Hortonworks DataFlow

Command Line Installation

docs.cloudera.com

Hortonworks DataFlow: Command Line Installation

Copyright © 2012-2017 Hortonworks, Inc. Some rights reserved.

Table of Contents

List of Tables

1. Installing NiFi

- Installing NiFi [1]

- Starting NiFi [1]

- Launching the UI [2]

1.1. Installing NiFi

To install NiFi:

1. Extract the files to the location from which you want to run the application.

For information on how to configure NiFi (for instance, security, data storage, or the port on which NiFi is running), see the System Administrator's Guide.

1.1.1. Installing as a Service

You can also install NiFi as a service.

1. Navigate to the NiFi installation directory.

2. Enter:

```
bin/nifi.sh install
```

3. You can specify a custom name by specifying that name during your install command.

For example, to install NiFi as a service with the name `dataflow`, enter:

```
bin/nifi.sh install dataflow
```

The NiFi service is installed with the default name `nifi`.

1.2. Starting NiFi

Once you have downloaded and installed HDF, you can start it by using the mechanism appropriate for your operating system.

If you are running NiFi on a Linux or Mac OSX operating system, you can start NiFi to run in the foreground, background, or as a service.

Launching NiFi in the foreground:

1. From a terminal windows, navigate to the NiFi installation directory.

2. Enter:

```
bin/nifi.sh run
```

Launching NiFi in the background:

1. From a terminal window, navigate to the NiFi installation directory.

2. Enter:

```
bin/nifi.sh start
```

Launching NiFi as a service:

1. From a terminal window, enter:

```
sudo service nifi start
```

1.3. Launching the UI

Once you have started NiFi, you can bring up the User Interface (UI) to create and monitor dataflows. Open a web browser and navigate to `http://localhost:8080/nifi`.

The toolbars at the top of the UI are very important to create your first dataflow:
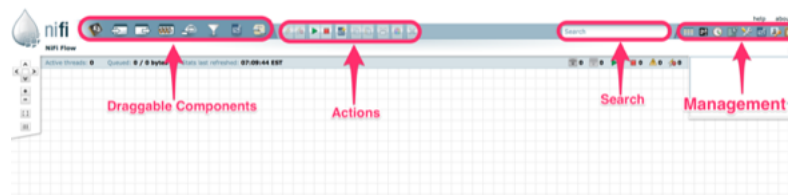
2. Installing Apache ZooKeeper

This section describes installing and testing Apache ZooKeeper, a centralized tool for providing services to highly distributed systems.

> **Note**
>
> HDFS and YARN depend on ZooKeeper, so install ZooKeeper first.

1. Install the ZooKeeper Package [3]

2. Securing ZooKeeper with Kerberos (optional) [3]

3. Securing ZooKeeper Access [4]

4. Set Directories and Permissions [6]

5. Set Up the Configuration Files [7]

6. Start ZooKeeper [8]

2.1. Install the ZooKeeper Package

> **Note**
>
> In a production environment, Hortonworks recommends installing ZooKeeper server on three (or a higher odd number) nodes to ensure that ZooKeeper service is available.

On all nodes of the cluster that you have identified as ZooKeeper servers, type:

• For RHEL/CentOS/Oracle Linux

```
yum install zookeeper-server
```

• For SLES

```
zypper install zookeeper
```

• For Ubuntu and Debian:

```
apt-get install zookeeper
```

> **Note**
>
> Grant the zookeeper user shell access on Ubuntu and Debian.
>
> ```
> usermod -s /bin/bash zookeeper
> ```

2.2. Securing ZooKeeper with Kerberos (optional)

(Optional) To secure ZooKeeper with Kerberos, perform the following steps on the host that runs KDC (Kerberos Key Distribution Center):

1. Start the kadmin.local utility:

   ```
   /usr/sbin/kadmin.local
   ```

2. Create a principal for ZooKeeper:

   ```
   sudo kadmin.local -q 'addprinc zookeeper/
   <ZOOKEEPER_HOSTNAME>@STORM.EXAMPLE.COM'
   ```

3. Create a keytab for ZooKeeper:

   ```
   sudo kadmin.local -q "ktadd -k /tmp/zk.keytab zookeeper/
   <ZOOKEEPER_HOSTNAME>@STORM.EXAMPLE.COM"
   ```

4. Copy the keytab to all ZooKeeper nodes in the cluster.

   > **Note**
   >
   > Verify that only the ZooKeeper and Storm operating system users can access the ZooKeeper keytab.

5. Administrators must add the following properties to the zoo.cfg configuration file located at `/etc/zookeeper/conf`:

   ```
   authProvider.1 = org.apache.zookeeper.server.auth.SASLAuthenticationProvider
   kerberos.removeHostFromPrincipal = true
   kerberos.removeRealmFromPrincipal = true
   ```

   > **Note**
   >
   > Grant the zookeeper user shell access on Ubuntu and Debian.
   >
   > ```
   > usermod -s /bin/bash zookeeper
   > ```

2.3. Securing ZooKeeper Access

The default value of `yarn.resourcemanager.zk-acl` allows anyone to have full access to the znode. Hortonworks recommends that you modify this permission to restrict access by performing the steps in the following sections.

- ZooKeeper Configuration [4]

- YARN Configuration [5]

- HDFS Configuration [6]

2.3.1. ZooKeeper Configuration

> **Note**
>
> The steps in this section only need to be performed once for the HDP cluster. If this task has been done to secure HBase for example, then there is no need

to repeat these ZooKeeper steps if the YARN cluster uses the same ZooKeeper server.

1. Create a keytab for ZooKeeper called `zookeeper.service.keytab` and save it to `/etc/security/keytabs`.

```
sudo kadmin.local -q "ktadd -k /tmp/zk.keytab zookeeper/
 <ZOOKEEPER_HOSTNAME>@STORM.EXAMPLE.COM"
```

2. Add the following to the `zoo.cfg` file:

```
authProvider.1=org.apache.zookeeper.server.auth.SASLAuthenticationProvider
jaasLoginRenew=3600000
kerberos.removeHostFromPrincipal=true
kerberos.removeRealmFromPrincipal=true
```

3. Create the `zookeeper_client_jaas.conf` file.

```
Client {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=false
useTicketCache=true;
};
```

4. Create the `zookeeper_jaas.conf` file.

```
Server {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
storeKey=true
useTicketCache=false
keyTab="$PATH_TO_ZOOKEEPER_KEYTAB"
(such as"/etc/security/keytabs/zookeeper.service.keytab")
principal="zookeeper/$HOST";
(such as "zookeeper/xuan-sec-yarn-ha-2.novalocal@SCL42.HORTONWORKS.COM";)
};
```

5. Add the following information to `zookeeper-env-sh`:

```
export CLIENT_JVMFLAGS="-Djava.security.auth.login.config=/etc/zookeeper/
conf/zookeeper_client_jaas.conf"
export SERVER_JVMFLAGS="-Xmx1024m
-Djava.security.auth.login.config=/etc/zookeeper/conf/zookeeper_jaas.conf"
```

2.3.2. YARN Configuration

Note

The following steps must be performed on all nodes that launch the ResourceManager.

1. Create a new configuration file called `yarn_jaas.conf` in the directory that contains the Hadoop Core configurations (typically, `/etc/hadoop/conf`).

```
Client {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
storeKey=true
```

```
useTicketCache=false
keyTab="$PATH_TO_RM_KEYTAB"
(such as "/etc/security/keytabs/rm.service.keytab")
principal="rm/$HOST";
(such as "rm/xuan-sec-yarn-ha-1.novalocal@EXAMPLE.COM";)
};
```

2. Add a new property to the `yarn-site.xml` file.

```
<property>
<name>yarn.resourcemanager.zk-acl</name>
<value>sasl:rm:rwcda</value>
</property>
```

> **Note**
>
> Because `yarn-resourcemanager.zk-acl` is set to `sasl`, you do
> not need to set any value for `yarn.resourcemanager.zk-auth`.
> Setting the value to sasl also means that you cannot run the command
> `addauth<scheme><auth>` in the zkclient CLI.

3. Add a new YARN_OPTS to the `yarn-env.sh` file and make sure this YARN_OPTS is
picked up when you start your ResourceManagers.

```
YARN_OPTS="$YARN_OPTS -Dzookeeper.sasl.client=true
-Dzookeeper.sasl.client.username=zookeeper
-Djava.security.auth.login.config=/etc/hadoop/conf/yarn_jaas.conf
-Dzookeeper.sasl.clientconfig=Client"
```

2.3.3. HDFS Configuration

1. In the `hdfs-site.xml` file, set the following property, for security of ZooKeeper based
fail-over controller. when NameNode HA is enabled:

```
<property>
    <name>ha.zookeeper.acl</name>
    <value>sasl:nn:rwcda</value>
</property>
```

2.4. Set Directories and Permissions

Create directories and configure ownership and permissions on the appropriate hosts as
described below. If any of these directories already exist, we recommend deleting and
recreating them.

Hortonworks provides a set of configuration files that represent a working ZooKeeper
configuration. (See Download Companion Files.) You can use these files as a reference
point, however, you need to modify them to match your own cluster environment.

If you choose to use the provided configuration files to set up your ZooKeeper
environment, complete the following steps to create the appropriate directories.

1. Execute the following commands on all nodes:

```
mkdir -p $ZOOKEEPER_LOG_DIR;
```

```
chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_LOG_DIR;
chmod -R 755 $ZOOKEEPER_LOG_DIR;

mkdir -p $ZOOKEEPER_PID_DIR;
chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_PID_DIR;
chmod -R 755 $ZOOKEEPER_PID_DIR;

mkdir -p $ZOOKEEPER_DATA_DIR;
chmod -R 755 $ZOOKEEPER_DATA_DIR;
chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_DATA_DIR
```

where:

- $ZOOKEEPER_USER is the user owning the ZooKeeper services. For example, zookeeper.

- $ZOOKEEPER_LOG_DIR is the directory to store the ZooKeeper logs. For example, `/var/log/zookeeper`.

- $ZOOKEEPER_PID_DIR is the directory to store the ZooKeeper process ID. For example, `/var/run/zookeeper`.

- $ZOOKEEPER_DATA_DIR is the directory where ZooKeeper stores data. For example, `/grid/hadoop/zookeeper/data`.

2. Initialize the ZooKeeper data directories with the 'myid' file. Create one file per ZooKeeper server, and put the number of that server in each file:

   `vi $ZOOKEEPER_DATA_DIR/myid`

   - In the myid file on the first server, enter the corresponding number: **1**

   - In the myid file on the second server, enter the corresponding number: **2**

   - In the myid file on the third server, enter the corresponding number: **3**

2.5. Set Up the Configuration Files

You must set up several configuration files for ZooKeeper. Hortonworks provides a set of configuration files that represent a working ZooKeeper configuration. (See Download Companion Files. You can use these files as a reference point, however, you need to modify them to match your own cluster environment.

If you choose to use the provided configuration files to set up your ZooKeeper environment, complete the following steps:

1. Extract the ZooKeeper configuration files to a temporary directory.

   The files are located in the `configuration_files/zookeeper` directories where you decompressed the companion files.

2. Modify the configuration files.

   In the respective temporary directories, locate the `zookeeper-env.sh` file and modify the properties based on your environment including the JDK version you downloaded.

3. Edit the `zookeeper-env.sh` file to match the Java home directory, ZooKeeper log directory, ZooKeeper PID directory in your cluster environment and the directories you set up above.

See below for an example configuration:

4.
```
export JAVA_HOME=/usr/jdk64/jdk1.8.0_40
export ZOOKEEPER_HOME=/usr/hdf/current/zookeeper-server
export ZOO_LOG_DIR=/var/log/zookeeper
export ZOOPIDFILE=/var/run/zookeeper/zookeeper_server.pid
export SERVER_JVMFLAGS=-Xmx1024m
export JAVA=$JAVA_HOME/bin/java
export CLASSPATH=$CLASSPATH:$ZOOKEEPER_HOME/*
```

5. Edit the zoo.cfg file to match your cluster environment. Below is an example of a typical `zoo.cfs` file:

```
dataDir=$zk.data.directory.path
server.1=$zk.server1.full.hostname:2888:3888
server.2=$zk.server2.full.hostname:2888:3888
server.3=$zk.server3.full.hostname:2888:3888
```

6. Copy the configuration files.

• On all hosts create the config directory:

```
rm -r $ZOOKEEPER_CONF_DIR ;
mkdir -p $ZOOKEEPER_CONF_DIR ;
```

• Copy all the ZooKeeper configuration files to the $ZOOKEEPER_CONF_DIR directory.

• Set appropriate permissions:

```
chmod a+x $ZOOKEEPER_CONF_DIR/;
chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_CONF_DIR/../;
chmod -R 755 $ZOOKEEPER_CONF_DIR/../
```

Note:

• $ZOOKEEPER_CONF_DIR is the directory to store the ZooKeeper configuration files. For example, `/etc/zookeeper/conf`.

• $ZOOKEEPER_USER is the user owning the ZooKeeper services. For example, zookeeper.

2.6. Start ZooKeeper

To install and configure HBase and other Hadoop ecosystem components, you must start the ZooKeeper service and the ZKFC:

```
sudo -E -u zookeeper bash -c "export ZOOCFGDIR=$ZOOKEEPER_CONF_DIR ; export
 ZOOCFG=zoo.cfg;
      source $ZOOKEEPER_CONF_DIR/zookeeper-env.sh ; $ZOOKEEPER_HOME/bin/
zkServer.sh
    start"
```

For example:

```
su - zookeeper -c "export ZOOCFGDIR=/usr/hdf/current/zookeeper-server/
conf ; export ZOOCFG=zoo.cfg; source /usr/hdf/current/zookeeper-server/conf/
zookeeper-env.sh ; /usr/hdf/current/zookeeper-server/bin/zkServer.sh start"
```

```
su -l hdfs -c "/usr/hdf/current/hadoop-hdfs-namenode/../hadoop/sbin/hadoop-
daemon.sh start zkfc"
```

3. Installing and Configuring Apache Kafka

This section describes how to install Apache Kafka, a high-throughput messaging system with publish-and-subscribe semantics. Kafka is often used in place of traditional message brokers like JMS and AMQP because of its higher throughput, replication, and fault tolerance.

To install Apache Kafka, complete the following instructions:

1. Install Kafka [10]

2. Configure Kafka [11]

3. Validate Kafka [12]

3.1. Install Kafka

Prerequisites and Considerations

When installing Kafka, note the following prerequisites and considerations:

• Administrators must use Apache ZooKeeper to manage Kafka for an HDP cluster. Verify that you have successfully installed ZooKeeper before installing and configuring Kafka.

• Kafka does not currently support user authentication and authorization.

• The following underlying file systems are supported for use with Kafka:

  • EXT4: recommended

  • EXT3: supported

> ### Caution
>
> Encrypted file systems such as SafenetFS are not supported for Kafka. Index file corruption can occur.

Installation

Install the Kafka RPMs or packages by completing the following steps.

> ### Note
>
> Hortonworks recommends avoiding using multiple brokers in a single node for Kafka. However, if you need to install a multi-node cluster, use the following instructions to install Kafka on another machine, make sure each `broker.id` is unique on the cluster, and ensure that `zookeeper.connect` is the same for all brokers.

1. Run the following command on each client cluster node and gateway node:

   - For RHEL/CentOS/Oracle Linux

     ```
     yum install kafka
     ```

   - For SLES

     ```
     zypper install kafka
     ```

   - For Ubuntu and Debian

     ```
     apt-get install kafka
     ```

2. Check the JAVA_HOME environment variable. If it has not yet been set, add the following to the PATH variable:

   ```
   export JAVA_HOME=<path of installed jdk version folder>
   ```

3.2. Configure Kafka

Use the following procedure to configure Kafka.

1. By default, Kafka is installed at `/usr/hdf/current/kafka-broker`.

2. Verify the values of the following configuration properties in the `server.properties` file:

Table 3.1. Kafka Configuration Properties

Kafka Configuration Property	Description
log.dirs	Comma-separated list of directories where Kafka log files are stored. The default value is /kafka-logs.
zookeeper.connect	The hostname or IP address of the host running ZooKeeper and the port to which ZooKeeper listens. The default value is localhost:2181.
log.retention.hours	The number of hours to wait before a Kafka log file is eligible for deletion. The default value is 168 hours (7 days).

Listeners	listener - Comma-separated list of URIs on which we listen and their protocols.
	
	Specify hostname as 0.0.0.0 to bind to all interfaces.
	
	Leave hostname empty to bind to default interface.
	
	Examples of legal listener lists:
	
	`PLAINTEXT://myhost:9092,SASL_PLAINTEXT://:9091`
	
	`PLAINTEXT://0.0.0.0:9092, SASL_PLAINTEXT://localhost:9093`

File descriptor	Kafka uses a very large number of files and also large number of sockets to communicate with clients. We suggest the following values: • The maximum number of open files. Recommended value: 128000

Kafka Configuration Property	Description
	• The maximum number of processes. Recommended value: 65536

3.3. Validate Kafka

Use the following procedure to verify the Kafka installation and configuration.

Before you begin:

• Verify that ZooKeeper is running before starting Kafka and validating the installation.

• Set KAFKA_HOME to /usr/hdf/current/kafka-broker.

1. Start the Kafka service using user kafka:

   ```
   su kafka -c "KAFKA_HOME/bin/kafka start"
   ```

2. Create a Kafka topic with the name "test" that has a replication factor of 1 and 1 partition.

   ```
   bin/kafka-topics.sh --zookeeper localhost:2181 --create --topic
   test --replication-factor 1 --partitions 1
   ```

   After running this command, you should see the following output:

   ```
   Created topic "test"
   ```

   > **Note**
   >
   > The value of `--replication-factor` must be less then or equal to the number of Kafka brokers in the cluster. Otherwise an error occurs. Usually the replication-factor equals the number of Kafka brokers in the cluster.

3. Start a command line Kafka console producer that you can use to send messages. You can type your message once the process is started.

   ```
   <KAFKA_HOME>/bin/kafka-console-producer.sh --broker-list
   localhost:9092 --topic test
   ```

   You should see your test message, for example:

   ```
   This is a message.
   ```

   > **Note**
   >
   > To return to the command prompt after sending the test message, type Ctrl + C.

4. Start a command line kafka consumer that you can use to read the messages sent by the producer.

   ```
   <KAFKA_HOME>/bin/kafka-console-consumer.sh --zookeeper
   localhost:2181 --topic test --from-beginning
   ```

4. Installing and Configuring Apache Storm

This section describes how to install and configure Apache Storm, a distributed, fault-tolerant, and high-performance real time computation tool used to stream data into Hadoop.

To install Apache Storm, complete the following instructions.

1. Install the Storm Package [13]

2. Configure Storm [14]

3. Configure a Process Controller [14]

4. (Optional) Configure Kerberos Authentication for Storm [16]

5. (Optional) Configuring Authorization for Storm [19]

6. Validate the Installation [21]

4.1. Install the Storm Package

Prerequisite: Storm requires version 2.6 or higher of the default system Python interpreter.

1. To install the Storm RPMs, run the following command on each client cluster node and gateway node:

   • For RHEL/CentOS/Oracle Linux:

   ```
   yum install storm
   ```

   • For SLES:

   ```
   zypper install storm
   ```

   • For Ubuntu and Debian:

   ```
   apt-get install storm
   ```

   ⚠️ **Important**

   Ubuntu and Debian users must manually create the `/etc/storm/conf` directory and the storm.yaml file that resides there.

2. Run the following command to create the conf directory:

   ```
   sudo mkdir -p /etc/storm/conf
   ```

3. Run the following command to create the `storm.yaml` file:

   ```
   sudo touch /etc/storm/conf/storm.yaml
   ```

4.2. Configure Storm

In Storm 1.0 , Java package naming moved from backtype.storm to org.apache.storm.

If you intend to run any topologies that used to run previous versions of Storm in Storm 1.0, you can do so by using one of two options:

• You can rebuild the topology by renaming the imports of the backtype.storm package to org.apache in all of your topology code.

or

• You can add config `Client.jartransformer.class = org.apache.storm.hack.StormShadeTransformer` to `storm.yaml`.

Either of these configurations allows the storm jar to transform all of the `backtype.storm` imports in your topology to `org.apache.storm`.

Use the following procedure to configure Storm:

1. Add the following properties to the `/etc/storm/conf/storm.yaml` file, substituting your own list of hostnames and ports:

```
storm.zookeeper.servers: [<zookeeper-servers>]
nimbus.seeds: [<nimbus-hostnames>]
storm.local.dir: $STORM_LOCAL_DIR
logviewer.port: 8081
```

   where:

   <zookeeper-servers> is a comma-separated list of ZooKeeper servers.

   <nimbus-hostnames> is a comma-separated list of hosts where the Storm Nimbus server is started.

   $STORM_LOCAL_DIR should be `/storm/local`, and it must exist on all Storm nodes.

   For example:

```
storm.zookeeper.servers: ["host1:port1", "host2:port2", "host3:port3"]
nimbus.seeds: ["host1:port1", "host2:port2"]
storm.local.dir: /mnt/storm
logviewer.port: 8081
```

2. Run the following commands:

   ```
   chown -R storm:storm $STORM_LOCAL_DIR
   ```

   ```
   chmod -R 755 $STORM_LOCAL_DIR
   ```

4.3. Configure a Process Controller

Storm administrators should install and configure a process controller to monitor and run Apache Storm under supervision. Storm is a fail-fast application, meaning that it is designed

to fail under certain circumstances, such as a runtime exception or a break in network connectivity. Without a watchdog process, these events can quickly take down an entire Storm cluster in production. A watchdog process prevents this by monitoring for failed Storm processes and restarting them when necessary.

This section describes how to configure supervisord to manage the Storm processes, but administrators may use another process controller of their choice, such as monitor daemontools.
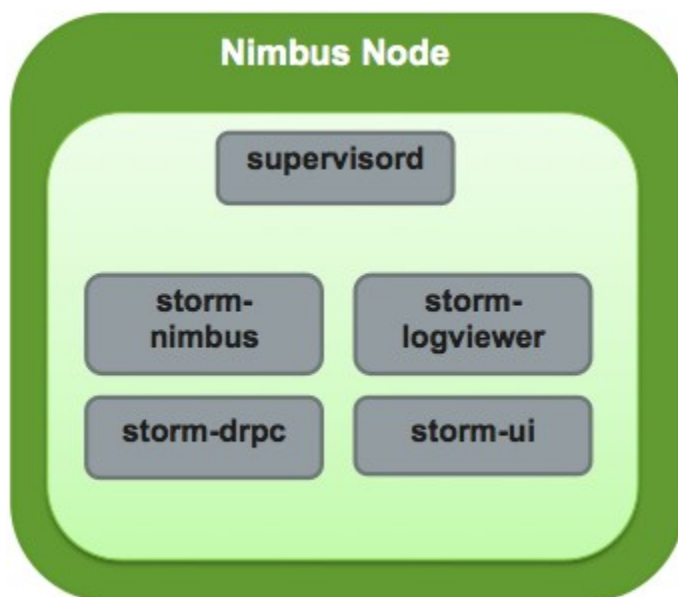
Add the following stanzas to the `/etc/supervisord.conf` to configure Supervisor to start and stop all of the Storm daemons:

```
...
[program:storm-nimbus]
command=storm nimbus
directory=/home/storm
autorestart=true
user=storm

[program:storm-supervisor]
command=storm supervisor
directory=/home/storm
autorestart=true
user=storm

[program:storm-ui]
command=storm ui
directory=/home/storm
autorestart=true
user=storm

[program:storm-logviewer]
command=storm logviewer
autorestart=true
user=storm
```

```
[program:storm-drpc]
command=storm drpc
directory=/home/storm
autorestart=true
user=storm
```

4.4. (Optional) Configure Kerberos Authentication for Storm

Storm supports authentication using several models. This topic describes how to configure your Storm installation to use Kerberos authentication. At a high level, administrators must perform the tasks in this section.

Create Keytabs and Principals for Storm Daemons

Storm requires a principal and keytab when using Kerberos for authentication. A principal name in a given realm consists of a primary name and an instance name, the FQDN of the host that runs the service, in this case Storm. As services do not log in with a password to acquire their tickets, the authentication credentials for the service principal are stored in a keytab file, which is extracted from the Kerberos database and stored locally with the service principal on the service component host. First, create the principal using mandatory naming conventions. Then, create the keytab file with information from the new principal and copy the keytab to the keytab directory on the appropriate Storm host.

> **Note**
>
> Principals can be created either on the Kerberos Key Distribution Center (KDC) host or over the network using an "admin" principal. The following instructions assume you are using the KDC machine and using the kadmin.local command line administration utility. Using kadmin.local on the KDC machine allows you to create principals without needing to create a separate "admin" principal before you start.

Perform the following procedure on the host that runs KDC:

1. Make sure that you have performed the steps in Securing ZooKeeper with Kerberos.

2. Create a principal for the Nimbus server and the Storm DRPC daemon:

   ```
   sudo kadmin.local -q 'addprinc storm/
   <STORM_HOSTNAME>@STORM.EXAMPLE.COM'
   ```

3. Create a keytab for the Nimbus server and the Storm DRPC daemon:

   ```
   sudo kadmin.local -q "ktadd -k /tmp/storm.keytab storm/
   <STORM_HOSTNAME>@STORM.EXAMPLE.COM"
   ```

4. Copy the keytab to the Nimbus node and the node that runs the Storm DRPC daemon.

5. Run the following command to create a principal for the Storm UI daemon, the Storm Logviewer daemon, and the nodes running the process controller, such as Supervisor. A process controller is used to start and stop the Storm daemons.

```
sudo kadmin.local -q 'addprinc storm@STORM.EXAMPLE.COM'
```

6. Create a keytab for the Storm UI daemon, the Storm Logviewer daemon, and Supervisor:

```
sudo kadmin.local -q "ktadd -k /tmp/storm.keytab
storm@STORM.EXAMPLE.COM"
```

7. Copy the keytab to the cluster nodes running the Storm UI daemon, the Storm Logviewer daemon, and Supervisor.

Update the jaas.conf Configuration File

Both Storm and ZooKeeper use Java Authentication and Authorization Services (JAAS), an implementation of the Pluggable Authentication Model (PAM), to authenticate users. Administrators must update the `jaas.conf` configuration file with the keytab and principal information from the last step. The file must appear on all Storm nodes, the Nimbus node, the Storm DRPC node, and all Gateway nodes. However, different cluster nodes require different stanzas, as indicated in the following table:

Table 4.1. Required jaas.conf Sections for Cluster Nodes

Cluster Node	Required Sections in jaas.conf
Storm	StormClient
Nimbus	StormServer, Client
DRPC	StormServer
Supervisor	StormClient, Client
Gateway	StormClient (different structure than used on Storm and Supervisor nodes)
ZooKeeper	Server

Note

JAAS ignores unnecessary sections in `jaas.conf`. Administrators can put all sections in all copies of the file to simplify the process of updating it. However, the StormClient stanza for the Gateway nodes uses a different structure than the StormClient stanza on other cluster nodes. In addition, the StormServer stanza for the Nimbus node requires additional lines, as does the zoo.cfg configuration file for the ZooKeeper nodes.

The following example `jaas.conf` file contains all sections and includes information about the keytabs and principals generated in the previous step.

```
StormServer {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/keytabs/storm.keytab"
storeKey=true
useTicketCache=false
principal="storm/storm.example.com@STORM.EXAMPLE.COM";
};
```

```
StormClient {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/keytabs/storm.keytab"
storeKey=true
useTicketCache=false
serviceName="storm"
principal="storm@STORM.EXAMPLE.COM";
};

Client {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/keytabs/storm.keytab"
storeKey=true
useTicketCache=false
serviceName="zookeeper"
principal="storm@STORM.EXAMPLE.COM";
};
```

The StormServer section for the Nimbus node must have the following additional lines:

```
StormServer {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/keytabs/storm.keytab"
storeKey=true
useTicketCache=false
principal="storm/storm.example.com@STORM.EXAMPLE.COM";
};
```

The StormClient stanza for the Gateway nodes must have the following structure:

```
StormClient {
com.sun.security.auth.module.Krb5LoginModule required
doNotPrompt=false
useTicketCache=true
serviceName="$nimbus_user";
};
```

The Server stanza for the ZooKeeper nodes must have the following structure:

```
Server {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/keytabs/zk.keytab"
storeKey=true
useTicketCache=false
serviceName="zookeeper"
principal="zookeeper/zk1.example.com@STORM.EXAMPLE.COM";
};
```

In addition, add the following `childopts` lines to the stanzas for the nimbus, ui, and supervisor processes:

```
nimbus.childopts: "-Xmx1024m -Djava.security.auth.login.config=/path/to/jaas.
conf"
ui.childopts: "-Xmx768m -Djava.security.auth.login.config=/path/to/jaas.conf"
supervisor.childopts: "-Xmx256m -Djava.security.auth.login.config=/path/to/
jaas.conf"
```

Note

When starting ZooKeeper, include the following command-line option so that ZooKeeper can find jaas.conf:

```
-Djava.security.auth.login.config=/jaas/zk_jaas.conf
```

Update the storm.yaml Configuration File

To enable authentication with Kerberos, add the following lines to the `storm.yaml` configuration file:

```
storm.thrift.transport:
"org.apache.storm.security.auth.kerberos.KerberosSaslTransportPlugin"
java.security.auth.login.config: "/path/to/jaas.conf"
nimbus.authorizer: "org.apache.storm.security.auth.authorizer.
SimpleACLAuthorizer"
storm.principal.tolocal: "org.apache.storm.security.auth.
KerberosPrincipalToLocal"
storm.zookeeper.superACL: "sasl:storm
"nimbus.admins: - "storm"
nimbus.supervisor.users: - "storm"
nimbus.childopts: "-Xmx1024m -Djavax.net.debug=ssl -Dsun.security.krb5.debug=
true
-Djava.security.auth.login.config=/vagrant/storm_jaas.conf
-Djava.security.krb5.realm=EXAMPLE.COM -Djava.security.krb5.kdc=kdc.example.
com"
ui.childopts: "-Xmx768m -Djavax.net.debug=ssl -Dsun.security.krb5.debug=true
-Djava.security.auth.login.config=/vagrant/storm_jaas.conf
-Djava.security.krb5.realm=EXAMPLE.COM -Djava.security.krb5.kdc=kdc.example.
com"
supervisor.childopts: "-Xmx256m -Djavax.net.debug=ssl -Dsun.security.krb5.
debug=true
-Djava.security.auth.login.config=/vagrant/storm_jaas.conf
-Djava.security.krb5.realm=EXAMPLE.COM -Djava.security.krb5.kdc=example.host1.
com"
ui.filter: "org.apache.hadoop.security.authentication.server.
AuthenticationFilter"
ui.filter.params: "type": "kerberos""kerberos.principal":
"HTTP/nimbus.example.com""kerberos.keytab":
"/vagrant/keytabs/http.keytab""kerberos.name.rules":
"RULE:[2:$1@$0]([jt]t@.*EXAMPLE.COM)s/.*/$MAPRED_USER/
RULE:[2:$1@$0]([nd]n@.*EXAMPLE.COM)s/.*/$HDFS_USER/DEFAULT"
```

4.5. (Optional) Configuring Authorization for Storm

Apache Storm supports authorization using Pluggable Authentication Modules, or PAM, with secure Hadoop clusters. Currently, Storm supports the following authorizers:

Table 4.2. Supported Authorizers

Authorizer	Description
org.apache.storm.security.auth.authorizer. SimpleACLAuthorizer	Default authorizer for the Nimbus node and all Storm nodes except DRPC.

Authorizer	Description
org.apache.storm.security.auth.authorizer. DRPCSimpleACLAuthorizer	Default authorizer for Storm DRPC nodes.
com.xasecure.authorization.storm.authorizer. XaSecureStormAuthorizer	Default authorizer for centralized authorization with Apache Ranger.

To enable authorization, perform the following steps:

1. Configure `storm.yaml` for Nimbus and Storm nodes.

2. Configure `worker-launcher.cfg` for worker-launcher.

3. Configure the Storm multi-tenant job scheduler.

Configure storm.yaml for Nimbus and Storm Nodes

When authorization is enabled, Storm prevents users from seeing topologies run by other users in the Storm UI. To do this, Storm must run each topology as the operating system user who submitted it rather than the user that runs Storm, typically storm, which is created during installation.

Use the following procedure to configure supervisor to run Storm topologies as the user who submits the topology, rather than as the storm user:

1. Verify that a headless user exists for supervisor, such as supervisor, on each Storm cluster node.

2. Create a headless operating system group, such as supervisor, on each Storm cluster node.

3. Set the following configuration properties in the storm.yaml configuration file for each node in the Storm cluster:

Table 4.3. storm.yaml Configuration File Properties

Configuration Property	Description
supervisor.run.worker.as.user	Set to true to run topologies as the user who submits them.
topology.auto-credentials	Set to a list of Java plugins that pack and unpack user credentials for Storm workers. This should be set to `org.apache.storm.security.auth.kerberos.AutoTGT`.
drpc.authorizer	Set to `org.apache.storm.security.auth.authorizer.DRPCSimpleACLAu` to enable authorizer for Storm DRPC node.
nimbus.authorizer:	Set to `org.apache.storm.security.auth.authorizer.SimpleACLAuthor` to enable authorizer for Storm nimbus node.
storm.principal.tolocal:	Set to `org.apache.storm.security.auth.KerberosPrincipalToLocal` to enable transforming kerberos principal to local user names.
storm.zookeeper.superACL:	Set to `sasl:storm` to set the acls on zookeeper nodes so only user `storm` can modify those nodes.

4. Change the owner of `worker-launcher.cfg` to root and verify that only root has write permissions on the file.

5. Change the permissions for the worker-launcher executable to 6550.

6. Verify that all Hadoop configuration files are in the CLASSPATH for the Nimbus server.

7. Verify that the nimbus operating system user has superuser privileges and can receive delegation tokens on behalf of users submitting topologies.

8. Restart the Nimbus server.

Configure worker-launcher.cfg

`/usr/hdf/current/storm-client/bin/worker-launcher` is a program that runs Storm worker nodes. You must configure worker-launcher to run Storm worker nodes as the user who submitted a topology, rather than the user running the supervisor process controller. To do this, set the following configuration properties in the `/etc/storm/conf/worker-launcher.cfg` configuration file on all Storm nodes:

Table 4.4. worker-launcher.cfg File Configuration Properties

Configuration Property	Description
storm.worker-launcher.group	Set this to the headless OS group that you created earlier.
min.user.id	Set this to the first user ID on the cluster node.

Configure the Storm Multi-tenant Scheduler

The goal of the multi-tenant scheduler is to both isolate topologies from one another and to limit the resources that an individual user can use on the cluster. Add the following configuration property to `multitenant-scheduler.yaml` and place it in the same directory with `storm.yaml`.

Table 4.5. multitenant-scheduler.yaml Configuration File Properties

Configuration Property	Description
multitenant.scheduler.user.pools	Specifies the maximum number of nodes a user may use to run topologies.

The following example limits users evans and derek to ten nodes each for all their topologies:

```
multitenant.scheduler.user.pools: "evans": 10 "derek": 10
```

Note

The multi-tenant scheduler relies on Storm authentication to distinguish between individual Storm users. Verify that Storm authentication is already enabled.

4.6. Validate the Installation

Validate the Apache Storm installation to verify a successful installation and configuration.

Important

You must start ZooKeeper before starting Storm.

1. Run the following command to start the Storm daemons:

    • RHEL/CentOS/Oracle Linux

      ```
      etc/init.d/supervisor start
      ```

    • SLES

      ```
      etc/init.d/supervisor start
      ```

    • Ubuntu or Debian

      ```
      etc/init.d/supervisor start
      ```

2. Run the following command to view the status of the Storm daemons:

    • RHEL/CentOS/Oracle Linux

      ```
      /usr/bin/supervisorctl status
      ```

    • SLES

      ```
      /usr/bin/supervisorctl status
      ```

    • Ubuntu

      ```
      service supervisor status
      ```

      You should see output similar to the following:

      ```
      storm-drpc RUNNING pid 3368, uptime 0:31:31
      storm-logviewer RUNNING pid 3365, uptime 0:31:31
      storm-nimbus RUNNING pid 3370, uptime 0:31:31
      storm-supervisor RUNNING pid 8765, uptime 0:00:12
      storm-ui RUNNING pid 3369, uptime 0:31:31
      ```

3. Point your browser to the following URL:

      ```
      http://<storm-ui-server>:8080
      ```

      You should see the Storm UI web page:

Storm UI

Cluster Summary

Version	Nimbus uptime	Supervisors	Used slots	Free slots	Total slots	Executors	Tasks
0.9.0.1	34m 15s	0	0	0	0	0	0

Topology summary

Name	Id	Status	Uptime	Num workers	Num executors	Num tasks
WordCount	WordCount-2-1391628483	ACTIVE	21d 6h 5m 24s	0	0	0

Supervisor summary

Id	Host	Uptime	Slots	Used slots

Nimbus Configuration

Key	Value
dev.zookeeper.path	/tmp/dev-storm-zookeeper
drpc.childopts	-Xmx768m
drpc.invocations.port	3773
drpc.port	3772
drpc.queue.size	128
drpc.request.timeout.secs	600
drpc.servers	["localhost"]
drpc.worker.threads	64

4. Run the following command to run the WordCount sample topology:

```
storm jar /usr/hdp/current/storm-client/contrib/storm-starter/storm-starter-
topologies-*.jar org.apache.storm.starter.WordCountTopology wordcount
```

```
storm jar /usr/hdf/current/storm-client/contrib/storm-starter/storm-starter-
topologies-*.jar org.apache.storm.starter.WordCountTopology wordcount
```

5. Installing Apache Ranger

Apache Ranger delivers a comprehensive approach to security for a Hadoop cluster. It provides central security policy administration across the core enterprise security requirements of authorization, auditing, and data protection.

This chapter describes the manual installation process for Apache Ranger and the Ranger plug-ins in a Linux Hadoop environment. It includes information about the following steps:

- Installation Prerequisites [24]

- Installing Policy Manager [24]

- Installing UserSync [30]

- Installing Ranger Plug-ins [39]

- Verifying the Installation [76]

For information about Apache Ranger security, refer to the HDP Security Features of the *Hadoop Security Guide*.

For information about installing Ranger using Ambari, see Installing Ranger Using Ambari.

For information about installing Ranger KMS, see Installing Ranger KMS.

5.1. Installation Prerequisites

Before beginning Ranger installation, make sure the following software is already installed:

- Solr 5.2 or above

- Refer to the HDP Release Notes for information regarding supported operating systems.

- Refer to the HDP Release Notes for information regarding supported JDKs.

- Refer to the HDP Release Notes for information regarding supported databases.

  If the database server is not installed at the same host, Ranger services need to have access to the database server host.

- If you plan to setup Ranger authorization to use LDAP/AD, refer to the Setting Up Hadoop Group Mapping for LDAP/AD in the *Hadoop Security Guide*.

5.2. Installing Policy Manager

This section describes how to perform the following administrative tasks:

1. Configure and install the Ranger Policy Manager

2. Start the Policy Manager service

5.2.1. Install the Ranger Policy Manager

1. Make sure the HDP 2.6.0 resource-based service is added to your site's list of available repositories.

   If it has not yet been added, add it now by performing the following steps:

   • For RHEL6/Centos6/Oracle LINUX 6:

   ```
   wget -nv http://public-repo-1.hortonworks.com/HDP/centos6/2.x/GA/2.6.0.0/
   hdp.repo -O /etc/yum.repos.d/hdp.repo
   ```

   • For Ubuntu 12/14:

   ```
   apt-get update wget http://public-repo-1.hortonworks.com/HDP/ubuntu12/2.x/
   GA/2.6.0.0/hdp.list -O /etc/apt/sources.list.d/hdp.list
   ```

   • For Debian:

   ```
   apt-get update wget http://public-repo-1.hortonworks.com/HDP/
   debian<version>/2.x/GA/2.6.0.0/hdp.list -O /etc/apt/sources.list.d/hdp.
   list
   ```

2. Find the Ranger Policy Admin software:

   a. For RHEL/Centos/Oracle LINUX:

      yum search ranger

   b. For Ubuntu 12/14, Debian:

   ```
   aptitude search ranger
   ```

3. Install the Ranger Policy Admin software:

   ```
   yum install ranger-admin
   ```

4. ```
   apt-get install <package_name>
   ```

   In the Ranger Policy Administration installation directory, update the `install.properties` file:

   • Go to the installation directory:

   ```
   cd /usr/hdp/<version>/ranger-admin/
   ```

   • Edit the following `install.properties` entries:

   ### Table 5.1. install.properties Entries

Configuration Property	Default/Example Value	Required?
Ranger Policy Database		
DB_FLAVOR Specifies the type of database used (MYSQL,ORACLE,POSTGRES,MSSQL)	MYSQL (default)	Y
SQL_CONNECTOR_JAR Path to SQL connector jar of the DB Flavor selected. The value should be the	/usr/share/java/mysql-connector-java.jar (default)	Y

Configuration Property	Default/Example Value	Required?
absolute path including the jar name.	/usr/share/java/postgresql.jar /usr/share/java/sqljdbc4.jar /usr/share/java/ojdbc6.jar	
db_root_user database username who has privileges for creating database schemas and users	root (default)	Y
db_root_password database password for the "db_root_user"	rootPassW0Rd	Y
db_host Hostname of the Ranger policy database server	localhost	Y
db_name Ranger Policy database name	ranger (default)	Y
db_user db username used for performing all policy mgmt operation from policy admin tool	rangeradmin (default)	Y
db_password database password for the "db_user"	RangerAdminPassW0Rd	Y
Ranger Audit		
audit_solr_urls	http://<solr_host>:8886/solr/ranger_audits	Y
audit_solr_user		Y
audit_solr_password		Y
audit_solr_zookeepers		Only required if SolrCloud is used.
Policy Admin Tool Config		
policymgr_external_url URL used within Policy Admin tool when a link to its own page is generated in the Policy Admin Tool website	*http://localhost:6080* (default) *http:// myexternalhost.xasecure.net:6080N*	
policymgr_http_enabled Enables/ disables HTTP protocol for downloading policies by Ranger plug-ins	true (default)	Y
unix_user UNIX user who runs the Policy Admin Tool process	ranger (default)	Y
unix_group UNIX group associated with the UNIX user who runs the Policy Admin Tool process	ranger (default)	Y
Policy Admin Tool Authentication		
authentication_method Authentication Method used to log in to the Policy Admin Tool. NONE: only users created within the Policy Admin Tool may log in UNIX: allows UNIX userid authentication using the UNIX authentication service (see below) LDAP: allows Corporate LDAP authentication (see below) ACTIVE_DIRECTORY: allows authentication using an Active Directory	none (default)	Y

Configuration Property	Default/Example Value	Required?
UNIX Authentication Service		
remoteLoginEnabled Flag to enable/disable remote Login via Unix Authentication Mode	true (default)	Y, if UNIX authentication_method is selected
authServiceHostName Server Name (or ip-addresss) where ranger-usersync module is running (along with Unix Authentication Service)	localhost (default) myunixhost.domain.com	Y, if UNIX authentication_method is selected
authServicePort Port Number where ranger-usersync module is running Unix Authentication Service	5151 (default)	Y, if UNIX authentication_method is selected
LDAP Authentication		
xa_ldap_url URL for the LDAP service	ldap://<ldapServer>:389	Y, if LDAP authentication_method is selected
xa_ldap_userDNpattern LDAP DN Pattern used to uniquely locate the login user	uid={0},ou=users,dc=xasecure,dc=net	Y, if LDAP authentication_method is selected
xa_ldap_groupSearchBase LDAP Base node location to get all groups associated with login user	ou=groups,dc=xasecure,dc=net	Y, if LDAP authentication_method is selected
xa_ldap_groupSearchFilter LDAP search filter used to retrieve groups for the login user	(member=uid={0},ou=users, dc=xasecure,dc=net)	Y, if LDAP authentication_method is selected
xa_ldap_groupRoleAttribute Attribute used to retrieve the group names from the group search filters	cn	Y, if LDAP authentication_method is selected
Active Directory Authentication		
xa_ldap_ad_domain Active Directory Domain Name used for AD login	xasecure.net	Y, if ACTIVE_DIRECTORY authentication_method is selected
xa_ldap_ad_url Active Directory LDAP URL for authentication of user	ldap://ad.xasecure.net:389	Y, if ACTIVE_DIRECTORY authentication_method is selected

5. If you are using an SSL-enabled, MySQL database for Ranger, add the following properties to `install.properties`:

```
db_ssl_enabled=false
db_ssl_required=false
db_ssl_verifyServerCertificate=false
javax_net_ssl_keyStore=
javax_net_ssl_keyStorePassword=
javax_net_ssl_trustStore=
javax_net_ssl_trustStorePassword=
```

6. If Ranger Admin is SSL-enabled, add the following Ranger Admin SSL properties to the `install.properties` file. These properties secure the Ranger SSL password in the `jceks` file.

```
policymgr_https_keystore_file=<SSL keystore file path used to configure
 Ranger in SSL>
policymgr_https_keystore_keyalias=rangeradmin
policymgr_https_keystore_password=<SSL password used to create keystore>
```

7. If Unix-Auth and User-Sync service is SSL-enabled, add the following Ranger Unix-Auth SSL properties to the `install.properties` file:

```
ranger_unixauth_keystore=keystore.jks
ranger_unixauth_keystore_password=password
ranger_unixauth_truststore=cacerts
ranger_unixauth_truststore_password=changeit
```

8. The RANGER_PID_DIR_PATH property introduces a custom PID path for the Ranger Admin Process. To configure this property to start and stop the Ranger Admin service, add the following property to `install.properties`. The default value is `/var/run/ranger`.

```
RANGER_PID_DIR_PATH=/var/run/ranger
```

9. Check the JAVA_HOME environment variable. If it has not yet been set, enter:

```
export JAVA_HOME=<path of installed jdk version folder>
```

5.2.2. Install the Ranger Policy Administration Service

To install the Ranger Policy Administration service, run the following commands:

```
cd /usr/hdp/<version>/ranger-admin
```

```
./setup.sh
```

5.2.3. Start the Ranger Policy Administration Service

To start the Ranger Policy Administration service, enter the following command:

```
service ranger-admin start
```

To verify that the service started, visit the external URL specified in install.properties in browser; for example:

```
http://<host_address>:6080/
```

Note

The default user is "admin" with a password of "admin". After login, change the password for "admin".

5.2.4. Configuring the Ranger Policy Administration Authentication Mode

The Ranger service also enables you to configure the authentication method that the Ranger Policy Administration component uses to authenticate users. There are three different authentication methods supported with Ranger, which include:

• Active Directory (AD)

• LDAP

- UNIX

Depending on which authentication method you choose, you will need to modify the following sample values in the `install.properties` file:

<u>Active Directory</u>

- authentication_method=ACTIVE_DIRECTORY

- xa_ldap_ad_domain= example.com

- xa_ldap_ad_url=ldap://127.0.0.1:389

- xa_ldap_ad_base_dn=DC=example,DC=com

- xa_ldap_ad_bind_dn=CN=Administrator,CN=Users,DC=example,DC=com

- xa_ldap_ad_bind_password=PassW0rd

- xa_ldap_ad_referral=ignore, follow or throw. Default is follow.

<u>LDAP</u>

- authentication_method=LDAP

- xa_ldap_url=LDAP server URL (e.g. ldap://127.0.0.1:389)

- xa_ldap_userDNpattern=uid={0},ou=users,dc=example,dc=com

- xa_ldap_groupSearchBase=dc=example,dc=com

- xa_ldap_groupSearchFilter=(member=cn={0},ou=users,dc=example,dc=com

- xa_ldap_groupRoleAttribute=cn

- xa_ldap_base_dn=dc=example,dc=com

- xa_ldap_bind_dn=cn=ldapadmin,ou=users,dc=example,dc=com

- xa_ldap_bind_password=PassW0rd

- xa_ldap_referral=ignore, follow, or throw. Default is follow.

- xa_ldap_userSearchFilter=(uid={0}) property at Ranger-admin side

<u>UNIX</u>

- authentication_method=UNIX

- remoteLoginEnabled=true

- authServiceHostName= an address of the host where the UNIX authentication service is running.

- authServicePort=5151

5.2.5. Configuring Ranger Policy Administration High Availability

If you would like to enable high availability for the Ranger Policy Administration component, you can configure the component by following the steps listed below.

1. Install the Ranger Admin component on the hosts you wish to use.

   ### Note

   Make sure you use the same configuration and policy database settings for each host, otherwise, you will be unable to configure HA for the hosts.

2. Configure a load balancer to balance the loads among the various Ranger Admin instances and take note of the load balancer URL.

   ### Note

   Describing the steps you need to follow in order to install and configure a load balancer is not in the scope of this book.

3. Update the Policy Manager external URL in all Ranger Admin clients (Ranger UserSync and Ranger plug-ins) to point to the load balancer URL.

5.3. Installing UserSync

In this section:

- Using the LDAP Connection Check Tool [30]

- Install UserSync and Start the Service [36]

5.3.1. Using the LDAP Connection Check Tool

The LDAP Connection Check tool is a command line tool that helps Ranger administrators configure LDAP properties for the UserSync module. The tool collects minimal input from the administrator about the LDAP/AD server and discovers various properties for users and groups in order to successfully pull only targeted Users and Groups from the LDAP/ AD server. It provides options such as discovering/verifying UserSync-related properties as well as authentication properties, generating install properties for manual installation, etc. Once all of the required properties have been discovered and tested, these properties can be applied to the Ranger configuration during Ambari or non-Ambari cluster installation.

The LDAP Connection tool can be accessed in the `/usr/hdp/current/ranger-usersync/ldaptool` directory.

5.3.1.1. LDAP Connection Check Tool Parameters

You can use the `./run.sh -h` command to list the LDAP Connection Check tool parameters:

```
cd /usr/hdp/current/ranger-usersync/ldaptool
./run.sh -h
usage: run.sh
 -noauth          ignore authentication properties
 -d <arg>   {all|users|groups}
 -h                      show help.
 -i <arg>          Input file name
 -o <arg>          Output directory
 -r <arg>           {all|users|groups}
```

All these parameters are optional.

- If "-i" (for input file) is not specified, the tool will fall back to the CLI option for collecting values for mandatory properties.

- if "-o" (for output directory) is not specified, the tool will write all of the output files to the `/usr/hdp/current/ranger-usersync/ldaptool/output` directory.

- if "-noauth" (for ignoring authentication) is not specified, the tool will discovery and verify authentication-related properties.

- if "-d" (for discovering usersync properties) is not specified, the tool will default to discovering all of the usersync-related properties.

- if "-r" (for retrieving users and/or groups) is not specified, the tool will fallback to the "-d" option.

5.3.1.2. Input Properties

In order to discover the usersync and authentication related properties, the LDAP Connection Check tool collects some mandatory information as part of the input properties. These mandatory properties include:

- ranger.usersync.ldap.url (<ldap or ldaps>://<server ip/fqdn>:<port>)

- ranger.usersync.ldap.binddn (ldap user like AD user or ldap admin user)

- ranger.usersync.ldap.bindpassword (user password or ldap admin password)

- ranger.usersync.ldap.user.searchbase (Mandatory only for non AD environment)

- ranger.usersync.ldap.user.searchfilter (Mandatory only for non AD environment)

- ranger.admin.auth.sampleuser (Mandatory only for discovering authentication properties)

- ranger.admin.auth.samplepassword (Mandatory only for discovering authentication properties)

1. Modify the input.properties file provided as part of the tool installation and provide that file (with the complete path as the command line argument while running the tool.

2. Use the CLI to input the values for these mandatory properties.

The CLI option is provided to the user when the input file is not provided as the command line option (-i <arg>) while running the tool. Once the values are collected from the CLI,

these values are stored in the input.properties file (in the conf dir of the installation folder) for later use.

The following is the CLI provided by the tool when input file is not specified. The tool provides two options for collecting values for these mandatory properties:

```
Ldap url [ldap://ldap.example.com:389]:
Bind DN [cn=admin,ou=users,dc=example,dc=com]:
Bind Password:
User Search Base [ou=users,dc=example,dc=com]:
User Search Filter [cn=user1]:
Sample Authentication User [user1]:
Sample Authentication Password:
```

Note

In order to use secure LDAP, the Java default truststore must be updated with the server's self signed certificate or the CA certificate for validating the server connection. The truststore should be updated before running the tool.

5.3.1.3. Discovery of UserSync Properties

Usersync-related properties are divided into two categories: User search related properties and group search related properties. This tool provides a `-d` option to discover user related and group related properties separately or all at once. The discover properties option is used as follows:

```
./run.sh -d <arg>
```

where `<arg>` can be

- `all`: discover all of the properties at once or

- `users`: discover only user search related properties or

- `groups`: discover only group search related properties

These properties are discovered based on the values provided in the input file for all of the mandatory properties.

The following are the user search related properties that are discovered using this tool:

1. Basic properties:

    - ranger.usersync.ldap.user.objectclass

    - ranger.usersync.ldap.user.groupnameattribute

    - ranger.usersync.ldap.user.nameattribute

2. Advanced properties:

    - ranger.usersync.ldap.user.searchbase

    - ranger.usersync.ldap.user.searchfilter

Group search related properties that are discovered by this tool are as follows:

1. Basic properties:

   - ranger.usersync.group.searchenabled

   - ranger.usersync.group.objectclass

   - ranger.usersync.group.memberattributename

   - ranger.usersync.group.nameattribute

2. Advanced properties:

   - ranger.usersync.group.searchbase

   - ranger.usersync.group.searchfilter

Once all of the properties are discovered, the tool also retrieves the total count and details of first 20 users and/or groups and displays them in the output.

1. The value for the user search base is derived as the OU with max. no of users (from the first 20 users that are retrieved).

2. The value for the user search filter is derived as <user name attribute>=*

3. The value for the group search base is derived as the OU with max. no of groups (from the first 20 retrieved groups).

4. The value for the group search filter is derived as <group name attribute>=*

5.3.1.4. Discovery of Authentication Properties

The LDAP Connection Check tool provides a `-noauth` option to skip discovery of authentication properties. When this option is used, the tool will not suggest the values for authentication related properties.

```
./run.sh -noauth
```

If the LDAP server is of type active directory, the following properties are suggested:

- ranger.authentication.method

- ranger.ldap.ad.domain

If the LDAP server is not an active directory, the following properties are suggested:

- ranger.authentication.method

- ranger.ldap.user.dnpattern

- ranger.ldap.group.roleattribute

- ranger.ldap.group.searchbase

- ranger.ldap.group.searchfilter

These authentication properties can be discovered either by providing the values in the input file for only mandatory properties, or for all of the user and/or group related

properties. After discovering the authentication properties, the tool also validates those properties by authenticating the given user, and reports authentication success or failure in the output.

5.3.1.5. Retrieval of Users and Groups

Usersync-related properties are divided into two categories: User search related properties and group search related properties. This tool provides a `-d` option to discover user related and group related properties separately or all at once. The discover properties option is used as follows:

```
./run.sh -r <arg>
```

where `<arg>` can be

- `users` : retrieve the total count and details of the first 20 users and associated groups, given the user search related properties in the input file.

- `groups` : retrieve the total count and details of the first 20 groups and associated users, given the group search related properties in the input file.

- `all` : retrieve both users and groups, given all of the corresponding properties in the input file.

5.3.1.6. Output Directory Content

This tool generates three files in the output directory specified with the `-o` option, or by default to the `/usr/hdp/current/ranger-usersync/ldaptool/output` directory.

- ambari.properties

- install.properties

- ldapConfigCheck.log

All of the discovered properties (related to usersync and/or authentication) are written to both the ambari.properties and install.properties files with the corresponding property names.

All of the other information, such as any retrieved users/groups, total count, authentication result, etc. are written to the ldapConfigCheck.log file. This log file also contains any errors or warnings generated while running the tool.

5.3.1.7. Other UserSync Related Properties

Some of the other usersync-related properties that are used by the tool and left with default values are:

- ranger.usersync.ldap.authentication.mechanism - Default authentication mechanism used is "simple".

- ranger.usersync.pagedresultsenabled - Default is set to "true".

- ranger.usersync.pagedresultssize - Default value for this property is "500". This value can be tweaked depending on the bandwidth and resource availability in the deployment.

- ranger.usersync.ldap.username.caseconversion - Default value is set to "lower"

- ranger.usersync.ldap.groupname.caseconversion - Default value is set to "lower"

- ranger.usersync.ldap.user.searchscope - Default is set to "sub". This value can be set to either "base" or "one" depending on how the user search is to be performed.

- ranger.usersync.group.searchscope - Default is set to "sub". This value can be set to either "base" or "one" depending on how the group search is to be performed.

The following are the remaining usersync-related properties. These properties are not currently used by the tool and the values are left empty in the input file.

- ranger.usersync.credstore.filename - this property is unused as the tool supports only cleartext password.

- ranger.usersync.ldap.bindalias - this property is also not used by the tool.

- ranger.usersync.ldap.searchBase - This property is used as the user search base or group search base when they are not configured. Hence this value is left blank and not used by the tool.

- ranger.usersync.group.usermapsyncenabled - Mainly used for computing group memberships while retrieving users. Currently this value is set to "true", but is not used by the tool.

5.3.1.8. Assumptions

Some properties are assumed to have one or more values as follows:

- User name attribute : **"sAMAccountName"** , **"uid"** , **"cn"**

- User Object class value : **"person"** , **"posixAccount"**

- User group member attribute : **"memberOf"** , **"ismemberOf"**

- Group Object class : **"group"** , **"groupOfNames"** , **"posixGroup"**

- Group name attribute : **"distinguishedName"** , **"cn"**

- Group member attribute : **"member"** , **"memberUid"**

5.3.1.9. Sample input.properties File

```
# Mandatory ldap configuration properties.
ranger.usersync.ldap.url=
ranger.usersync.ldap.binddn=
ranger.usersync.ldap.ldapbindpassword=

# Mandatory only for openLdap
ranger.usersync.ldap.user.searchbase=
ranger.usersync.ldap.user.searchfilter=

# For verifying authentication please provide sample username and password
ranger.admin.auth.sampleuser=
ranger.admin.auth.samplepassword=
```

```
# Optional properties will be determined based on the above search
# User attributes
ranger.usersync.ldap.user.nameattribute=
ranger.usersync.ldap.user.objectclass=
ranger.usersync.ldap.user.groupnameattribute=

# Group attributes
ranger.usersync.group.searchenabled=false
ranger.usersync.group.memberattributename=
ranger.usersync.group.nameattribute=
ranger.usersync.group.objectclass=
ranger.usersync.group.searchbase=
ranger.usersync.group.searchfilter=

# Other UserSync related attributes
ranger.usersync.ldap.authentication.mechanism=simple
ranger.usersync.pagedresultsenabled=true
ranger.usersync.pagedresultssize=500
ranger.usersync.ldap.username.caseconversion=lower
ranger.usersync.ldap.groupname.caseconversion=lower
ranger.usersync.ldap.user.searchscope=sub
ranger.usersync.group.searchscope=sub

ranger.usersync.credstore.filename=
ranger.usersync.ldap.bindalias=
ranger.usersync.ldap.searchBase=
ranger.usersync.group.usermapsyncenabled=false

# Authentication properties
ranger.authentication.method=
ranger.ldap.ad.domain=
ranger.ldap.user.dnpattern=
ranger.ldap.group.roleattribute=
ranger.ldap.group.searchbase=
ranger.ldap.group.searchfilter=
```

5.3.2. Install UserSync and Start the Service

Important

To ensure that LDAP/AD group level authorization is enforced in Hadoop, you should set up Hadoop group mapping for LDAP/AD.

To install Ranger UserSync and start the service, do the following:

1. Find the Ranger UserSync software:

```
yum search usersync
```

or

```
yum list | grep usersync
```

2. Install Ranger UserSync:

Note

Make sure the database on which Ranger will be installed is up and running.

```
yum install ranger_<version>-usersync.x86_64
```

3. At the Ranger UserSync installation directory, update the following properties in the `install.properties` file:

Table 5.2. Properties to Update in the install.properties File

Configuration Property Name	Default/Example Value	Required?
Policy Admin Tool		
POLICY_MGR_URL URL for policy admin	*http:// policymanager.xasecure.net:6080*	Y
User Group Source Information		
SYNC_SOURCE Specifies where the user/group information is extracted to be put into Ranger database. unix - get user information from /etc/passwd file and gets group information from /etc/group file ldap - gets user information from LDAP service (see below for more information)	unix	N
SYNC_INTERVAL Specifies the interval (in minutes) between synchronization cycle. Note, the 2nd sync cycle will NOT start until the first sync cycle is COMPLETE.	5	N
UNIX user/group Synchronization		
MIN_UNIX_USER_ID_TO_SYNC UserId below this parameter values will not be synchronized to Ranger user database	300 (Unix default), 1000 (LDAP default)	Mandatory if SYNC_SOURCE is selected as unix
LDAP user/group synchronization		
SYNC_LDAP_URL URL of source ldap	ldap://ldap.example.com:389	Mandatory if SYNC_SOURCE is selected as ldap
SYNC_LDAP_BIND_DN ldap bind dn used to connect to ldap and query for users and groups	cn=admin,ou=users,dc=hadoop, dc=apache,dc-org	Mandatory if SYNC_SOURCE is selected as ldap
SYNC_LDAP_BIND_PASSWORD ldap bind password for the bind dn specified above	LdapAdminPassW0Rd	Mandatory if SYNC_SOURCE is selected as ldap
CRED_KEYSTORE_FILENAME Location of the file where encrypted password is kept	*/usr/lib/xausersync/.jceks/ xausersync.jceks* (default) */ etc/ranger/usersync/.jceks/ xausersync.jceks*	Mandatory if SYNC_SOURCE is selected as ldap
SYNC_LDAP_USER_SEARCH_BASE Search base for users	ou=users,dc=hadoop,dc=apache, dc=org	Mandatory if SYNC_SOURCE is selected as ldap
SYNC_LDAP_USER_SEARCH_SCOPE Search scope for the users, only base, one, and sub are supported values	sub (default)	N
SYNC_LDAP_USER_OBJECT_CLASS objectclass to identify user entries	person (default)	N (defaults to person)
SYNC_LDAP_USER_SEARCH_FILTER Optional additional filter constraining the users selected for syncing	(dept=eng)	N (defaults to an empty string)
SYNC_LDAP_USER_NAME _ATTRIBUTE Attribute from user	cn (default)	N (defaults to cn)

Configuration Property Name	Default/Example Value	Required?
entry that would be treated as user name		
SYNC_LDAP_USER_GROUP_NAME _ATTRIBUTE attribute from user entry whose values would be treated as group values to be pushed into Policy Manager database. You can provide multiple attribute names separated by comma	memberof,ismemberof (default)	N (defaults to memberof, ismemberof)
SYNC_LDAP_SEARCH_BASE	Default is False. dc=example,de=com	N

SYNC_GROUP_SEARCH_ENABLED	Default is False.
	
	If set to True, and **SYNC_GROUP_USER_MAP_SYNC_ENABLED** is also set to True, you must set the following properties:
	
	`SYNC_GROUP_SEARCH_BASE=ou=`
	`People,dc=example,dc=com`
	`SYNC_GROUP_SEARCH_SCOPE=sub`
	`SYNC_GROUP_OBJECT_CLASS=`
	`groupofnames`
	`SYNC_LDAP_GROUP_SEARCH_FILTER=`
	`SYNC_GROUP_NAME_ATTRIBUTE=cn`
	`SYNC_GROUP_MEMBER_ATTRIBUTE_NAME=`
	`member`
	`SYNC_PAGED_RESULTS_ENABLED=true`
	`SYNC_PAGED_RESULTS_SIZE=`
	`500RANGER_`
	`SYNC_LDAP_REFERRAL=follow,`
	`ignore`
		N

User Synchronization		
unix_user UNIX User who runs the ranger-usersync process	ranger (default)	Y
unix_group UNIX group associated with Unix user who runs the ranger-usersync process	ranger (default)	Y
SYNC_LDAP_USERNAME_CASE _CONVERSION Convert all username to lower/upper case none - no conversation will be done. Kept as it is in the SYNC_SOURCE lower - convert it to lower case when saving it to ranger db upper - convert it to upper case when saving it to ranger db	lower (default)	N (defaults to lower)
SYNC_LDAP_GROUPNAME_CASE _CONVERSION Convert all username to lower/upper case none - no conversation will be done. Kept as it is in the SYNC_SOURCE lower - convert it to lower case when saving it to ranger db upper - convert it to upper case when saving it to ranger db	lower (default)	N (defaults to lower)
logdir Location of the log directory where the usersync logs are stored	logs (default)	Y

4. Add the following property to the `install.properties` file to set the base directory for the Ranger Usersync process:

```
ranger_base_dir=/etc/ranger
```

5. Add the following properties to the `install.properties` file to set SSL configurations for Ranger Usersync:

```
AUTH_SSL_ENABLED=false
AUTH_SSL_KEYSTORE_FILE=/etc/ranger/usersync/conf/cert/unixauthservice.jks
AUTH_SSL_KEYSTORE_PASSWORD=UnIx529p
AUTH_SSL_TRUSTSTORE_FILE=
AUTH_SSL_TRUSTSTORE_PASSWORD=
```

6. Add the following property to the `install.properties` file to configure the Ranger Usersync PID directory to start and stop the Ranger Usersync service:

```
USERSYNC_PID_DIR_PATH=/var/run/ranger
```

7. Set the Policy Manager URL to *http://<ranger-admin-host>:6080*

8. Check the JAVA_HOME environment variable. If JAVA_HOME has not yet been set, enter:

```
export JAVA_HOME=<path of installed jdk version folder>
```

9. Install the Ranger UserSync service:

```
cd /usr/hdp/<version>/ranger-usersync
```

```
./setup.sh
```

10 Start the Ranger UserSync service:

```
service ranger-usersync start
```

11 To verify that the service was successfully started, wait 6 hours for LDAP/AD to synchronize, then do the following:

• Go to

```
http://<ranger-admin-host>:6080
```

• Click the Users/Group tab. See if users and groups are synchronized.

• Add a UNIX/LDAP/AD user, then check for the presence of that user in the Ranger Admin tab.

5.4. Installing Ranger Plug-ins

The following sections describe how to install Ranger plug-ins.

Note

To ensure that you are installing the HDP version of the plug-ins instead of the Apache version, make sure you enter the following commands when installing each plug-in:

- For CentOS and RHEL:

```
yum install ranger_ <version_number>
```

- For SLES:

```
zypper -n --no-gpg-checks install --auto-agree-with-licenses
 ranger_ <version_number>
```

- For Debian/Ubuntu:

```
apt-get install <version_number>
```

- Set up the JAVA_HOME environment variable to point to Java distribution on the installation machine.

```
export JAVA_HOME=location-of-java-home-on-the-machine
```

- Edit the `install.properties` file in the `ranger-tagsync-install-directory` to support the operational environment.

- Keeping in mind the following two guidelines, edit the `install.properties` file in the `ranger-tagsync-install-directory` to add Audit to solr properties:

  - You must configure the XAAUDIT.SOLR.URL property based on your Solr installation. See `http://<solr_host>:8886/solr/ranger_audits` for details.

  - You must configure the XAAUDIT.SOLR.ZOOKEEPER property to NONE, if you are using stand alone Solr. or <zk1>:2181,<zk2>:2181/ranger_audits, using the correct zookeeper URL, if you are using SolrCloud.

```
XAAUDIT.SOLR.ENABLE=true
XAAUDIT.SOLR.URL=http://<solr_host>:8886/solr/ranger_audits
XAAUDIT.SOLR.USER=NONE
XAAUDIT.SOLR.PASSWORD=NONE
XAAUDIT.SOLR.ZOOKEEPER=NONE
XAAUDIT.SOLR.FILE_SPOOL_DIR=/var/log/hadoop/hdfs/audit/solr/spool
```

5.4.1. Installing the Ranger HDFS Plug-in

The Ranger HDFS plug-in helps to centralize HDFS authorization policies.

This section describes how to create an HDFS resource-based service and install the HDFS plug-in.

Install the HDFS Plug-in

1. Create an HDFS resource-based service in the Ranger Policy Manager. To do this, complete the HDFS Create Service screen, as described in the Configure an HDFS Service section of the *Hadoop Security Guide*.

   Make a note of the name you gave to this resource-based service; you will need to use it again during HDFS plug-in setup.

2. At all servers where NameNode is installed, install the HDFS plug-in by following the steps listed below:

a. Go to the home directory of the HDFS plug-in:

```
cd /usr/hdp/<version>/ranger-hdfs-plugin
```

b. Edit the following HDFS-related properties in the install.properties file:

Table 5.3. Properties to Edit in the install.properties File

Configuration Property Name	Default/Example Value	Required?
Policy Admin Tool		
POLICY_MGR_URL URL for policy admin	http://policymanager.xasecure.net:6080	Y
REPOSITORY_NAME The repository name used in Policy Admin Tool for defining policies	hadoopdev	Y
Audit Database		
SQL_CONNECTOR_JAR Path to SQL connector jar of the DB Flavor selected. The value should be the absolute path including the jar name.	/usr/share/java/mysql-connector-java.jar (default) /usr/share/java/postgresql.jar /usr/share/java/sqljdbc4.jar /usr/share/java/ojdbc6.jar	Y
HDFS Audit		
XAAUDIT.HDFS.IS_ENABLED Flag to enable/disable hdfs audit logging. If the hdfs audit logging is turned off, it will not log any access control to hdfs		Y
XAAUDIT.HDFS.DESTINATION _DIRECTORY HDFS directory where the audit log will be stored	hdfs://__REPLACE__NAME_NODE_HOST:8020/ (format) hdfs://namenode.mycompany.com:8020/ranger/audit/%app-type%/%time:yyyyMMdd%	Y
XAAUDIT.HDFS.LOCAL_BUFFER _DIRECTORY Local directory where the audit log will be saved for intermediate storage	hdfs://__REPLACE__NAME_NODE_HOST:8020/ (format) /var/log/%app-type%/audit	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _DIRECTORY Local directory where the audit log will be archived after it is moved to hdfs	__REPLACE__LOG_DIR%app-type%/audit/archive (format) /var/log/%app-type%/audit/archive	Y
XAAUDIT.HDFS.DESTINATION_FILE hdfs audit file name (format)	%hostname%-audit.log (default)	Y
XAAUDIT.HDFS.DESTINATION _FLUSH_INTERVAL_SECONDS hdfs audit log file writes are flushed to HDFS at regular flush interval	900	Y
XAAUDIT.HDFS.DESTINATION _ROLLOVER_INTERVAL_SECONDS hdfs audit log file is rotated to write to a new file at a rollover interval specified here	86400	Y

Configuration Property Name	Default/Example Value	Required?
XAAUDIT.HDFS.DESTINATION _OPEN_RETRY_INTERVAL_SECONDS hdfs audit log open() call is failed, it will be re-tried at this interval	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FILE Local filename used to store in audit log (format)	%time:yyyyMMdd-HHmm.ss%.log (default)	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FLUSH_INTERVAL_SECONDS Local audit log file writes are flushed to filesystem at regular flush interval	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _ROLLOVER_INTERVAL_SECONDS Local audit log file is rotated to write to a new file at a rollover interval specified here	600	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _MAX_FILE_COUNT The maximum number of local audit log files that will be kept in the archive directory	10	Y
XAAUIDT.SOLR.ENABLE	true	Y
XAAUDIT.SOLR.URL	http://<solr_host>:8886/solr/ ranger_audits	Y
SSL Information (https connectivity to Policy Admin Tool)		
SSL_KEYSTORE_FILE_PATH Java Keystore Path where SSL key for the plug-in is stored. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	/etc/hadoop/conf/ranger-plugin-keystore.jks (default)	Only if SSL is enabled
SSL_KEYSTORE_PASSWORD Password associated with SSL Keystore. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	none (default)	Only if SSL is enabled
SSL_TRUSTSTORE_FILE_PATH Java Keystore Path where the trusted certificates are stored for verifying SSL connection to Policy Admin Tool. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	/etc/hadoop/conf/ranger-plugin-truststore.jks (default)	Only if SSL is enabled
SSL_TRUSTSTORE_PASSWORD Password associated with Truststore file. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	none (default)	Only if SSL is enabled

3. To enable the HDFS plug-in, run the following commands:

```
cd /usr/hdp/<version>/ranger-hdfs-plugin
```

```
./enable-hdfs-plugin.sh
```

4. To restart the service, issue the following commands:

```
su hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh stop
 namenode"
su hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh start
 namenode"
```

5. To confirm that installation and configuration are complete, go to the Audit Tab of the Ranger Admin Console and check Plugins. You should see HDFS listed there.

5.4.2. Installing the Ranger YARN Plug-in

This section describes how to install and enable the Ranger YARN plug-in.

1. The Ranger YARN plug-in is automatically installed when YARN is installed. You can verify this plug-in is present by using the following command:

```
rpm -qa | grep yarn-plugin
ranger_2_4_0_0_2950-yarn-plugin-0.5.0.2.6.0.0-2950.el6.x86_64
```

2. Navigate to `/usr/hdp/<version>/ranger-yarn-plugin`.

```
cd /usr/hdp/<version>/ranger-yarn-plugin
```

3. Edit the following entries in the `install.properties` file.

Table 5.4. Properties to Edit in the install.properties File

Configuration Property Name	Default/Example Value	Required?
Policy Admin Tool		
POLICY_MGR_URL URL for policy admin	http://<FQDN of ranger admin host>:6080	Y
REPOSITORY_NAME The repository name used in Policy Admin Tool for defining policies	yarndev	Y
Audit Database		
SQL_CONNECTOR_JAR Path to SQL connector jar of the DB Flavor selected. The value should be the absolute path including the jar name.	/usr/share/java/mysql-connector-java.jar (default) /usr/share/java/postgresql.jar /usr/share/java/sqljdbc4.jar /usr/share/java/ojdbc6.jar	Y
XAAUDIT.DB.IS_ENABLED Enable or disable database audit logging.	FALSE (default), TRUE	Y
XAAUDIT.DB.FLAVOUR Specifies the type of database used for audit logging (MYSQL,ORACLE)	MYSQL (default)	Y
XAAUDIT.DB.HOSTNAME Hostname of the audit database server	localhost	Y
XAAUDIT.DB.DATABASE_NAME Audit database name	ranger_audit	Y

Configuration Property Name	Default/Example Value	Required?
XAAUDIT.DB.USER_NAME Username used for performing audit log inserts (should be same username used in the ranger-admin installation process)	rangerlogger	Y
XAAUDIT.DB.PASSWORD Database password associated with the above database user - for db audit logging	rangerlogger	Y
HDFS Audit		
XAAUDIT.HDFS.IS_ENABLED Flag to enable/disable hdfs audit logging. If the hdfs audit logging is turned off, it will not log any access control to hdfs		Y
XAAUDIT.HDFS.DESTINATION _DIRECTORY HDFS directory where the audit log will be stored	hdfs:// __REPLACE__NAME_NODE_HOST:8020/ (format) hdfs:// namenode.mycompany.com:8020/ ranger/audit/%app-type%/ %time:yyyyMMdd%	Y
XAAUDIT.HDFS.LOCAL_BUFFER _DIRECTORY Local directory where the audit log will be saved for intermediate storage	hdfs:// __REPLACE__NAME_NODE_HOST:8020/ (format) /var/log/%app-type%/ audit	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _DIRECTORY Local directory where the audit log will be archived after it is moved to hdfs	__REPLACE__LOG_DIR%app-type %/audit/archive (format) /var/log/ %app-type%/audit/archive	Y
XAAUDIT.HDFS.DESTINATION_FILE hdfs audit file name (format)	%hostname%-audit.log (default)	Y
XAAUDIT.HDFS.DESTINATION _FLUSH_INTERVAL_SECONDS hdfs audit log file writes are flushed to HDFS at regular flush interval	900	Y
XAAUDIT.HDFS.DESTINATION _ROLLOVER_INTERVAL_SECONDS hdfs audit log file is rotated to write to a new file at a rollover interval specified here	86400	Y
XAAUDIT.HDFS.DESTINATION _OPEN_RETRY_INTERVAL_SECONDS hdfs audit log open() call is failed, it will be re-tried at this interval	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FILE Local filename used to store in audit log (format)	%time:yyyyMMdd-HHmm.ss%.log (default)	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FLUSH_INTERVAL_SECONDS Local audit log file writes are flushed to filesystem at regular flush interval	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _ROLLOVER_INTERVAL_SECONDS Local audit log file is rotated to write to a new file at a rollover interval specified here	600	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _MAX_FILE_COUNT The maximum number of local audit log files that will be kept in the archive directory	10	Y

Configuration Property Name	Default/Example Value	Required?
XAAUIDT.SOLR.ENABLE	true	Y
XAAUDIT.SOLR.URL	http://<solr_host>:8886/solr/ ranger_audits	Y
SSL Information (https connectivity to Policy Admin Tool)		
SSL_KEYSTORE_FILE_PATH Java Keystore Path where SSL key for the plug-in is stored. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	/etc/hadoop/conf/ranger-plugin-keystore.jks (default)	Only if SSL is enabled
SSL_KEYSTORE_PASSWORD Password associated with SSL Keystore. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	none (default)	Only if SSL is enabled
SSL_TRUSTSTORE_FILE_PATH Java Keystore Path where the trusted certificates are stored for verifying SSL connection to Policy Admin Tool. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	/etc/hadoop/conf/ranger-plugin-truststore.jks (default)	Only if SSL is enabled
SSL_TRUSTSTORE_PASSWORD Password associated with Truststore file. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	none (default)	Only if SSL is enabled

4. Enable the YARN plug-in by running the following commands:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-amd64
```

or

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-amd64

./enable-yarn-plugin.sh
```

5. Make sure HADOOP_YARN_HOME and HADOOP_LIBEXEC_DIR are set.

```
export HADOOP_YARN_HOME=/usr/hdp/current/hadoop-yarn-nodemanager/
export HADOOP_LIBEXEC_DIR=/usr/hdp/current/hadoop-client/libexec/
```

6. Enter the following commands to stop/start the ResourceManager on all of your Resource Manager hosts.

```
su yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.sh
 stop resourcemanager"
su yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.sh
 start resourcemanager"
ps -ef | grep -i resourcemanager
```

7. Enter the following command to stop/start the NodeManager on all of your NodeManager hosts.

```
su yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh
 stop nodemanager"
su yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh
 start nodemanager"
ps -ef | grep -i nodemanager
```

8. Create the default repo for YARN with the proper configuration specifying the same resource-based service name as in step 3.

9. You can verify the plug-in is communicating to Ranger admin via the Audit/plugins tab.

10. **Optional**: If Wire Encryption is enabled after Ranger authorization is set up for YARN, follow these steps to update the resource manager URL in Ranger:

   a. Login to Ranger Admin as an admin user.

   b. Click 'Edit' for the YARN service. The service name is something like cl1_yarn (where 'cl1' is the name of the cluster).

   c. Update property 'YARN REST URL', with the https URL for the resource manager.

   d. Click 'Save.'

5.4.3. Installing the Ranger Kafka Plug-in

This section describes how to install and enable the Ranger Kafka plug-in.

1. The Ranger Kafka plug-in is automatically installed when Kafka is installed. You can verify this plug-in is present by using the following command:

```
rpm -qa | grep kafka-plugin
ranger_2_4_0_0_2950-kafka-plugin-0.5.0.2.6.0-2950.el6.x86_64
```

2. Navigate to `/usr/hdp/<version>/ranger-kafka-plugin`.

```
cd /usr/hdp/<version>/ranger-kafka-plugin
```

3. Edit the following entries in the `install.properties` file.

Table 5.5. Properties to Edit in the install.properties File

Configuration Property Name	Default/Example Value	Required?
Policy Admin Tool		
COMPONENT_INSTALL_DIR_NAME	/usr/hdp/2.6.0.0-2950/kafka	Y
POLICY_MGR_URL URL for policy admin	http://<FQDN of ranger admin host>:6080	Y
REPOSITORY_NAME The repository name used in Policy Admin Tool for defining policies	kafkadev	Y
Audit Database		
SQL_CONNECTOR_JAR Path to SQL connector jar of the DB Flavor selected. The value should be the	/usr/share/java/mysql-connector-java.jar (default) /usr/share/java/postgresql.jar	Y

Configuration Property Name	Default/Example Value	Required?
absolute path including the jar name.	/usr/share/java/sqljdbc4.jar /usr/share/java/ojdbc6.jar	
XAAUDIT.DB.IS_ENABLED Enable or disable database audit logging.	FALSE (default), TRUE	Y
XAAUDIT.DB.FLAVOUR Specifies the type of database used for audit logging (MYSQL,ORACLE)	MYSQL (default)	Y
XAAUDIT.DB.HOSTNAME Hostname of the audit database server	localhost	Y
XAAUDIT.DB.DATABASE_NAME Audit database name	ranger_audit	Y
XAAUDIT.DB.USER_NAME Username used for performing audit log inserts (should be same username used in the ranger-admin installation process)	rangerlogger	Y
XAAUDIT.DB.PASSWORD Database password associated with the above database user - for db audit logging	rangerlogger	Y
XAAUDIT.SOLR.ENABLE	true	Y
XAAUDIT.SOLR.URL	http://<solr_host>:8886/solr/ranger_audits	Y
HDFS Audit		
XAAUDIT.HDFS.IS_ENABLED Flag to enable/disable hdfs audit logging. If the hdfs audit logging is turned off, it will not log any access events to hdfs		Y
XAAUDIT.HDFS.DESTINATION _DIRECTORY HDFS directory where the audit log will be stored	hdfs://__REPLACE__NAME_NODE_HOST:8020/ (format) hdfs://namenode.mycompany.com:8020/ranger/audit/%app-type%/%time:yyyyMMdd%	Y
XAAUDIT.HDFS.LOCAL_BUFFER _DIRECTORY Local directory where the audit log will be saved for intermediate storage	hdfs://__REPLACE__NAME_NODE_HOST:8020/ (format) /var/log/%app-type%/audit	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _DIRECTORY Local directory where the audit log will be archived after it is moved to hdfs	__REPLACE__LOG_DIR%app-type%/audit/archive (format) /var/log/%app-type%/audit/archive	Y
XAAUDIT.HDFS.DESTINATION_FILE hdfs audit file name (format)	%hostname%-audit.log (default)	Y
XAAUDIT.HDFS.DESTINATION _FLUSH_INTERVAL_SECONDS hdfs audit log file writes are flushed to HDFS at regular flush interval	900	Y
XAAUDIT.HDFS.DESTINATION _ROLLOVER_INTERVAL_SECONDS hdfs audit log file is rotated to write to a new file at a rollover interval specified here	86400	Y
XAAUDIT.HDFS.DESTINATION _OPEN_RETRY_INTERVAL_SECONDS	60	Y

Configuration Property Name	Default/Example Value	Required?
hdfs audit log open() call is failed, it will be re-tried at this interval		
XAAUDIT.HDFS.LOCAL_BUFFER _FILE Local filename used to store in audit log (format)	%time:yyyyMMdd-HHmm.ss%.log (default)	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FLUSH_INTERVAL_SECONDS Local audit log file writes are flushed to filesystem at regular flush interval	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _ROLLOVER_INTERVAL_SECONDS Local audit log file is rotated to write to a new file at a rollover interval specified here	600	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _MAX_FILE_COUNT The maximum number of local audit log files that will be kept in the archive directory	10	Y
SSL Information (https connectivity to Policy Admin Tool)		
SSL_KEYSTORE_FILE_PATH Java Keystore Path where SSL key for the plug-in is stored. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	/etc/hadoop/conf/ranger-plugin-keystore.jks (default)	Only if SSL is enabled
SSL_KEYSTORE_PASSWORD Password associated with SSL Keystore. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	none (default)	Only if SSL is enabled
SSL_TRUSTSTORE_FILE_PATH Java Keystore Path where the trusted certificates are stored for verifying SSL connection to Policy Admin Tool. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	/etc/hadoop/conf/ranger-plugin-truststore.jks (default)	Only if SSL is enabled
SSL_TRUSTSTORE_PASSWORD Password associated with Truststore file. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used	none (default)	Only if SSL is enabled

4. Enable the Kafka plug-in by running the following commands:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-amd64
./enable-kafka-plugin.sh
```

5. Enter the following commands to stop/start the Kafka service.

```
su kafka -c "/usr/hdp/current/kafka-broker/bin/kafka stop"
su kafka -c "/usr/hdp/current/kafka-broker/bin/kafka start"
```

6. Create the default repo for Kafka with the proper configuration specifying the same resource-based service name as in step 3.

7. You can verify the plug-in is communicating to Ranger admin via the Audit/plugins tab.

8. If the plug-in is not able to communicate with Ranger admin, check the property `authorizer.class.name` in `/usr/hdp/2.6.0.0-2950/kafka/config/server.properties`. The value of the authorizer.class.name should be org.apache.ranger.authorization.kafka.authorizer.RangerKafkaAuthorizer.

5.4.4. Installing the Ranger HBase Plug-in

The Ranger HBase Plug-in integrates with HBase to enforce authorization policies.

This section describes how to install the HBase plug-in:

1. Create an HBase resource-based service

2. Install the HBase plug-in and configure related HBase properties

3. Enable the HBase plug-in

4. Restart HBase

Install the HBase Plug-in

1. Create an HBase resource-based service in the Ranger Policy Manager. To do this, complete the HBase Create Service screen, as described in the Configure an HBase Service section of the *Hadoop Security Guide*.

   Make a note of the name you gave to this resource-based service; you will use it again during HBase plug-in setup.

2. At all servers where the HBase Master and RegionServers are installed, install and configure the HBase plug-in, as follows:

   a. Go to the home directory of the HBase plug-in:

   ```
   cd /usr/hdp/<version>/ranger-hbase-plugin
   ```

   b. Edit the following HBase-related properties in the `install.properties` file:

   Table 5.6. HBase Properties to Edit in the `install.properties` file

Configuration Property Name	Default/Example Value	Required?
Policy Admin Tool		
POLICY_MGR_URL URL for policy admin	*http://policymanager.xasecure.net:6080*	Y
REPOSITORY_NAME The repository name used in Policy Admin Tool for defining policies	hbasedev	Y
Audit Database		
SQL_CONNECTOR_JAR Path to SQL connector jar of the DB Flavor selected. The value should be the absolute path including the jar name.	/usr/share/java/mysql-connector-java.jar (default) /usr/share/java/postgresql.jar	Y

Configuration Property Name	Default/Example Value	Required?
	/usr/share/java/sqljdbc4.jar /usr/share/java/ojdbc6.jar	
XAAUDIT.DB.IS_ENABLED Enable or disable database audit logging. *Note*: If this property is set to FALSE, Ranger will not store audit logs in the audit DB, and audit logs will not be visible in the Ranger UI. If you would like to access audit logs from the UI, set this value to TRUE.	FALSE (default)	Y
XAAUDIT.DB.FLAVOUR Specifies the type of database used for audit logging (MYSQL,ORACLE)	MYSQL (default)	Y
XAAUDIT.DB.HOSTNAME Hostname of the audit database server	localhost	Y
XAAUDIT.DB.DATABASE_NAME Audit database name	ranger_audit	Y
XAAUDIT.DB.USER_NAME Username used for performing audit log inserts (should be same username used in the ranger-admin installation process)	rangerlogger	Y
XAAUDIT.DB.PASSWORD Database password associated with the above database user - for db audit logging	rangerlogger	Y
HDFS Audit		
XAAUDIT.HDFS.IS_ENABLED Flag to enable/disable hdfs audit logging. If the hdfs audit logging is turned off, it will not log any access control to hdfs	TRUE	Y
XAAUDIT.HDFS.DESTINATION _DIRECTORY HDFS directory where the audit log will be stored	*hdfs://__REPLACE__NAME_NODE_HOST:8020/ ranger/audit/%app-type%/ %time:yyyyMMdd% (format) hdfs:// namenode.mycompany.com:8020/ ranger/audit/%app-type%/ %time:yyyyMMdd%*	Y
XAAUDIT.HDFS.LOCAL _BUFFER_DIRECTORY Local directory where the audit log will be saved for intermediate storage	__REPLACE__LOG_DIR/%app-type %/audit (format) /var/tmp/%app-type%/audit	Y
XAAUDIT.HDFS.LOCAL _ARCHIVE_DIRECTORY Local directory where the audit log will be archived after it is moved to hdfs	__REPLACE__LOG_DIR/%app-type %/audit/archive (format) /var/ tmp/%app-type%/audit/archive	Y
XAAUDIT.HDFS.DESTINATION_FILE HDFS audit file name (format)	%hostname%-audit.log (default)	Y
XAAUDIT.HDFS.DESTINATION _FLUSH_INTERVAL_SECONDS HDFS audit log file writes are flushed to HDFS at regular flush interval	900	Y

Configuration Property Name	Default/Example Value	Required?
XAAUDIT.HDFS.DESTINATION _ROLLOVER_INTERVAL_SECONDS HDFS audit log file is rotated to write to a new file at a rollover interval specified here	86400	Y
XAAUDIT.HDFS.DESTINATION _OPEN_RETRY_INTERVAL_SECONDS If HDSF audit log open() call fails, it will be re-tried at this interval	60	Y
XAAUDIT.HDFS.LOCAL _BUFFER_FILE Local filename used to store in audit log (format)	%time:yyyyMMdd-HHmm.ss%.log (default)	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FLUSH_INTERVAL_SECONDS Interval that local audit log file writes are flushed to filesystem	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _ROLLOVER_INTERVAL_SECONDS Interval that local audit log file is rolled over (rotated to write to a new file)	600	Y
XAAUDIT.SOLR.ENABLE	true	Y
XAAUDIT.SOLR.URL	http://<solr_host>:8886/solr/ ranger_audits	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _MAX_FILE_COUNT The maximum number of local audit log files will be kept in the archive directory	10	Y
SSL_KEYSTORE_FILE_PATH Java Keystore Path where SSL key for the plug-in is stored. Used only if SSL is enabled between Policy Admin Tool and Plugin. If SSL is not enabled, leave the default value as it is (should not be set as EMPTY).	*/etc/hbase/conf/ranger-plugin-keystore.jks* (default)	Y, if SSL is enabled
SSL_KEYSTORE_PASSWORD Password associated with SSL Keystore. Used only if SSL is enabled between Policy Admin Tool and Plugin. If SSL is not Enabled, leave the default value as it is (should not be set as EMPTY).	myKeyFilePassword (default)	Y, if SSL is enabled
SSL_TRUSTSTORE_FILE_PATH Java Keystore Path where the trusted certificates are stored for verifying SSL connection to Policy Admin Tool. Used only if SSL is enabled between Policy Admin Tool and Plugin. If SSL is not enabled, leave the default value as it is (should not be set as EMPTY).	*/etc/hbase/conf/ranger-plugin-truststore.jks* (default)	Y, if SSL is enabled
SSL_TRUSTSTORE_PASSWORD Password associated with Truststore file. Used only if SSL is enabled between Policy Admin Tool and Plugin. If SSL is not Enabled, leave the default value as it is (should not be set as EMPTY).	changeit (default)	Y, if SSL is enabled
HBase GRANT/REVOKE Commands		

Configuration Property Name	Default/Example Value	Required?
UPDATE_XAPOLICIES_ON_GRANT_RE VOKE Provide ability for XAAgent to update the policies based on the GRANT/REVOKE commands from the HBase client	TRUE (default)	Y

3. To enable the HBase plug-in, enter the following commands:

```
cd /usr/hdp/<version>l/ranger-hbase-plugin
```

```
./enable-hbase-plugin.sh
```

4. Restart HBase.

5. To confirm that the HBase plug-in installation and configuration are complete, go to the Audit Tab of the Ranger Admin Console and check Plugins. You should see HBase listed there.

5.4.5. Installing the Ranger Hive Plug-in

The Ranger Hive plug-in integrates with Hive to enforce authorization policies.

> **Note**
>
> The Ranger plugin for HIve only needs to be set up for HiveServer2. For Hive clients, it is recommended that you protect data using HDFS policies in Ranger. Do not install or set up Ranger plugins on individual Hive client machines.

This section describes how to install the Ranger Hive plug-in:

1. Create a Hive resource-based service .

2. Install the Hive plug-in and configure related Hive properties.

3. Enable the Hive plug-in.

4. Restart Hive.

Install the Hive Plug-in

1. Create a Hive resource-based service. To create the Hive resource-based service, complete the Hive Create Service screen as described in the Configure a Hive Service section of the *Hadoop Security Guide*.

   Make a note of the name you gave to this resource-based service; you will need to use it again during Hive plug-in setup.

2. At the server where HiveServer2 is installed, install the Hive plug-in:

   • Go to the home directory of the Hive plug-in:

   ```
   cd /usr/hdp/<version>/ranger-hive-plugin
   ```

   • Edit the following Hive-related properties in the install.properties file:

Table 5.7. Hive-Related Properties to Edit in the install.properties File

Configuration Property Name	Default/Example Value	Required?
Policy Admin Tool		
POLICY_MGR_URL URL for policy admin	*http://policymanager.xasecure.net:6080*	Y
REPOSITORY_NAME The repository name used in Policy Admin Tool for defining policies	hivedev	Y
Audit Database		
SQL_CONNECTOR_JAR Path to SQL connector jar of the DB Flavor selected. The value should be the absolute path including the jar name.	/usr/share/java/mysql-connector-java.jar (default) /usr/share/java/postgresql.jar /usr/share/java/sqljdbc4.jar /usr/share/java/ojdbc6.jar	Y
XAAUDIT.DB.IS_ENABLED Enable or disable database audit logging. *Note*: If this property is set to FALSE, Ranger will not store audit logs in the audit DB, and audit logs will not be visible in the Ranger UI. If you would like to access audit logs from the UI, set this value to TRUE.	FALSE (default) TRUE	Y
XAAUDIT.DB.FLAVOUR Specifies the type of database used for audit logging (MYSQL,ORACLE)	MYSQL (default)	Y
XAAUDIT.DB.HOSTNAME Hostname of the audit database server	localhost	Y
XAAUDIT.DB.DATABASE_NAME Audit database name	ranger_audit	Y
XAAUDIT.DB.USER_NAME Username used for performing audit log inserts (should be same username used in the ranger-admin installation process)	rangerlogger	Y
XAAUDIT.DB.PASSWORD database password associated with the above database user - for db audit logging	rangerlogger	Y
HDFS Audit		
XAAUDIT.HDFS.IS_ENABLED Flag to enable/disable hdfs audit logging.If the hdfs audit logging is turned off, it will not log any access control to hdfs		Y
XAAUDIT.HDFS.DESTINATION _DIRECTORY HDFS directory where the audit log will be stored	*hdfs://__REPLACE__NAME_NODE_HOST:8020/ranger/audit/%app-type%/%time:yyyyMMdd%* (format) *hdfs://namenode.mycompany.com:8020/ranger/audit/%app-type%/%time:yyyyMMdd%*	Y

Configuration Property Name	Default/Example Value	Required?
XAAUDIT.HDFS.LOCAL_BUFFER _DIRECTORY Local directory where the audit log will be saved for intermediate storage	__REPLACE__LOG_DIR/%app-type %/audit (format) /var/tmp/%app-type%/audit	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _DIRECTORY Local directory where the audit log will be archived after it is moved to hdfs	__REPLACE__LOG_DIR/%app-type %/audit (format) /var/tmp/%app-type%/audit/archive	Y
XAAUDIT.HDFS.DESTINATION_FILE hdfs audit file name (format)	%hostname%-audit.log (default)	Y
XAAUDIT.HDFS.DESTINATION _FLUSH_INTERVAL_SECONDS hdfs audit log file writes are flushed to HDFS at regular flush interval	900	Y
XAAUDIT.HDFS.DESTINATION _ROLLOVER_INTERVAL_SECONDS hdfs audit log file is rotated to write to a new file at a rollover interval specified here	86400	Y
XAAUDIT.HDFS.DESTINATION _OPEN_RETRY_INTERVAL_SECONDS If hdfs audit log open() call is failed, it will be re-tried at this interval	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FILE Local filename used to store in audit log (format)	%time:yyyyMMdd-HHmm.ss%.log (default)	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FLUSH_INTERVAL_SECONDS Local audit log file writes are flushed to filesystem at regular flush interval	60	Y
X**AAUDIT.HDFS.LOCAL_BUFFER _ROLLOVER_INTERVAL_SECONDS** Local audit log file is rotated to write to a new file at a rollover interval specified here	600	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _MAX_FILE_COUNT The maximum number of local audit log files that will be kept in the archive directory	10	Y
XAAUDIT.SOLR.ENABLE	true	Y
XAAUDIT.SOLR.URL	http://<solr_host>:8886/solr/ ranger_audits	Y
SSL Information (https connectivity to Policy Admin Tool)		
SSL_KEYSTORE_FILE_PATH Java Keystore Path where SSL key for the plug-in is stored. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	/etc/hive/conf/ranger-plugin-keystore.jks (default)	If SSL is enabled
SSL_KEYSTORE_PASSWORD Password associated with SSL Keystore. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	none (default)	If SSL is enabled

Configuration Property Name	Default/Example Value	Required?
SSL_TRUSTSTORE_FILE_PATH Java Keystore Path where the trusted certificates are stored for verifying SSL connection to Policy Admin Tool. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	*/etc/hive/conf/ranger-plugin-truststore.jks* (default)	If SSL is enabled
SSL_TRUSTSTORE_PASSWORD Password associated with Truststore file. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	none (default)	If SSL is enabled
Hive GRANT/REVOKE Command Handling		
UPDATE_XAPOLICIES_ON_GRANT _REVOKE Provide ability for XAAgent to update the policies based on the grant/revoke commands from the Hive beeline client	TRUE (default)	Y

3. To enable the Hive plug-in, enter the following commands:

```
cd /usr/hdp/<version>/ranger-hive-plugin
```

```
./enable-hive-plugin.sh
```

4. Restart Hive.

5. To confirm that the Hive plug-in installation and configuration are complete, go to the Audit Tab of the Ranger Admin Console and check Plugins. You should see Hive listed there.

5.4.6. Installing the Ranger Knox Plug-in

The Ranger Knox plug-in integrates with Knox to enforce authorization policies.

This section describes how to install the Knox plug-in:

1. Create a Knox resource-based service.

2. Install the Knox plug-in and configure related Hive properties.

3. Enable the Knox plug-in.

4. Restart Knox.

Instructions assume that Knox has already been installed, as described in "*Installing Knox*."

Install the Knox Plug-in

1. Create a Knox resource-based service. To do this, complete the Knox Create Service screen as described in the Configure a Knox Service section of the *Hadoop Security Guide*.

2. Set the URL to *https://knox_host:8443/gateway/admin/api/v1/topologies*, where knox_host is the full-qualified name of your Knox host machine.

3. Make a note of the name you gave to this resource-based service; you will need to use it again during Knox plug-in setup.

4. At all servers where Knox Gateway is installed, install the Knox plug-in:

   a. Go to the home directory of the Knox plug-in:

   ```
   cd /usr/hdp/<version>/ranger-knox-plugin
   ```

   b. Edit the following Knox-related properties in the `install.properties` file:

   ### Table 5.8. Knox-Related Properties to Edit in the install.properties File

Configuration Property Name	Default/Example Value	Mandatory?
Policy Admin Tool		
POLICY_MGR_URL URL for policy admin	*http://policymanager.xasecure.net:6080*	Y
REPOSITORY_NAME The repository name used in Policy Admin Tool for defining policies	knoxdev	Y
Knox Component Installation		
KNOX_HOME Home directory where Knox software is installed	*/usr/hdp/current/knox*	Y
Audit Database		
SQL_CONNECTOR_JAR Path to SQL connector jar of the DB Flavor selected. The value should be the absolute path including the jar name.	*/usr/share/java/mysql-connector-java.jar*	Y
XAAUDIT.DB.IS_ENABLED Enable or disable database audit logging. *Note*: If this property is set to FALSE, Ranger will not store audit logs in the audit DB, and audit logs will not be visible in the Ranger UI. If you would like to access audit logs from the UI, set this value to TRUE.	true	Y
XAAUDIT.DB.FLAVOUR Specifies the type of database used for audit logging (MYSQL,ORACLE)	MYSQL	Y
XAAUDIT.DB.HOSTNAME Hostname of the audit database server	localhost	Y
XAAUDIT.DB.DATABASE_NAME Audit database name	ranger_audit	Y
XAAUDIT.DB.USER_NAME Username used for performing audit log inserts (should be same username used in the ranger-admin installation process)	rangerlogger	Y
XAAUDIT.DB.PASSWORD database password associated with the	rangerlogger	Y

Configuration Property Name	Default/Example Value	Mandatory?
above database user - for db audit logging		
HDFS Audit		
XAAUDIT.HDFS.IS_ENABLED Flag to enable/disable hdfs audit logging. If the hdfs audit logging is turned off, it will not log any access control to hdfs.		Y
XAAUDIT.HDFS.DESTINATION _DIRECTORY HDFS directory where the audit log will be stored	*hdfs:// namenode.mycompany.com:8020/ ranger/audit/%app-type%/ %time:yyyyMMdd%*	Y
XAAUDIT.HDFS.LOCAL_BUFFER _DIRECTORY Local directory where the audit log will be saved for intermediate storage	*/var/tmp/%app-type%/audit*	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _DIRECTORY Local directory where the audit log will be archived after it is moved to hdfs	*/var/tmp/%app-type%/audit/archive*	Y
XAAUDIT.HDFS.DESTINATION_FILE hdfs audit file name (format)	%hostname%-audit.log	Y
XAAUDIT.HDFS.DESTINATION _FLUSH_INTERVAL_SECONDS hdfs audit log file writes are flushed to HDFS at regular flush interval	900	Y
XAAUDIT.HDFS.DESTINATION _ROLLOVER_INTERVAL_SECONDS hdfs audit log file is rotated to write to a new file at a rollover interval specified here	86400	Y
XAAUDIT.HDFS.DESTINATION _OPEN_RETRY_INTERVAL_SECONDS If hdfs audit log open() call is failed, it will be re-tried at this interval	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FILE Local filename used to store in audit log (format)	%time:yyyyMMdd-HHmm.ss%.log	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FLUSH_INTERVAL_SECONDS Local audit log file writes are flushed to filesystem at regular flush interval	60	Y
XAAUDIT.HDFS.LOCAL_BUFFER _ROLLOVER_INTERVAL_SECONDS Local audit log file is rotated to write to a new file at a rollover interval specified here	600	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _MAX_FILE_COUNT The maximum number of local audit log files will be kept in the archive directory	10	Y
XAAUDIT.SOLR.ENABLE	true	Y
XAAUDIT.SOLR.URL	http://<solr_host>:8886/solr/ ranger_audits	Y
SSL (https connectivity to Policy Admin Tool)		

Configuration Property Name	Default/Example Value	Mandatory?
SSL_KEYSTORE_FILE_PATH Java Keystore Path where SSL key for the plug-in is stored. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	*/etc/knox/conf/ranger-plugin-keystore.jks*	If SSL is enabled
SSL_KEYSTORE_PASSWORD Password associated with SSL Keystore. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	myKeyFilePassword	If SSL is enabled
SSL_TRUSTSTORE_FILE_PATH Java Keystore Path where the trusted certificates are stored for verifying SSL connection to Policy Admin Tool. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	*/etc/knox/conf/ranger-plugin-truststore.jks*	If SSL is enabled
SSL_TRUSTSTORE_PASSWORD Password associated with Truststore file. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	changeit	If SSL is enabled

5. To enable the Knox plug-in, enter the following commands:

```
cd /usr/hdp/<version>/ranger-knox-plugin
./enable-knox-plugin.sh
```

6. Restart the Knox Gateway:

```
su knox -c "/usr/hdp/current/knox-server/bin/gateway.sh stop"
su knox -c "/usr/hdp/current/knox-server/bin/gateway.sh start"
```

7. Optionally, add a new user by following these steps. This is required only if you want users to verify the communication between the Ranger Knox plug-in and Ranger `admin`.

   a. Add the following information in the `usertest` block in the `/usr/hdp/current/knox-server/templates/users.ldif` file, assuming that Knox is using the demo LDAP. If Knox is configured with a different LDAP, you need to add the information in the `users.ldif` file at that location.

      • # entry for sample user usertest

      • dn: uid=usertest,ou=people,dc=hadoop,dc=apache,dc=org

      • objectclass:top

      • objectclass:person

      • objectclass:organizationalPerson

- objectclass:inetOrgPerson

- cn: usertest

- sn: usertest

- uid: usertest

- userPassword:usertest-password

b. Restart ldap. Enter the following command:

```
./usr/hdp/current/knox-server/bin/ldap.sh stop / start
```

c. Verify in which topology the Knox plugin is enabled and then run the command accordingly. Typically, the topology is `admin`.

d. Issue the **curl** command to check policy enforcement for Knox. Typically, the `topology` in the command is `admin`.

```
curl -iku admin:admin-password -X GET 'https://<knox_host>:8443/gateway/
<topology>/webhdfs/v1?op=LISTSTATUS'
```

e. Create a certificate for the test connection to be successful. Follow these steps to import a Knox SSL certificate in truststore used by Ranger admin:

   i. Login to the machine running Knox.

   ii. Export the Knox certificate:

   ```
   cd $GATEWAY_HOME/data/security/keystores
   keytool -exportcert -alias gateway-identity -keystore gateway.jks -file
   knox.crt
   ```

   Typically `$GATEWAY_HOME/data/security/keystores` is `/usr/hdp/current/knox-server/data/security/keystores` on a Linux machine.

   iii. Copy knox.crt onto the machine running Ranger admin to a working directory, for example, `/etc/ranger/admin`.

   iv. Replicate cacerts bundled with the JDK:

   ```
   cd /etc/ranger/admin
   cp <JDK_HOME>/jre/lib/security/cacerts cacertswithknox
   ```

   v. Import the Knox certificate into the replicated new keystore.

   ```
   keytool -import -trustcacerts -file <knox.crt
    created above>  -alias knox  -keystore cacertswithknox
   password: changeit
   ```

   vi. Edit /usr/hdp/current/ranger-admin/ews/ranger-admin-services.sh and add the parameter `-Djavax.net.ssl.trustStore=<path to the cacertswithknox>` to the `JAVA_OPTS` parameter in the script.

   vii.Restart Ranger Admin.

8. To confirm that the Knox plug-in installation and configuration are complete, go to the Audit Tab of the Ranger Admin Console and check Plugins. You should see Knox listed there.

5.4.7. Installing the Ranger Storm Plug-in

The Ranger Storm plug-in integrates with Storm to enforce authorization policies.

This section describes how to perform the following administrative tasks: It assumes that Storm has already been installed, as described earlier in this guide.

1. Create a Storm resource-based service.

2. Install the Storm plug-in and configure related Storm properties.

3. Enable the Storm plug-in.

4. Restart Storm.

Install the Storm Plug-in

1. Create a Storm resource-based service, as described in the Configure a Storm Service section of the *Hadoop Security Guide*.

   Make a note of the name you gave to this resource-based service; you will need to use it again during Storm plug-in setup.

2. On the Nimbus server, install the Storm plug-in:

   a. Go to the home directory of the Storm plug-in:

   ```
   cd /usr/hdp/<version>/ranger-storm-plugin
   ```

   b. Edit the following Storm-related properties in the `install.properties` file:

   Table 5.9. Storm-Related Properties to Edit in the `install.properties` file

Configuration Property Name	Default/Example Value	Mandatory?
Policy Admin Tool		
POLICY_MGR_URL URL for policy admin	*http:// policymanager.xasecure.net:6080*	Y
REPOSITORY_NAME The repository name used in Policy Admin Tool for defining policies	stormdev	Y
Audit Database		
SQL_CONNECTOR_JAR Path to SQL connector jar of the DB Flavor selected. The value should be the absolute path including the jar name.	/usr/share/java/mysql-connector-java.jar (default) /usr/share/java/postgresql.jar /usr/share/java/sqljdbc4.jar /usr/share/java/ojdbc6.jar	Y
XAAUDIT.DB.IS_ENABLED Enable or disable database audit logging.	false (default) true	Y

Configuration Property Name	Default/Example Value	Mandatory?
Note: If this property is set to FALSE, Ranger will not store audit logs in the audit DB, and audit logs will not be visible in the Ranger UI. If you would like to access audit logs from the UI, set this value to TRUE.		
XAAUDIT.DB.FLAVOUR Specifies the type of database used for audit logging (MYSQL,ORACLE, PostgreSQL 8.4.2, SQL Server 2012)	MYSQL (default)	Y
XAAUDIT.DB.HOSTNAME Hostname of the audit database server	localhost	Y
XAAUDIT.DB.DATABASE_NAME Audit database name	ranger_audit	Y
XAAUDIT.DB.USER_NAME Username used for performing audit log inserts (should be same username used in the ranger-admin installation process)	rangerlogger	Y
XAAUDIT.DB.PASSWORD Database password associated with the above database user - for db audit logging	rangerlogger	Y
HDFS Audit		
XAAUDIT.HDFS.IS_ENABLED Flag to enable/disable hdfs audit logging. If the hdfs audit logging is turned off, it will not log any access control to hdfs.	false	Y
XAAUDIT.HDFS.DESTINATION _DIRECTORY HDFS directory where the audit log will be stored	*hdfs:// __REPLACE__NAME_NODE_HOST:8020/ ranger/audit/%app-type%/ %te:yyyyMMdd%* (format) *hdfs:// namenode.mycompany.com:8020/ ranger/audit/%app-type%/ %time:yyyyMMdd%*	Y
XAAUDIT.HDFS.LOCAL_BUFFER _DIRECTORY Local directory where the audit log will be saved for intermediate storage	__REPLACE__LOG_DIR/%app-type %/audit (format) /var/log/%app-type%/audit	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _DIRECTORY Local directory where the audit log will be archived after it is moved to hdfs	__REPLACE__LOG_DIR/%app-type %/audit/archive (format) /var/log/ %app-type%/audit/archive	Y
XAAUDIT.HDFS.DESTINATION_FILE hdfs audit file name (format)	%hostname%-audit.log (default)	Y
XAAUDIT.HDFS.DESTINATION _FLUSH_INTERVAL_SECONDS hdfs audit log file writes are flushed to HDFS at regular flush interval	900 (default)	Y
XAAUDIT.HDFS.DESTINATION _ROLLOVER_INTERVAL_SECONDS hdfs audit log file is rotated to write to a new file at a rollover interval specified here	86400 (default)	Y

Configuration Property Name	Default/Example Value	Mandatory?
XAAUDIT.HDFS.DESTINATION _OPEN_RETRY_INTERVAL_SECONDS If hdfs audit log open() call is failed, it will be re-tried at this interval	60 (default)	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FILE Local filename used to store in audit log (format)	%time:yyyyMMdd-HHmm.ss%.log (default)	Y
XAAUDIT.HDFS.LOCAL_BUFFER _FLUSH_INTERVAL_SECONDS Local audit log file writes are flushed to filesystem at regular flush interval	60 (default)	Y
XAAUDIT.HDFS.LOCAL_BUFFER _ROLLOVER_INTERVAL_SECONDS Local audit log file is rotated to write to a new file at a rollover interval specified here	600 (default)	Y
XAAUDIT.HDFS.LOCAL_ARCHIVE _MAX_FILE_COUNT The maximum number of local audit log files will be kept in the archive directory	10 (default)	Y
XAAUDIT.SOLR.ENABLE	true	Y
XAAUDIT.SOLR.URL	http://<solr_host>:8886/solr/ ranger_audits	Y
SSL Information (https connectivity to policy Admin Tool)		
SSL_KEYSTORE_FILE_PATH Java Keystore Path where SSL key for the plug-in is stored. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	/etc/storm/conf/ranger-plugin-keystore.jks (default)	If SSL is enabled
SSL_KEYSTORE_PASSWORD Password associated with SSL Keystore. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	myKeyFilePassword (default)	If SSL is enabled
SSL_TRUSTSTORE_FILE_PATH Java Keystore Path where the trusted certificates are stored for verifying SSL connection to Policy Admin Tool. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	/etc/storm/conf/ranger-plugin-truststore.jks (default)	If SSL is enabled
SSL_TRUSTSTORE_PASSWORD Password associated with Truststore file. Is used only if SSL is enabled between Policy Admin Tool and Plugin; If SSL is not Enabled, leave the default value as it is - do not set as EMPTY if SSL not used.	changeit (default)	If SSL is enabled

3. Enable the Storm plug-in by entering the following commands:

```
cd /usr/hdp/<version>/ranger-storm-plugin
```

```
./enable-storm-plugin.sh
```

4. Restart Storm.

5. To confirm that the Storm plug-in installation and configuration are complete, go to the Audit Tab of the Ranger Admin Console and check Plugins. You should see Storm listed there.

5.5. Installing Ranger in a Kerberized Environment

This section focuses specifically on installing Ranger in a kerberized environment.

> ⚠️ **Important**
>
> The steps in this section apply only to manual installation of Ranger services and plug-ins in a kerberized environment.

5.5.1. Creating Keytab and Principals

5.5.1.1. Before You Begin

Perform the following initial checks before starting your installation:

1. Login as user `ranger`.

   If user ranger is not found, create it using the **useradd** command, for example, **su - ranger**.

2. Use the **kinit** command to check for HTTP principal:

   ```
   kinit -kt <HTTP keytab path> HTTP/<FQDN_OF_Ranger_Admin_Cluster>@<REALM>
   ```

   After running the **kinit** command, there should not be any errors. You can use the **klist** command to verify that your **kinit** command was successful.

3. Use the **kdestroy** command to destroy your active Kerberos authorization tickets by overwriting and deleting the credentials cache that contains them:

   ```
   kdestroy
   ```

5.5.1.2. Prepare Ranger Admin

Follow these instructions to prepare Ranger admin:

1. Create `rangeradmin/<FQDN_of_Ranger_Admin>@<REALM>`:

   ```
   > kadmin.local
   > addprinc -randkey rangeradmin/<FQDN_of_Ranger_Admin>
   ```

```
> xst -k /etc/security/keytabs/rangeradmin.keytab rangeradmin/
<FQDN_of_Ranger_Admin>@<REALM>
> exit
```

2. Verify that `rangeradmin` created principal:

```
> kinit -kt  /etc/security/keytabs/rangeradmin.keytab rangeradmin/
<FQDN_of_Ranger_Admin>@<REALM>
```

After using the **kinit** command, there should not be any errors. You can use the **klist** command to verify that your kinit command was successful.

3. Use the **kdestroy** command to destroy your active Kerberos authorization tickets by overwriting and deleting the credentials cache that contains them:

```
kdestroy
```

5.5.1.3. Prepare Ranger Lookup

Follow these instructions to prepare Ranger lookup:

1. Create `rangerlookup/<FQDN_of_Ranger_Admin>@<REALM>`:

```
> kadmin.local
> addprinc -randkey rangerlookup/<FQDN_of_Ranger_Admin>
> xst -k /etc/security/keytabs/rangerlookup.keytab rangerlookup/
<FQDN_of_Ranger_Admin>@<REALM>
> exit
```

2. Verify that `rangerlookup` created principal:

```
> kinit -kt  /etc/security/keytabs/rangerlookup.keytab rangerlookup/
<FQDN_of_Ranger_Admin>@<REALM>
```

After using the **kinit** command, there should not be any errors. You can use the **klist** command to verify that your **kinit** command was successful.

3. Use the **kdestroy** command to destroy your active Kerberos authorization tickets by overwriting and deleting the credentials cache that contains them:

```
kdestroy
```

5.5.1.4. Prepare Ranger Usersync

Follow these instructions to prepare Ranger usersync:

1. Create `rangersync/<FQDN_of_Ranger_Admin>@<REALM>`:

```
> kadmin.local
> addprinc -randkey rangersync/<FQDN_of_Ranger_Usersync>
> xst -k /etc/security/keytabs/rangerusersync.keytab
> exit
```

2. Verify that `rangersync` created principal:

```
> kinit -kt  /etc/security/keytabs/rangersync.keytab rangersync/
<FQDN_of_Ranger_usersync>@<REALM>
```

After using the **kinit** command, there should not be any errors. You can use the **klist** command to verify that your kinit command was successful.

3. Use the **kdestroy** command to destroy your active Kerberos authorization tickets by overwriting and deleting the credentials cache that contains them:

```
kdestroy
```

5.5.1.5. Prepare Ranger Tagsync

Follow these instructions to prepare Ranger usersync:

1. Create `rangertagsync/<FQDN>@<REALM>`:

```
> kadmin.local
>addprinc -randkey rangertagsync/<FQDN_of_Ranger_tagsync>
> xst -k /etc/security/keytabs/rangertagsync.keytab rangertagsync/
<FQDN>@<REALM>
> exit
```

2. Verify that `rangertagsync` created principal:

```
> kinit -kt /etc/security/keytabs/rangertagsync.keytab rangertagsync/
<FQDN_of_Ranger_tagsync>@<REALM>
```

After using the **kinit** command, there should not be any errors. You can use the **klist** command to verify that your kinit command was successful.

3. Use the **kdestroy** command to destroy your active Kerberos authorization tickets by overwriting and deleting the credentials cache that contains them:

```
kdestroy
```

4. Change the keytab permission to read-only and assign it to user `ranger`.

5.5.2. Installing Ranger Services

5.5.2.1. Prerequisites

Before you install Ranger services, you must complete the following tasks:

• Install JDK7 or higher.

• Install the latest version of your database and its respective connector jar.

5.5.2.2. Install Ranger Admin

Follow these steps to install Ranger Admin:

1. Untar the ranger-<version>-admin.tar.gz file:

```
tar zxf ranger-<version>-admin.tar.gz
```

2. Change directory to `ranger-<version>-admin`.

```
cd ranger-<version>-admin
```

3. Edit the `install.properties` file.

Enter the appropriate values for each of the following properties:

Table 5.10. install.properties Property Values

Property	Value
db_root_use	
db_root_password	
db_host	
db_name	
db_user	
db_password	
policymgr_external_url	http://<FQDN_OF_Ranger_Admin_Cluster>:6080
authentication_method	UNIX, LDAP, or AD
spnego_principal	HTTP/<FQDN_OF_Ranger_Admin_Cluster>@<REALM>
spnego_keytab	<HTTP keytab path>
token_value	30
cookie_domain	<FQDN_OF_Ranger_Admin_Cluster>
cookie_path	/
admin_principal	rangeradmin/ <FQDN_OF_Ranger_Admin_Cluster>@<REALM>
admin_keytab	<rangeradmin keytab path>
lookup_principal	rangerlookup/ <FQDN_OF_Ranger_Admin_Cluster>@<REALM>
lookup_keytab	<rangerlookup_keytab_path>
hadoop_conf	/etc/hadoop/conf

> **Note**
>
> If the Kerberos server and admin are on different hosts, copy the keytab to admin host and assign permission to user `ranger`:
>
> • scp the `rangeradmin` keytab file to the respective path of another host.
>
> • **chown ranger <rangeradmin_keytab_path>**
>
> • **chmod 400 <rangeradmin_keytab_path>**

4. Run setup.

```
./setup.sh
```

5. Start the Ranger admin server.

```
./ranger-admin-services.sh start
```

5.5.2.3. Install Ranger Usersync

Follow these steps to install Ranger Usersync:

1. Untar the `ranger-<version>-usersync.tar.gz` file:

   ```
   tar zxf ranger-<version>-usersync.tar.gz
   ```

2. Change directory to `ranger-<version>-usersync`.

   ```
   cd ranger-<version>-usersync
   ```

3. Edit the `install.properties` file.

   Enter the appropriate values for each of the following properties:

   Table 5.11. install.properties Property Values

Property	Value
POLICY_MGR_URL	http://<FQDN_OF_Ranger_Admin_Cluster>:6080
usersync_principal	rangerusersync/<FQDN>@<REALM>
usersync_keytab	<rangerusersync keytab path>
hadoop_conf	/etc/hadoop/conf

   Note

   If the Kerberos server and usersync are on different hosts, copy the keytab on usersync host and assign permission to user `ranger`:

   • scp the rangerusersync keytab file to the respective path of another host

   • **chown ranger <rangerusersync_keytab_path>**

   • **chmod 400 <rangerusersync_keytab_path>**

4. Run setup.

   ```
   ./setup.sh
   ```

5. Start the Ranger usersync server.

   ```
   ./ranger-usersync-services.sh start
   ```

5.5.2.4. Install Ranger Tagsync

Follow these steps to install Ranger Tagsync:

1. Untar the `ranger-<version>-tagsync.tar.gz` file:

   ```
   tar zxf ranger-<version>-tagsync.tar.gz
   ```

2. Change directory to `ranger-<version>-tagsync`.

   ```
   cd ranger-<version>-tagsync
   ```

3. Edit the `install.properties` file.

Enter the appropriate values for each of the following properties:

Table 5.12. install.properties Property Values

Property	Value
TAGADMIN_ENDPOINT	http://<FQDN_OF_Ranger_Admin_Cluster>:6080
tagsync_principal	rangertagsync/<FQDN>@<REALM>
tagsync_keytab	<rangertagsync keytab path>
hadoop_conf	/etc/hadoop/conf
TAG_SOURCE	`,file`

Note

If the Kerberos server and tagsync are on different hosts, copy the keytab on admin host and assign permission to user `ranger`:

- scp the rangertagsync keytab file to the respective path of another host.

- **chown ranger <rangertagsync_keytab_path>**

- **chmod 400 <rangertagsync_keytab_path>**

4. Run setup.

```
./setup.sh
```

5. Start the Ranger tagsync server.

```
./ranger-tagsyn-services.sh start
```

5.5.2.5. Install Ranger KMS

Follow these steps to install Ranger KMS:

1. Untar the `ranger-<version>-SNAPSHOT-kms.tar.gz file`:

```
 tar zxf ranger-<version>-SNAPSHOT-kms.tar.gz
```

2. Change directory to `ranger-<version>-SNAPSHOT-kms`.

```
cd ranger-<version>-SNAPSHOT-kms
```

3. Edit the `install.properties` file.

Enter the appropriate values for each of the following properties:

Table 5.13. install.properties Property Values

Property	Value
KMS_MASTER_KEY_PASSWD	<Master_Key_Password>
kms_principal	rangerkms/<FQDN_of_ranger_kms host>@<REALM>
kms_keytab	<ranger_kms_keytab_path>
hadoop_conf	<hadoop_core-site.xml_path>

Property	Value
POLICY_MGR_URL	http://<FQDN_of_ranger_admin_host>:6080

Note

If the Kerberos server and tagsync are on different hosts, copy the keytab on Ranger KMS host and assign permission to user `kms`:

- scp the rangerkms keytab file to the respective path of another host.

- chown ranger <rangerkms_keytab_path>

- chmod 400 <rangerkms_keytab_path>

4. Run setup.

```
export JAVA_HOME=<JAVA_path>
./setup.sh
```

5. Perform other setup required for a kerberized cluster such as creating keytabs, and adding a proxy user. ???ADD REFERENCE HERE???

6. Start the Ranger KMS server.

```
./ranger-kms start
```

5.5.3. Manually Installing and Enabling the Ranger Plug-ins

5.5.3.1. Install and Enable Ranger HDFS Plug-in

1. Extract your build at the appropriate place.

   Copy `ranger-<version>-SNAPSHOT-hdfs-plugin.tar.gz` to NameNode_host in directory `/usr/hdp/<hdp-version>/`.

2. Change directory to `/usr/hdp/<hdp-version>`.

3. Untar `ranger-<version>-SNAPSHOT-SNAPSHOT-hdfs-plugin.tar.gz`.

4. Change directories to `ranger-<version>-SNAPSHOT-hdfs-plugin`.

5. Edit the `install.properties` file.

   Enter the appropriate values for each of the following properties:

   #### Table 5.14. install.properties Property Values

Property	Values
POLICY_MGR_URL	http://<FQDN_of_ranger_admin_host>:6080
REPOSITORY_NAME	hadoopdev

   Additionally, for the Audit info, Solr/HDFS options are available.

6. Enable the HDFS plug-in:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk.x86_64
./enable-hdfs-plugin.sh
```

7. Stop and start the namenode:

```
su hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh stop
 namenode"
su hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh start
 namenode"
```

8. Create the default repo for HDFS with proper configuration.

   In the custom repo configuration, set the component user to `hdfs` for each of the following properties:

   • policy.download.auth.users or policy.grantrevoke.auth.users

   • tag.download.auth.users

9. Use the `Audit->plugins` tab to verify that the HDFS plug-in is communicating with Ranger admin.

5.5.3.2. Install and Enable Ranger Hive Plug-in

1. Extract your build at the appropriate place.

   Copy `ranger-<version>-SNAPSHOT-hive-plugin.tar.gz` to `hiveServer2` host in directory `/usr/hdp/<hdp-version>/`.

2. Change directory to `/usr/hdp/<hdp-version>`.

3. Untar `ranger-<version>-SNAPSHOT-SNAPSHOT-hive-plugin.tar.gz`.

4. Change directories to `ranger-<version>-SNAPSHOT-hive-plugin`.

5. Edit the `install.properties` file.

   Enter the appropriate values for each of the following properties:

Table 5.15. install.properties Property Values

Property	Values
POLICY_MGR_URL	http://<FQDN_of_ranger_admin_host>:6080
REPOSITORY_NAME	hivedev

Additionally, for the Audit info, Solr/HDFS options are available.

6. Enable the Hive plug-in:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk.x86_64
```

```
./enable-hive-plugin.sh
```

7. Stop and start hiveserver2:

```
ps -aux | grep hive | grep -i hiveserver2 | awk '{print $1,$2}' | grep hive
 | awk '{print $2}' |
     xargs kill >/dev/null 2>&1
su hive -c "nohup /usr/hdp/current/hive-server2/bin/hiveserver2 -hiveconf
 hive.metastore.uris=""
     -hiveconf hive.log.dir=/var/log/hive -hiveconf hive.log.file=
hiveserver2.log >/var/log/hive/hiveserver2.out 2>
     /var/log/hive/hiveserver2err.log &"
```

8. Create the default repo for Hive with proper configuration.

   In the custom repo configuration, set the component user to `hive` for each of the following properties:

   - policy.grantrevoke.auth.users

   - tag.download.auth.users

9. Use the `Audit->plugins` tab to verify that the Hive plug-in is communicating with Ranger admin.

5.5.3.3. Install and Enable Ranger HBase Plug-in

1. Extract your build at the appropriate place.

   Copy `ranger-<version>-SNAPSHOT-hbase-plugin.tar.gz` to `Active_Hbasemaster` host in directory `/usr/hdp/<hdp-version>/`.

2. Change directory to `/usr/hdp/<hdp-version>`.

3. Untar `ranger-<version>-SNAPSHOT-SNAPSHOT-hbase-plugin.tar.gz`.

4. Change directories to `ranger-<version>-SNAPSHOT-hbase-plugin`.

5. Edit the `install.properties` file.

   Enter the appropriate values for each of the following properties:

   Table 5.16. install.properties Property Values

Property	Values
POLICY_MGR_URL	http://<FQDN_of_ranger_admin_host>:6080
REPOSITORY_NAME	hbasedev

   Additionally, for the Audit info, Solr/HDFS options are available.

6. Enable the HBase plug-in:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk.x86_64
```

```
./enable-hbase-plugin.sh
```

7. Stop and start hbase:

```
su hbase -c "/usr/hdp/current/hbase-master/bin/hbase-daemon.sh stop
 regionserver; sleep 25"
su hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-daemon.sh stop
 master"
su hbase -c "/usr/hdp/current/hbase-master/bin/hbase-daemon.sh start master;
 sleep 25"
su hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-daemon.sh start
 regionserver"
```

8. Create the default repo for HBase with proper configuration.

   In the custom repo configuration, add the component user to `hbase` for each of the
   following properties:

   • policy.grantrevoke.auth.users

   • tag.download.auth.users

9. Use the `Audit->plugins` tab to verify that the HBase plug-in is communicating with
   Ranger admin.

5.5.3.4. Install and Enable Ranger YARN Plug-in

1. Extract your build at the appropriate place.

   Copy `ranger-<version>-SNAPSHOT-yarn-plugin.tar.gz` to
   `Active_Hbasemaster` host in directory `/usr/hdp/<hdp-version>/`.

2. Change directory to `/usr/hdp/<hdp-version>`.

3. Untar `ranger-<version>-SNAPSHOT-SNAPSHOT-yarn-plugin.tar.gz`.

4. Change directories to `ranger-<version>-SNAPSHOT-yarn-plugin`.

5. Edit the `install.properties` file.

   Enter the appropriate values for each of the following properties:

   ### Table 5.17. install.properties Property Values

Property	Values
POLICY_MGR_URL	http://<FQDN_of_ranger_admin_host>:6080
REPOSITORY_NAME	yarndev

   Additionally, for the Audit info, Solr/HDFS options are available.

6. Enable the YARN plug-in:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk.x86_64
./enable-yarn-plugin.sh
```

7. Stop and start the ResourceManager and the NodeManager:

First, stop and start the ResourceManager on all of your ResourceManager hosts:

```
su yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.sh
 stop resourcemanager"
su yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.sh
 start resourcemanager"
ps -ef | grep -i resourcemanager
```

Next, stop and start the NodeManager on all your NodeManager hosts:

```
su yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh
 stop nodemanager"
su yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh
 start nodemanager"
ps -ef | grep -i nodemanager
```

8. Create the default repo for YARN with proper configuration.

In the custom repo configuration, add the component user to `yarn` for each of the following properties:

• policy.grantrevoke.auth.users or policy.download.auth.users

• tag.download.auth.users

9. Use the `Audit->plugins` tab to verify that the YARN plug-in is communicating with Ranger admin.

5.5.3.5. Install and Enable Ranger Knox Plug-in

1. Extract your build at the appropriate place.

Copy `ranger-<version>-SNAPSHOT-knox-plugin.tar.gz` to `Active_Resourcemanager` host in directory `/usr/hdp/<hdp-version>/`.

2. Change directory to `/usr/hdp/<hdp-version>`.

3. Untar `ranger-<version>-SNAPSHOT-SNAPSHOT-knox-plugin.tar.gz`.

4. Change directories to `ranger-<version>-SNAPSHOT-knox-plugin`.

5. Edit the `install.properties` file.

Enter the appropriate values for each of the following properties:

Table 5.18. install.properties Property Values

Property	Values
POLICY_MGR_URL	http://<FQDN_of_ranger_admin_host>:6080
KNOX_HOME	/usr/hdp/<version>/knox/
REPOSITORY_NAME	knoxdev

Additionally, for the Audit info, Solr/HDFS options are available.

6. Enable the Knox plug-in:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk.x86_64
./enable-knox-plugin.sh
```

> **Note**
>
> In the HA environment, the Knox plug-in must be enabled on all Knox instances.

7. Stop and start the Knox gateway:

```
su knox -c "/usr/hdp/current/knox-server/bin/gateway.sh stop"
su knox -c "/usr/hdp/current/knox-server/bin/gateway.sh start"
```

8. Create the default repo for Knox with proper configuration.

   In the custom repo configuration, add the component user `knox` for each of the following properties:

   - policy.grantrevoke.auth.users or policy.download.auth.users

   - tag.download.auth.users

9. Use the `Audit->plugins` tab to verify that the Knox plug-in is communicating with Ranger admin.

10. For your test connection to be successful, follow the additional step "Trusting Self Signed Knox Certificate." (?? THIS LINK DOES NOT WORK??)

5.5.3.6. Install and Enable Ranger Storm Plug-in

1. Extract your build at the appropriate place.

   Copy `ranger-<version>-SNAPSHOT-storm-plugin.tar.gz` to `Active_Resourcemanager` host in directory `/usr/hdp/<hdp-version>/`.

2. Change directory to `/usr/hdp/<hdp-version>`.

3. Untar `ranger-<version>-SNAPSHOT-SNAPSHOT-storm-plugin.tar.gz`.

4. Change directories to `ranger-<version>-SNAPSHOT-storm-plugin`.

5. Edit the `install.properties` file.

   Enter the appropriate values for each of the following properties:

Table 5.19. install.properties Property Values

Property	Values
POLICY_MGR_URL	http://<FQDN_of_ranger_admin_host>:6080

Property	Values
REPOSITORY_NAME	stormdev

Additionally, for the Audit info, Solr/HDFS options are available.

6. Enable the Storm plug-in:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk.x86_64
./enable-storm-plugin.sh
```

7. Stop and start Storm:

```
???Need this info???
```

8. Create the default repo for Storm with proper configuration.

    In the custom repo configuration, add the component user `storm` for each of the following properties:

    • policy.grantrevoke.auth.users or policy.download.auth.users

    • tag.download.auth.users

9. Use the `Audit->plugins` tab to verify that the Storm plug-in is communicating with Ranger admin.

5.5.3.7. Install and Enable Ranger Kafka Plug-in

1. Extract your build at the appropriate place.

    Copy `ranger-<version>-SNAPSHOT-kafka-plugin.tar.gz` to `Active_Resourcemanager` host in directory `/usr/hdp/<hdp-version>/`.

2. Change directory to `/usr/hdp/<hdp-version>`.

3. Untar `ranger-<version>-SNAPSHOT-SNAPSHOT-kafka-plugin.tar.gz`.

4. Change directories to `ranger-<version>-SNAPSHOT-kafka-plugin`.

5. Edit the `install.properties` file.

    Enter the appropriate values for each of the following properties:

Table 5.20. install.properties Property Values

Property	Values
COMPONENT_INSTALL_DIR_NAME	/usr/hdp/<hdp-version>/kafka
POLICY_MGR_URL	http://<FQDN_of_ranger_admin_host>:6080
REPOSITORY_NAME	kafkadev

Additionally, for the Audit info, Solr/HDFS options are available.

6. Enable the Kafka plug-in:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk.x86_64
./enable-kafka-plugin.sh
```

Note

In the HA environment, the Knox plug-in must be enabled on all Knox instances.

7. Stop and start the Kafka gateway:

```
su kafka -c "/usr/hdp/current/kafka-broker/bin/kafka stop"
su kafka -c "/usr/hdp/current/kafka-broker/bin/kafka start"
```

8. Create the default repo for Kafka with proper configuration.

   In the custom repo configuration, add the component user `kafka` for each of the following properties:

   • policy.grantrevoke.auth.users or policy.download.auth.users

   • tag.download.auth.users

9. Use the `Audit->plugins` tab to verify that the Kafka plug-in is communicating with Ranger admin.

   Note

   If the Kafka plugin is unable to communicate with Ranger admin, check that the authorizer.class.name property in file `/usr/hdp/<hdp-version>/kafka/config/server.properties`, is set to org.apache.ranger.authorization.kafka.authorizer.RangerKafkaAuthorizer .

5.6. Verifying the Installation

To verify that installation was successful, perform the following checks:

• Check whether the Database `RANGER_ADMIN_DB_NAME` is present in the MySQL server running on `RANGER_ADMIN_DB_HOST`

• Check whether the Database `RANGER_AUDIT_DB_NAME` is present in the MySQL server running on `RANGER_AUDIT_DB_HOST`

• Check whether the "ranger-admin" service is installed in services.msc (Windows only)

• Check whether the "ranger-usersync" service is installed in services.msc (Windows only)

• If you plan to use the Ranger Administration Console with the UserSync feature, check whether both services start

• Go to the Ranger administration console host URL and make sure you can log in using the default user credentials