

Supported OS

Kaa supports the following operation system families and provides an installation package for each of them.

- Ubuntu and other Debian systems
- Red Hat/CentOS/Oracle 5 or Red Hat 6 systems

System requirements

To use Kaa, your system must meet the following minimum system requirements.

- 64-bit OS
- 4 Gb RAM

Third party components

Kaa requires the following third party components to be installed and configured.

- Oracle JDK. Kaa has been tested on JDK 7. The framework has not been tested on JDK 8 yet.
- MongoDB 2.6.3. Kaa has been tested on the latest production release of MongoDB.
- PostgreSQL 9.3. Kaa has been tested on the latest production release of PostgreSQL.
- Tomcat 7. Kaa Admin UI has been tested on Tomcat 7.
- Zookeeper 3.4.6. Kaa requires ZooKeeper for coordination of server components.

  NOTE
  Kaa was designed to be independent from NoSQL databases.
  You can implement support of custom NoSQL database with little effort.
  Kaa is going to add support for popular NoSQL databases in the nearest releases.

Installation steps

To install Kaa on either Ubuntu/Debian systems or Red Hat/Red Hat 6/CentOS/Oracle 5 systems, perform the following steps.

Configuration overview

This section provides information on Kaa components configuration file locations and contents.

Each installed Kaa component stores configuration files in the /usr/lib/kaa-$component/conf folder. This folder contains Spring context, logging properties and one or more *.properties configuration files. All the database related properties are stored in the dao.properties file. All component specific properties are stored in the $component-server.properties file. Each property is documented with inline comments. Once reconfigured, a component requires a restart for the changes to take effect. You can find more detailed configuration instructions in the Single node setup and Cluster setup sections.

Database configuration

Kaa uses an SQL(relational) database for metadata and a NoSQL database for endpoint related information.

To configure an SQL database, edit /usr/lib/kaa-$component/conf/dao.properties where $component is *control and operations*.

To configure a NoSQL database, edit /usr/lib/kaa-$component/conf/common-dao-$nosql_db_provider_name.properties where $nosql_db_provider_name is one of supported NoSQL databases.

To switch between NoSQL databases, edit the *nosql_db_provider_name* parameter in /usr/lib/kaa-$component/conf/dao.properties.

SMTP configuration

SMTP properties are used to send emails to newly created users with the information about their passwords, as well as other notifications.

By default, SMTP properties are not configured for Admin UI. To configure Admin UI to target your SMTP server, proceed as follows:

1. Execute the following command to start editing the admin-server.properties file.

   ```
   sudo nano /usr/lib/kaa-admin/conf/admin-server.properties
   ```

2. Specify the following parameters according to your SMTP configuration.

   ```
   # SMTP server host
   smtp_host=localhost
   # SMTP server port
   smtp_port=25
   # Specifies if use SMTP authentication
   smtp_auth=false
   # Specifies SMTP protocol name
   smtp_protocol=smtp
   # Specifies SMTP authentication username
   smtp_username=
   # Specifies SMTP authentication password
   smtp_password=
   ```

3. To activate your changes, restart the Kaa-Admin component as follows:

```
sudo service kaa-admin restart
```

Logging overview

This section provides information on Kaa components log files location and their configuration.

Each installed Kaa component stores log files in the /var/log/kaa folder. This folder normally contains logs from multiple components. Each component stores logs in the kaa-$component-server[.init].log files that are rotated daily. Logging for each component is configured in the /usr/lib/kaa-$component/conf/log4j.xml file. Once reconfigured, a component requires a restart for the changes to take effect.

Configuring firewall port access

The following table provides information on what ports must be open for a single node setup and a cluster setup when the nodes are protected with a firewall.

Port number	**Description**	**Must be open** **for single node setup**	**Must be open** **for cluster setup**
22	SSH port	yes	yes
2181	Zookeeper server port	no	yes
8080	Kaa web admin port	yes	yes
9090	Control server thrift port for internal communication	no	yes
9094	Bootstrap server thrift port for internal communication	no	yes
9889	Bootstrap server public port	yes	yes
9093	Operations server thrift port for internal communication	no	yes
9999	Operations server public HTTP port for client communication	yes	yes
9998	Operations server public HTTP(log poll) port for client communication	yes	yes
9997	Operations server public TCP port for client communication	yes	yes

Validating third party component configuration

Validate Oracle JDK installation

We assume that Oracle JDK is already installed.

You can check if Oracle JDK is installed by executing the following command:

```
javac -version
```

As a result, you will receive the JDK version if it is installed.

Validate Postgresql and MongoDB configuration

We assume that the Postgresql and MongoDB servers are already installed.

The following command checks if the default database ports 5432 (postgresql) and 27017 (mongodb) are listening on network interfaces. If you have other ports configured for this purpose, replace 5432 and/or 27017 with your own ports.

```
sudo netstat -ntlp|grep -e 5432 -e 27017
```

As a result, you will receive a list of interfaces with the corresponding ports.

Specify Postgresql server configuration

The following procedure describes how to set a password for the default postgres user and create the Kaa database. If the password has already been set you can skip this procedure.

1. Connect to the postgresql-server via the psql utility by executing the following command.

   ```
   sudo -u postgres psql
   ```

2. Specify the password for the postgres user.
   The default password is *admin*. If you want to use another password, enter the new password and update the properties in the dao.properties files.

   ```
   postgres=# \password
   Enter new password:
   Enter it again:
   ```

3. Connect to the postgresql-server via the psql utilityas described in step 1 and then create the Kaa database by executing the following command.

   ```
   CREATE DATABASE kaa;
   ```

In case your postgresql server is installed on a separate node, you should bind it to the external ip/host (update the postgresql.conf file) and grant access for client hosts (update the pg_hba.conf file).

Single node setup

This section provides information on Kaa ecosystem setup on a single node. It means that all Kaa and third party components have to be installed on the same node.

The instructions in this section are based on the following assumptions.

- The ZooKeeper server is already installed.
- The external IP address is 192.160.10.172 (in a real-world configuration, it can be a public IP or a real host name).
- MongoDB is used as a NoSQL database for Kaa.

After the installation and validation processes are executed as described in the previous sections, you need to specify the configuration for components as described in this section.

Single node networking configuration

By default, Kaa components are configured to bind onto the local host. Thus, to connect to these components from endpoint applications, you need to reconfigure them for using an external IP or host name, (in our case, it's 192.168.10.172). For this purpose, perform the following steps.

1. Modify the *netty_host* property for the Bootstrap server config by executing the following command.

```
nano /usr/lib/kaa-bootstrap/conf/bootstrap-server.properties
```

The reconfigured *netty_host* to external IP address will look as follows:

bootstrap-server.properties

```
...
# Netty framework configurations. Bootstrap HTTP server implementation based
on Netty framework.
# Netty bootstrap server host (HTTP protocol)
netty_host=192.168.10.172
...
```

2. Modify the *channel_*_host* properties for the Operations server config by executing the following command.

```
nano /usr/lib/kaa-operations/conf/operations-server.properties
```

As a result, the property file will include the following lines.

operations-server.properties

```
...
# Operations channels framework configurations. Operations HTTP server
implementation based on Netty framework.
#  server host (HTTP transport type)
channel_http_host=192.168.10.172
...
#  server host (HTTP Long Polling transport type)
channel_http_lp_host=192.168.10.172
...
#  server host (KAA_TCP transport type)
channel_kaa_tcp_host=192.168.10.172
...
```

3. Modify the *app_base_url* property for Admin UI by executing the following command.

```
nano /usr/lib/kaa-admin/conf/admin-server.properties
```

The reconfigured *app_base_url* with the external IP address will look as follows:

admin-server.properties

```
...
# Web admin application base url
app_base_url=http://192.168.10.172:8080
...
```

4. Restart the Bootstrap server by executing the following command.

```
sudo service kaa-bootstrap restart
```

5. Restart the Operations server by executing the following command.

```
sudo service kaa-operations restart
```

6. Restart the Admin UI server by executing the following command.

```
sudo service kaa-admin restart
```

Single node database configuration

By default, Kaa components are configured in such a way that they are connected to databases on the local host with the default credentials. Execute the following steps to reconfigure the connection.

1. Modify the dao.properties file for the Operations and Control servers by executing the following commands.

```
nano /usr/lib/kaa-operations/conf/dao.properties
nano /usr/lib/kaa-control/conf/dao.properties
```

2. Set the jdbc.host and jdbc.port for the postgresql connection and the servers for the mongodb connection.
3. Replace the default postgresql credentials with your own *jdbc.username* and *jdbc.password*.

In the following example, we left the default values in the database configuration file.

dao.properties

```
...
# List of MongoDB nodes, possible to use multiply servers
# example: 127.0.0.1:27017,127.0.0.2:27017,127.0.0.3:27017
servers=localhost:27017
...
# MongoDB database name
db.name=kaa
...
# specify jdbc database user name
jdbc.username=postgres

# specify jdbc database password
jdbc.password=admin

# specify jdbc database host
jdbc.host=localhost

# specify jdbc database post
jdbc.port=5432

# specify jdbc database url
jdbc.url=jdbc:postgresql://${jdbc.host}:${jdbc.port}/${db.name}
```

Starting Kaa

To activate the introduced changes, you should restart the Kaa components as follows:

1. Restart the Bootstrap server by executing the following command.

```
sudo service kaa-bootstrap restart
```

2. Restart the Operations server by executing the following command.

```
sudo service kaa-operations restart
```

3. Restart the Admin UI server by executing the following command.

```
sudo service kaa-admin restart
```

4. Restart the Control server by executing the following command.

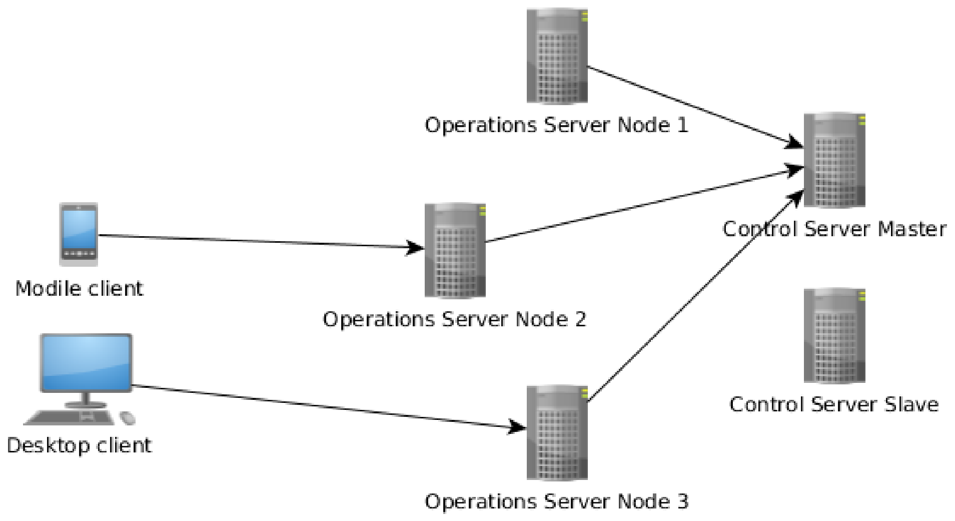
```
sudo service kaa-control restart
```

After restarting all the components, you can access Admin UI at the *app_base_url* specified during the single node networking configuration.

Cluster Setup

This section provides information on the Kaa cluster setup process and Kaa nodes configuration. It is assumed that you are using MongoDB as a NoSQL database for Kaa.

Let's consider a Kaa cluster example with five nodes as in the following picture.



The following table provides general information on Kaa cluster nodes from our example. We use the following range of private IP addresses: 192.168.10.172 - 192.168.10.176. In your own configuration these addresses may be different.

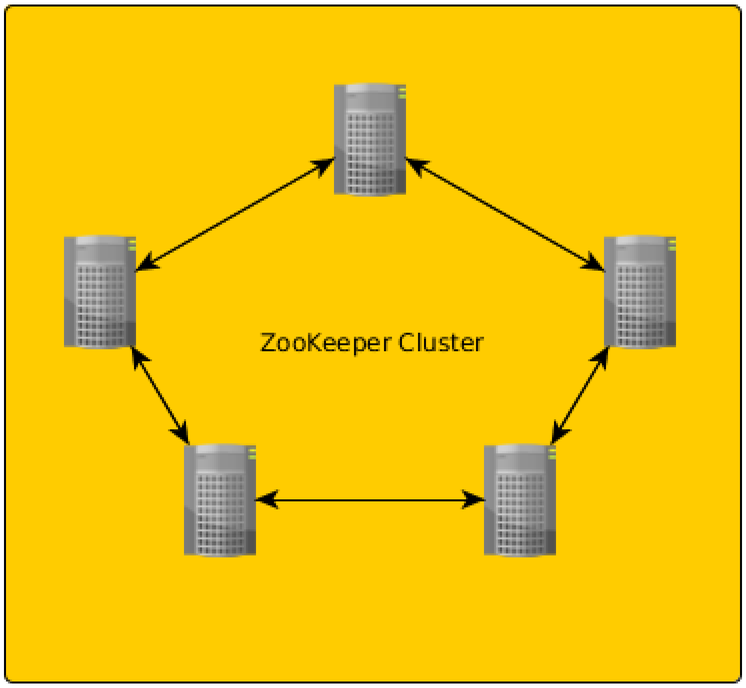
Kaa cluster nodes table

Node name	IP address	Components
Control Server Master	192.168.10.172	Control server, Admin UI
Control Server Slave	192.168.10.173	Control server, Admin UI
Operations Server Node 1	192.168.10.174	Operations server, Bootstrap server
Operations Server Node 2	192.168.10.175	Operations server, Bootstrap server
Operations Server Node 3	192.168.10.176	Operations server, Bootstrap server

You can see that we have dedicated two nodes for the Control server where one is primary (Master) and the other is secondary (Slave). Within the Control server installation, Admin UI component will also be installed.

The Operations and Bootstrap servers will be installed on the remaining three nodes.

ZooKeeper setup



Before starting the Kaa cluster setup, you need to set up the ZooKeeper server on several nodes of your cluster. It is preferable to use an odd number of nodes for this purpose. In our example, it is enough to set up the ZooKeeper server on three nodes, but we will use all five nodes.

1. Install the ZooKeeper on each target node by either using *the package manager* or downloading the ZooKeeper from the home page zookeeper.apache.org.
2. Set each ZooKeeper node's ID in the myid file.
   In our example, we use a sequential number of the node. You can find this file in the /etc/zookeeper/conf directory and set the ID by executing the following command.

   ```
   echo 1 > /etc/zookeeper/conf/myid
   ```

3. Add all ZooKeeper servers to the /etc/zookeeper/conf/zoo.cfg file.
   The following code block provide a list of ZooKeeper servers from our example.

```
server.1=192.168.10.172:2888:3888
server.2=192.168.10.173:2888:3888
server.3=192.168.10.174:2888:3888
server.4=192.168.10.175:2888:3888
server.5=192.168.10.176:2888:3888
```

4. Start the ZooKeeper server by executing the following command on the corresponding node.

   ```
   /usr/share/zookeeper/bin/zkServer.sh start
   ```

5. To view the ZooKeeper server status, execute the following command on the corresponding node.
   If the command is executed successfully, you will receive the ZooKeeper server status.

   ```
   /usr/share/zookeeper/bin/zkServer.sh status
   ```

Networking configuration

By default, Kaa components are configured to bind onto the local host. For a cluster configuration, you need to replace your default host parameters to external ones.

For this purpose, for both Control servers, replace the network parameters for the Control server and Admin UI components with the corresponding IPs: 192.168.10.172/192.168.10.173. To achieve this, proceed as follows:

1. Modify the *thrift_host* property for the Control server config by executing the following command.

   ```
   nano /usr/lib/kaa-control/conf/control-server.properties
   ```

2. Modify the *app_base_url* and *control_thrift_host* properties for Admin UI config by executing the following command.

   ```
   nano /usr/lib/kaa-admin/conf/admin-server.properties
   ```

For the remaining three nodes, replace the network parameters for the Operations and Bootstrap components with the corresponding IPs: 192.168.10.174/192.168.10.175/192.168.10.176. To achieve this, proceed as follows:

1. Modify ***thrift_host*** and *channel_*_host* properties for the Operations server config by executing the following command.

   ```
   nano /usr/lib/kaa-operations/conf/operations-server.properties
   ```

2. Modify the *thrift_host* and *netty_host* properties for the Bootstrap server config by executing the following command.

   ```
   nano /usr/lib/kaa-bootstrap/conf/bootstrap-server.properties
   ```

Database configuration

This section describes the databases configuration for Kaa components. The section does not describe the databases installation procedure. In our example scenario, we installed the MongoDB and Postgresql databases on the same Kaa cluster nodes. For the Postgresql database, the configuration procedure is the same as that for the single node installation, and you can use the procedure described in the Single node database configuration section.

For the MongoDB we decided to use the MongoDB Sharded Cluster instead of a single node database. Therefore, we installed MongoDB Config Servers on the Control Server Master, Control Server Slave, and Operations Server Node 1 nodes (one server per node) and MongoDB Routing Server on each node of the Kaa cluster. You can find more detailed information on shared clusters on the official site: Sharded Cluster.

To set the MongoDB configuration, execute the following steps.

1. On each Control server node, modify the *servers* property. In our scenario, it will be the external IP and port of Control Servers (192.168.10.172:27017,192.168.10.173:27017).

   ```
   nano /usr/lib/kaa-control/conf/dao.properties
   ```

2. Perform the same action on each Operations server node (192.168.10.174:27017,192.168.10.175:27017,192.168.10.176:27017).

   ```
   nano /usr/lib/kaa-operations/conf/dao.properties
   ```

Starting Kaa cluster

After all the configuration processes are complete, start/restart the installed Kaa component (see Kaa cluster nodes table) on each Kaa cluster node.

To activate your changes, restart the Kaa-Admin component as follows:

```
sudo service kaa-admin restart
```

Further reading

Use the following guides and references to make the most of Kaa.

Guide	What it is for
Building Kaa from source	Use this page to learn how to build Kaa from its source code (an alternative way of Kaa installation).
Installing Kaa flume agents	Use this page to learn how to install Kaa flume agents which provide advanced features for integration between Kaa and Hadoop as described in Flume log appender.
Administration UI guide	Use this guide to start working with the Kaa web UI.
Design reference	Use this reference to learn about features and capabilities of Kaa.
Programming guide	Use this guide to create your own Kaa applications.