Apache Ranger 0.5.0 Installation

Overview

This document details the steps involved in Installing latest version of Apache Incubator Ranger independently on RHEL / Ubuntu / SUSE / Debian OS.

Installation of Ranger includes the following. Plan to install and configure in the same order.

Ranger Admin Portal - This is the UI portal and RESTful server for managing policies, users and groups. It also contains the UI interface for Audit queries and adhoc reports.

Ranger UserSync - This is a standalone server with dual purpose. It is used to synchronize the users/groups from Unix system or LDAP into RangerAdmin. This standalone process can be also used as an authentication server for RangerAdmin to use the linux user/password for login into RangerAdmin.

Ranger KMS - This service provides key management for Hadoop HDFS Encryption (TDE). It is highly scalable and provides access control and auditing. This service is **optional** and only needed if you are planning to use HDFS TDE.

Component Plugins - Ranger provides authorization and auditing for the following Apache projects. Apache Ranger version 0.5.x is **compatible with only the component versions mentioned below**. You need to install and enable the plugins for only does components you want enable Ranger Authorization and Auditing.

Component name	Version	Reference

HDFS	2.7.0	https://hadoop.apache.org/releases.html
HIVE	1.2.0	https://hive.apache.org/downloads.html
HBase	1.1.0.1	http://hbase.apache.org/
Knox	0.6.0	http://knox.apache.org/
Solr	5.2.1	http://lucene.apache.org/solr/
Storm	0.10.0-beta1	https://storm.apache.org/downloads.html
YARN	2.7.0	http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html

Dependent Services: Refer to Prerequisites section

Prerequisites

Product	Vendors	Purpose
JDK	OpenJDK 1.7 SunJDK 1.7	Run RangerAdmin and RangerKMS
RDBMS	MySQL 5.7 + Oracle - latest Postgres - latest SQL Server - latest	1. Storing policies 2. Storing Ranger Users and Groups 3. Storing Audit Logs (Optional)
Solr	Apache Solr - 5.2.1 +	1. Store Audit Logs 2. Used by RangerAdmin Portal to search AuditLogs
HDFS	Apache HDFS - 2.7 +	1. Store Audit Logs (Optional, but recommended)
Kerberos	MIT Microsoft AD	Even though not mandatory for installing and using Ranger, it is strongly recommended to enable Kerberos in your Hadoop. This ensures all requests are authenticated, which is very important for authorization and auditing. Without Kerberos, users would you be able to impersonate other users and workaround any authorization policies

Installation Instructions

Since Apache Ranger is still an incubating project, binary distribution is still not available. To install Apache Ranger, you will need to build from source. The below instructions is to build on RHEL or CentOS 6+ Linux server.

Preparing to install

You need to login as root or will need "sudo" access to install some of the build tools.

Install Maven

Required to build the project

```
cd /usr/local
# Download maven latest distribution tar from apache maven site
# https://maven.apache.org/download.cgi
tar -xvf apache--maven-<Version>--bin.tar.gz

export M2_HOME=/usr/local/apache--maven--<Version>
export M2=$M2_HOME/bin
export PATH=$M2:$PATH

#Now to test your install of Maven, enter the following command
mvn -version
```

Install git

Required to download the source code from Apache Ranger git repository

```
yum install git
```

Install gcc

This is optional and required if you are using your Linux /etc/passwd to authenticate to login into Ranger Admin. It is not required if you are using RangerAdmin local user/password or LDAP for authentication.

```
yum install gcc
```

Install MySQL

Ranger support MySQL, Oracle, MS SQL and PostGress RDBMS. Please refer to official installation from the vendor site. This section gives quick instructions about how to install MySQL. For full detail please refer to http://dev.mysql.com/doc/mysql-yum-repo-quick-guide/en.

If you using RedHat or CentOS, find out the version of your distribution

```
$ cat /etc/issue
CentOS release 6.6 (Final)
Kernel \r on an \m
```

Go to the this site http://dev.mysql.com/downloads/repo/yum and download the release package for your platform.

```
sudo rpm -Uvh platform-and-version-specific-package-name.rpm
#Example
sudo rpm -Uvh mysql-community-release-el6-5.noarch.rpm
yum repolist all | grep mysql
yum repolist enabled | grep mysql
    mysql-connectors-community MySQL Connectors Community                      14
    mysql-tools-community      MySQL Tools Community                           25
    mysql56-community          MySQL 5.6 Community Server                     180
#By default 5.6 is enabled. If you want to install 5.7, then please refer to the document to enable it
sudo yum install mysql-community-server
service mysqld start
/usr/bin/mysql_secure_installation
```

You will also need to download the MySQL JDBC. Download it from the this site http://dev.mysql.com/downloads/connector/j and unzip/untar it. You should copy the .jar to a common location where everyone can access it. E.g. /usr/share/java

Build Ranger Admin from source

Clone the ranger source code

```
mkdir ~/dev
cd ~/dev
git clone https://github.com/apache/incubator-ranger.git
cd incubator-ranger
git checkout ranger-0.5
# If you are interested in brining ranger-0.5.0 release source code to work with
# you should execute the following command to get the source associated with ranger-0.5.0 release
#
#  git checkout tags/ranger-0.5.0-rc3
#
```

Build the source

```
cd ~/dev/incubator-ranger
export MAVEN_OPTS="-Xmx512M"
export JAVA_HOME=<Java Installation Directory>
#e.g. export JAVA_HOME=/usr/lib/jvm/java
#Make sure your java version is 1.7.x
$JAVA_HOME/bin/java -version
   java version "1.7.0_79"
   OpenJDK Runtime Environment (rhel-2.5.5.3.el6_6-x86_64 u79-b14)
   OpenJDK 64-Bit Server VM (build 24.79-b02, mixed mode)
export PATH=$JAVA_HOME/bin:$PATH
mvn clean compile package assembly:assembly install
ls target/*.tar.gz
   ranger-0.5.0-admin.tar.gz ranger-0.5.0-kms.tar.gz ranger-0.5.0-storm-plugin.tar.gz ranger-0.5.0-hbase-plugin.
tar.gz ranger-0.5.0-knox-plugin.tar.gz ranger-0.5.0-usersync.tar.gz ranger-0.5.0-hdfs-plugin.tar.gz ranger-
0.5.0-migration-util.tar.gz ranger-0.5.0-yarn-plugin.tar.gz ranger-0.5.0-hive-plugin.tar.gz ranger-0.5.0-solr-
plugin.tar.gz ranger-0.5.0-kafka-plugin.tar.gz ranger-0.5.0-src.tar.gz
```

Verify all the tar files under the target directory:

Install Steps for Ranger Policy Admin on RHEL/CentOS

Lay down the build into appropriate places.

```
cd /usr/local
sudo tar zxvf ~/dev/incubator-ranger/target/ranger-0.5.0-admin.tar.gz
sudo ln -s ranger-0.5.0-admin ranger-admin
cd /usr/local/ranger--admin
```

Install and configure Solr or SolrCloud

Ranger stores audit logs in Solr. The RangerAdmin UI depends on Solr for running adhoc queries on audit logs. Please refer to the following link to Install and Configure Solr for Ranger Audits

2. Update the install.properties

2.1 Configure database properties. For MySQL, you will need to use the root password that you had picked while installing mysql.

```
db_root_user=root
db_root_password=root
db_host=localhost
```

c.The install process would create a couple of users in the database for storing administration and audit information, pick passwords for those too. With my choices here's how the relevant sections in the install.properties file looks like.

```
  # DB UserId used for the XASecure schema
  db_name=ranger
  db_user=rangeradmin
  db_password=rangeradmin

  # DB UserId for storing auditlog infromation
  audit_db_name=ranger
  audit_db_user=rangerlogger
  audit_db_password=rangerlogger
```

d. Ranger allows you to get fancier with security and authentication mode but for now let's just leave rest of the things in this file as they are.

e. Once all the required properties are updated run the setup.sh script

```
./setup.sh
```

f. The setup script will setup the db user/tables using the properties in install.properties and also lay down the binaries and setup initd scripts.

g. Once installed Ranger Admin service can be started by issuing the following command

```
ranger-admin start
```

h. The ranger-admin script supports the following options:

Start the Ranger Admin service : ranger--admin start

Stop the Ranger Admin service : ranger-admin stop

Restart the Ranger Admin service : ranger--admin restart

i.You can verify by visiting the external URL of the server using browser, for example:

`http://<Host_Address>:6080`

Configuring Ranger Admin Authentication Modes

Ranger Admin supports three Authentication methods namely : Active Directory, LDAP and Unix.

- **ACTIVE DIRECTORY**

To enable active directory authentication on Ranger Admin, you need to configure following properties in install.properties

PROPERY	VALUE	DESCRIPTION
authentication_method	ACTIVE_DIRECTORY	The authentication method to use.
xa_ldap_ad_domain	E.g. : "example.com"	
xa_ldap_ad_url	E.g. : "ldap://127.0.0.1:389"	Ldap server URL.

xa_ldap_ad_base_dn	E.g. : “DC=example,DC=com”	The Distinguished Name (DN) of the starting point for directory server searches.
xa_ldap_ad_bind_dn	E.g. : “CN=Administrator, CN=Users, DC=example,DC=com”	Full distinguished name (DN), including common name (CN), of an Active Directory user account that has privileges to search for users. This user account must have at least domain user privileges.
xa_ldap_ad_bind_password	Password for the account that can search for users.	
xa_ldap_ad_referral	Possible values are “ignore ”, “follow ” and “throw ”. Default value is “follow”.	When searching a directory, the server might return several search results, in addition to a few continuation references that show where to obtain further results. These results and references might be interleaved at the protocol level. When property is set to "follow", the AD service provider processes all the normal entries first, before following the continuation references. When this property is set to "throw", all of normal entries are returned in the enumeration first, before the ReferralException is thrown. By contrast, a "referral" error response is processed immediately when property is set to "follow" or "throw".

- **LDAP**

PROPERTY	VALUE	DESCRIPTION
authentication_method	LDAP	
xa_ldap_url	E.g. : "ldap://127.0.0.1:389"	Ldap server URL
xa_ldap_user DNpattern	E.g. : "uid={0}, ou=users, dc=example, dc=com"	User DN pattern is expanded when a user is being logged in. For example, if the user ‘ldapadmin’ attempted to log in, the LDAP Server would attempt to bind against the DN ‘uid=ldapadmin,ou=users,dc=example,dc=com’ using the password the user provided
xa_ldap_groupSearchBase	E.g. : "dc=example,dc=com"	Defines the part of the directory tree under which group searches should be performed.
xa_ldap_groupSearchFilter	E.g. : "(member=cn={0}, ou=users, dc=example, dc=com)"	The filter which is used to search for group membership. The default is uniqueMember={0}, corresponding to the groupOfUniqueNames LDAP class. In case of Ranger authentication, the substituted parameter is the full distinguished name of the user. The parameter {0} can be used if you want to filter on the login name.
xa_ldap_groupRoleAttribute	E.g. : "cn"	The attribute which contains the name of the authority defined by the group entry.
xa_ldap_base_dn	E.g. : " dc=example, dc=com"	The Distinguished Name (DN) of the starting point for directory server searches.
xa_ldap_bind_dn	E.g. : "cn=ldapadmin, ou=users, dc=example, dc=com"	Full distinguished name (DN), including common name (CN), of an LDAP user account that has privileges to search for users.

xa_ldap_bind_password	Password for the account that can search for users.	
xa_ldap_referral	Default value is “follow”. Possible values are “ignore ”, “follow ” and “throw ”.	When searching a directory, the server might return several search results, in addition to a few continuation references that show where to obtain further results. These results and references might be interleaved at the protocol level. When property is setto "follow", the AD service provider processes all the normal entries first, before following the continuation references. When this property is set to "throw", all of normal entries are returned in the enumeration first, before the ReferralException is thrown. By contrast, a "referral" error response is processed immediately when property is set to "follow" or "throw".

- **UNIX**

PROPERTY	VALUE	DESCRIPTION
authentication_method	UNIX	
remoteLoginEnabled	true	
authServiceHostName	E.g. : localhost	An Address of host where unixauth service is running
authentication_method	5151	Port number on which unixauth service is running. default is 5151

Configuring Ranger Admin HA Mode

1. Follow the ranger admin install steps above to install it on multiple hosts
2. Make sure to use the same configuration and policy DB details
3. Configure a load balancer (Software e.g. Apache httpd or hardware load balancer could be used) and note down the load balancer URL. *Configuration details are outside the scope of this document.*
4. Update the policy manager external URL in all the clients of Ranger Admin (Ranger UserSync and Ranger plugins) to point to the load balancer URL.
5. Restart all the clients (Ranger UserSync and Ranger plugins).

Installing the Ranger UserSync Process

- We’ll start by extracting out build at the appropriate place.

```
cd /usr/local
sudo tar -zxvf ~/dev/ incubator-ranger/target/ranger-0.5.0--usersync.tar.gz
sudo ln -s ranger--0.5.0--usersync ranger--usersync
sudo mkdir -p /var/log/ranger--usersync
sudo chown ranger /var/log/ranger-usersync
sudo chgrp ranger /var/log/ranger-usersync
```

- Edit the install.properties file and fill out relevant information.

```
POLICY_MGR_URL = http://localhost:6080
SYNC_SOURCE = unix
logdir = /var/log/ranger/usersync
```

- Now run the setup.sh command to install the usersync

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

After installing ranger -usersync, follow the same steps to start/stop services of usersync work.
./ranger--usersync-services.sh start
```

h. The ranger-usersync script supports the following options:

Start the Ranger Usersync service : ranger-usersync start

Stop the Ranger Usersync service : ranger-usersync stop

Restart the Ranger Usersync service : ranger-usersync restart

Configuring Ranger UserSync process to use LDAP/AD server

- To Sync LDAP or AD users following properties should be configured in install.properties file before executing the setup.sh.

PROPERTY	DESCRIPTION	COMMENTS
SYNC_LDAP_USER_OBJECT_CLASS	Object class to identify user entries.	Please customize the value to suit your deployment.
SYNC_LDAP_USER_SEARCH_FILTER	Optional additional filter constraining the users selected for syncing.	Please customize the value to suit your deployment.
SYNC_LDAP_USER_NAME_ATTRIBUTE	Attribute from user entry that would be treated as user name	Please customize the value to suit your deployment.
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 could provide multiple attribute names separated by comma.	Please customize the value to suit your deployment.
SYNC_LDAP_USERNAME_CASE_CONVERSION	UserSync - Case Conversion Flags.	Possible values: none, lower, upper
SYNC_LDAP_GROUPNAME_CASE_CONVERSION	UserSync - Case Conversion Flags.	Possible values: none, lower, upper
SYNC_GROUP_SEARCH_ENABLED	Do we want to do ldapsearch to find groups instead of relying on user entry attributes. valid values: true, false	Any value other than true would be treated as false
SYNC_GROUP_USER_MAP_SYNC_ENABLED	Do we want to do ldapsearch to find groups instead of relying on user entry attributes and sync memberships of those groups valid values: true, false. any value other than true would be treated as false.	
SYNC_GROUP_SEARCH_BASE	Search base for groups. overrides value specified in SYNC_LDAP_SEARCH_BASE, SYNC_LDAP_USER_SEARCH_BASE. If a value is not specified, takes the value of SYNC_LDAP_SEARCH_BASE if SYNC_LDAP_SEARCH_BASE is also not specified, takes the value of SYNC_LDAP_USER_SEARCH_BASE	
SYNC_GROUP_SEARCH_SCOPE	Search scope for the groups, only base, one and sub are supported values	Any value other than true would be treated as false
SYNC_GROUP_OBJECT_CLASS	Object class to identify group entries. Default value: groupofnames	Any value other than true would be treated as false
SYNC_LDAP_GROUP_SEARCH_FILTER	Optional additional filter constraining the groups selected for syncing. Default value is empty.	Please customize the value to suit your deployment.
SYNC_GROUP_NAME_ATTRIBUTE	Attribute from group entry that would be treated as group name.	Please customize the value to suit your deployment.
PSYNC_GROUP_MEMBER_ATTRIBUTE_NAME	Attribute from group entry that is list of members. Default value: member.	
SYNC_PAGED_RESULTS_SIZE	Page size for paged results control. Search results would be returned page by page with the specified number of entries per page default value: 500	

SYNC_PAGED_RESULTS_ENABLED	Do we want to use paged results control during ldapsearch for user entries. Valid values: true, false. default value: true Any value other than true would be treated as false. If the value is false, typically AD would not return more than 1000 entries.	Please customize the value to suit your deployment.
SYNC_SOURCE	Sync source, “ldap” should be used for both LDAP or AD user syncing.	
SYNC_LDAP_URL	URL of source ldap server. Must specify a value if SYNC_SOURCE is "ldap"	
SYNC_LDAP_BIND_DN	Ldap bind dn used to connect to ldap and query for users and groups. Must specify a value if SYNC_SOURCE is ldap.	
SYNC_LDAP_BIND_PASSWORD	Ldap bind password for the bind dn specified above. Please ensure read access to this file is limited to root, to protect the password. Must specify a value if SYNC_SOURCE is ldap unless anonymous search is allowed by the directory on users and group.	
SYNC_LDAP_SEARCH_BASE	Search base for users and groups	
SYNC_LDAP_USER_SEARCH_BASE	Search base for users, overrides value specified in SYNC_LDAP_SEARCH_BASE. Must specify a value if SYNC_SOURCE is ldap and SYNC_LDAP_SEARCH_BASE is empty.	
SYNC_LDAP_USER_SEARCH_SCOPE	Search scope for the users, supported values : base, one and sub.	

Table: LDAP/AD Properties with sample values

PROPERTIES NAME	SAMPLE VALUES FOR LDAP USER-SYNC	SAMPLE VALUES FOR AD USER-SYNC
SYNC_LDAP_URL	ldap://127.0.0.1:389	ldap://127.0.0.1:389
SYNC_LDAP_BIND_DN	cn=ldapadmin,ou=users,dc=example,dc=com	cn=adadmin,cn=Users,dc=example,dc=com
SYNC_LDAP_BIND_PASSWORD	secret	secret
SYNC_LDAP_SEARCH_BASE	dc=example,dc=com	dc=example,dc=com
SYNC_LDAP_USER_SEARCH_BASE	ou=users,dc=example,dc=com	dc=example,dc=com
SYNC_LDAP_USER_SEARCH_SCOPE	sub	sub
SYNC_LDAP_USER_OBJECT_CLASS	person	person
SYNC_LDAP_USER_SEARCH_FILTER		(objectcategory=person)
SYNC_LDAP_USER_NAME_ATTRIBUTE	uid or cn	sAMAccountName
SYNC_LDAP_USER_GROUP_NAME_ATTRIBUTE	memberof,ismemberof	memberof,ismemberof
SYNC_LDAP_USERNAME_CASE_CONVERSION	lower	lower
SYNC_LDAP_GROUPNAME_CASE_CONVERSION	lower	lower
SYNC_GROUP_SEARCH_ENABLED	false	false
SYNC_GROUP_USER_MAP_SYNC_ENABLED	false	false
SYNC_GROUP_SEARCH_BASE	ou=groups,dc=example,dc=com	dc=example,dc=com
SYNC_GROUP_SEARCH_SCOPE	sub	sub
SYNC_GROUP_OBJECT_CLASS	groupofnames	groupofnames
SYNC_LDAP_GROUP_SEARCH_FILTER		
SYNC_GROUP_NAME_ATTRIBUTE	cn	cn
SYNC_GROUP_MEMBER_ATTRIBUTE_NAME	member	member

SYNC_PAGED_RESULTS_ENABLED	true	true
SYNC_PAGED_RESULTS_SIZE	500	500

Installing Apache Hadoop

- Now let's download and install Apache Hadoop by following the excellent instructions available on the hadoop site itself. Please follow the steps given in pseudo distributed mode.
- These instructions were written for version 2.7.0. So grab that tar (hadoop--2.7.0.tar.gz) and checksum file (hadoop--2.7.0.tar.gz.mds).
- If java is not there, install JDK first.

```
sudo yum install java-1.7.0--openjdk--devel
```

- Make a note of the location where you installed Apache Hadoop. Here I assume that you have installed it in

```
/usr/local/hadoop
```

- Create a user under which we could install and ultimately run the various hadoop processes. And login as that user.

```
sudo useradd --home--dir /var/hadoop --create-home ----shell /bin/bash --user--group hadoop
```

If you get the group exists error, try the following command:

```
sudo useradd ----home--dir /var/hadoop ----create--home ----shell /bin/bash hadoop --g hadoop
```

- Create a hdfs user :

```
sudo useradd hdfs
```

- Follow the instruction on the site and install Apache Hadoop

Enabling Ranger HDFS Plugins

a. We'll start by extracting our Ranger plugin build at the appropriate place (/usr/local).

```
cd /usr/local
sudo tar zxvf ~/dev/incubator--ranger/target/ranger--0.5.0--hdfs--plugin.tar.gz
sudo ln --s ranger--0.5.0--hdfs--plugin ranger--hdfs--plugin
cd ranger--hdfs--plugin
```

b. Now let's edit the install.properties file. Here are the relevant lines that you should edit

- Change the install.properties file

PROPERTY	VALUE
POLICY_MGR_URL	http://localhost:6080
REPOSITORY_NAME	hadoopdev

XAAUDIT.DB.IS_ENABLED	true
XAAUDIT.DB.FLAVOUR	MYSQL
XAAUDIT.DB.HOSTNAME	localhost
XAAUDIT.DB.DATABASE_NAME	ranger_audit
XAAUDIT.DB.USER_NAME	rangerlogger
XAAUDIT.DB.PASSWORD	rangerlogger

c. Now enable the hdfs--plugin by running the enable--hdfs--plugin.sh command (Remember to set JAVA_HOME before running the command)

Hadoop conf and hadoop lib folder are not found at expected locations as per the script because of which, Ranger hdfs plugin installation might fail. To resolve this issue create a symlink as conf dir of hadoop linking to hadoop conf dir. For e.g:

```
cd /usr/local/hadoop
ln --s /usr/local/hadoop/etc/hadoop/ /usr/local/hadoop/conf
```

- Export HADOOP_HOME and run the enable plugin script

```
echo "export HADOOP_HOME=/usr/local/hadoop" >> /etc/bashrc
cd /usr/local/ranger--hdfs--plugin
./enable--hdfs--plugin.sh
```

- One more change that we need to do is copy all the jar files from ${hadoop_home}/lib to share lib

```
cp /usr/local/hadoop/lib/*.jar /usr/local/hadoop/share/hadoop/hdfs/lib/
```

- Make sure the log dir exists and provide required permissions to it

```
  chown root:hadoop /usr/local/hadoop/logs
  chmod g+w /usr/local/hadoop/logs
```

- Provide required permission to users in OS file system and hdfs file system according to your environment and requirement.

d. Once these changes are done restart Hadoop NameNode and Secondary NameNode.

- Stop NameNode, SecondaryNameNode and DataNode daemon:

```
  su --l hdfs --c "/usr/local/hadoop/sbin/hadoop--daemon.sh stop namenode"
  su --l hdfs --c "/usr/local/hadoop/sbin/hadoop--daemon.sh stop secondarynamenode"
  su --l hdfs --c "/usr/local/hadoop/sbin/hadoop--daemon.sh stop datanode"
```

- Start NameNode, SecondaryNameNode and DataNode daemon:

```
  su --l hdfs --c "/usr/local/hadoop/sbin/hadoop--daemon.sh start namenode"
  su --l hdfs --c "/usr/local/hadoop/sbin/hadoop--daemon.sh start secondarynamenode"
  su --l hdfs --c "/usr/local/hadoop/sbin/hadoop--daemon.sh start secondarynamenode"
```

e. This will make necessary changes to Hadoop config files to enable Ranger. One can verify the communication between Ranger Admin and Ranger Plugin as below :

- By logging into the Ranger Admin Web interface and navigating to Audit > Agents.
- You can verify the plugin is communicating to Ranger admin by going to Audit --> plugins tab and checking for status code 200 under the HDFS repo.

Make sure the REPOSITORY_NAME service exists in Ranger Admin. If not, the hdfs--plugin will not be able to communicate with Ranger admin.

Installing Apache Hive(1.2.0)

- Let's download and install apache hive by following the excellent instructions available on the apache hive site itself

```
sudo tar xzvf ~/dev/apache--hive--1.2.0--bin.tar.gz --C /usr/local
cd /usr/local
sudo ln --s apache--hive--1.2.0--bin hive
useradd hive
cd hive

#Export HIVE_HOME to bashrc
echo "export HIVE_HOME=/usr/local/hive" >> /etc/bashrc
```

`HiveServer2 doesn't start unless HADOOP_VERSION is exported to bashrc`

Enabling Ranger Hive Plugin

- We'll start by extracting our build at the appropriate place.

```
cd /usr/local
sudo tar zxvf ~/dev/incubator--ranger/target/ranger--0.5.0--hive--plugin.tar.gz
sudo ln --s ranger--0.5.0--hive--plugin ranger--hive--plugin
cd ranger--hive--plugin
```

- Now let's edit the install.properties file. Here are the relevant lines that you should edit:
- Change the insall.properties file

PROPERTY	VALUE
POLICY_MGR_URL	http://localhost:6080
REPOSITORY_NAME	hivedev
XAAUDIT.DB.IS_ENABLED	true
XAAUDIT.DB.FLAVOUR=MYSQL	MYSQL
XAAUDIT.DB.HOSTNAME	localhost
XAAUDIT.DB.DATABASE_NAME	ranger_audit
XAAUDIT.DB.USER_NAME	rangerlogger
XAAUDIT.DB.PASSWORD	rangerlogger

- Now enable the hive--plugin by running the enable--hive--plugin.sh command (Remember to set JAVA_HOME)

```
cd /usr/local/ranger--hive--plugin
./enable--hive--plugin.sh
```

This will make necessary changes to Hadoop config files to enable Ranger. One can verify the communication between Ranger Admin and Ranger Plugin as below :

- By logging into the Ranger Admin Web interface and navigating to Audit > Agents.
- You can verify the plugin is communicating to Ranger admin by going to Audit --> plugins tab and checking for status code 200 under the HIVE repo.

Make sure the REPOSITORY_NAME service exists in Ranger Admin. If not, the hive-plugin will not be able to communicate with Ranger admin.

Helpful info Ranger Hive Plugin / Hive Services

If /var/log/hive directory does not exist then create one and give required rights to user hive.

```
mkdir /var/log/hive
chown --R hive:hive /var/log/hive
```

- Make sure hive user has access to the configuration files

```
chown --R hive:hadoop /usr/local/apache--hive--1.2.0--bin/conf/hiveserver2--site.xml
chown --R hive:hadoop /usr/local/apache--hive--1.2.0--bin/conf/hive--log4j.properties
chown --R hive:hadoop /usr/local/apache--hive--1.2.0--bin/conf/hive--site.xml
```

- **To start hive metastore**

```
su --l hive --c "env HADOOP_HOME=/usr/local/hadoop JAVA_HOME=<pathToJava> nohup hive ----service metastore > /var
/log/hive/hive.out 2> /var/log/hive/hive.log &”
```

- **To start HiveServer2**

```
su --l hive --c "env HADOOP_HOME=/usr/local/hadoop JAVA_HOME=<pathToJava> nohup /usr/local/hive/bin/hiveserver2 -
hiveconf hive.metastore.uris=\" \" > /var/log/hive/hiveServer2.out 2>/var/log/hive/hiveServer2.log &”
```

- **To Stop Hive service**

```
ps aux | awk '{print $1,$2}' | grep hive | awk '{print $2}' | xargs kill >/dev/null 2>&1
```

- **Using Beeline to connect to Hive Metastore**

```
/usr/local/hive/bin/beeline --u "jdbc:hive2://localhost:10000" --n hive --p hive
```

- **If hive metastore and hiveserver2 do not start then verify below given key--values according to your environment in following files.**

hiveserver2-site.xml

```
<configuration>
<property>
        <name>hive.security.authorization.enabled</name>
        <value>true</value>
</property>
<property>
        <name>hive.security.authorization.manager</name>
      <value>org.apache.ranger.authorization.hive.authorizer.RangerHiveAuthorizerFactory</value>
</property>
<property>
        <name>hive.security.authenticator.manager</name>
       <value>org.apache.hadoop.hive.ql.security.SessionStateUserAuthenticator</value>
</property>
<property>
        <name>hive.conf.restricted.list</name>
        <value>hive.security.authorization.enabled,hive.security.authorization.manager,hive.security.
authenticator.manager</value>
</property>
</configuration>
```

hive-site.xml

```
<property>
   <name>hive.exec.scratchdir</name>
   <value>/tmp/hive</value>
</property>
<property>
   <name>hive.exec.local.scratchdir</name>
   <value>/tmp/hive</value>
</property>
<property>
   <name>hive.downloaded.resources.dir</name>
   <value>/tmp/hive_resources</value>
</property>
<property>
   <name>hive.scratch.dir.permission</name>
   <value>733</value>
</property>
<property>
   <name>javax.jdo.option.ConnectionPassword</name>
   <value>hive</value>
</property>
<property>
   <name>javax.jdo.option.ConnectionUserName</name>
   <value>hive</value>
</property>
<property>
   <name>javax.jdo.option.ConnectionURL</name>
</property>
<property>
   <name>hive.hwi.listen.host</name>
   <value>localhost</value>
</property>
<property>
   <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://localhost:3306/hive?createDatabaseIfNotExist=true</value>
</property>
```

- You can verify the plugin is communicating to Ranger admin in Audit-->plugins tab

Make sure the REPOSITORY_NAME service exists in Ranger Admin. If not, the hive-plugin will not be able to communicate with Ranger admin.

Installing Apache HBase (1.1.0.1)

- Let's download and install Apache HBase by following the excellent instructions available on the Apache Hbase site itself.

```
sudo tar xzvf ~/dev/hbase--1.1.0.1--bin.tar.gz --C /usr/local
cd /usr/local
sudo ln --s hbase--1.1.0.1 hbase
useradd hbase
cd hbase

#Export HBASE_HOME to bashrc
echo "export HBASE_HOME=/usr/local/hbase" >> /etc/bashrc
```

- For HBase 0.98.5 and later, you are required to set the JAVA_HOME environment variable before starting HBase

Enabling Ranger HBase Plugins

- We'll start by extracting our build at the appropriate place.

```
cd /usr/local
sudo tar zxvf ~/dev/incubator--ranger/target/ranger--0.5.0--hbase--plugin.tar.gz
sudo ln --s ranger--0.5.0--hbase--plugin ranger--hbase--plugin
cd ranger--hbase--plugin
```

- Now let's edit the install.properties file. Here are the relevant lines that you should edit

PROPERTY	VALUE
POLICY_MGR_URL	http://localhost:6080
REPOSITORY_NAME	hbasedev
XAAUDIT.DB.IS_ENABLED	true
XAAUDIT.DB.FLAVOUR	MYSQL
XAAUDIT.DB.HOSTNAME	localhost
XAAUDIT.DB.DATABASE_NAME	ranger_audit
XAAUDIT.DB.USER_NAME	rangerlogger
XAAUDIT.DB.PASSWORD	rangerlogger

- Now enable the hbase--plugin by running the enable--hbase--plugin.sh command (Remember to set JAVA_HOME)

```
cd /usr/local/ranger--hbase--plugin
./enable--hbase--plugin.sh
```

This will make necessary changes to HBase config files to enable Ranger. Once these changes are done restart HBase Master and HBase Regionserver.

One can verify the communication between Ranger Admin and Ranger Plugin as below :

- By logging into the Ranger Admin Web interface and navigating to Audit > Agents.
- You can verify the plugin is communicating to Ranger admin by going to Audit --> plugins tab and checking for status code 200 under the HBASE repo.

Make sure the REPOSITORY_NAME service exists in Ranger Admin. If not, the hbase-plugin will not be able to communicate with Ranger admin.

Helpful info Ranger HBase Plugin / HBase Services

- To stop HBase Master and RegionServer:

```
/usr/local/hbase/bin/hbase--daemon.sh stop master
/usr/local/hbase/bin/hbase--daemon.sh stop regionserver
```

Installing Apache Knox Gateway

- Let's download and install Apache Knox from Apache Knox Mirrors.

```
sudo tar --zxvf ~/dev/knox--0.6.0.tar.gz -C /usr/local
cd /usr/local
sudo ln --s knox--0.6.0 knox
cd knox
```

- Follow the instructions available on the Apache Knox site itself to install Apache Knox Gateway

Enabling Ranger Knox Plugins

- We'll start by extracting our build at the appropriate place.

```
cd /usr/local
tar --zxvf ~/dev/incubator--ranger/target/ranger--0.5.0--knox--plugin.tar.gz
sudo ln --s ranger--0.5.0--knox--plugin ranger--knox--plugin
cd ranger--knox--plugin
```

- Now let's edit the install.properties file. Here are the relevant lines that you should edit:

PROPERY	VALUE
POLICY_MGR_URL	http://localhost:6080
REPOSITORY_NAME	knoxdev
KNOX_HOME	/usr/local/knox
XAAUDIT.DB.IS_ENABLED	true
XAAUDIT.DB.HOSTNAME	localhost
XAAUDIT.DB.DATABASE_NAME	ranger
XAAUDIT.DB.USER_NAME	rangerlogger
XAAUDIT.DB.PASSWORD	rangerlogger

- Now enable the knox--plugin by running the enable--knox--plugin.sh command (Remember to set JAVA_HOME)

```
cd /usr/local/ranger--knox--plugin
./enable--knox--plugin.sh
```

This will make necessary changes to HBase config files to enable Ranger. Once these changes are done restart Knox Gateway and optionally Knox LDAP.

One can verify the communication between Ranger Admin and Ranger Plugin as below :

- By logging into the Ranger Admin Web interface and navigating to Audit > Agents.
- You can verify the plugin is communicating to Ranger admin by going to Audit --> plugins tab and checking for status code 200 under the KNOX repo.

Make sure the REPOSITORY_NAME service exists in Ranger Admin. If not, the knox-plugin will not be able to communicate with Ranger admin.

Helpful info Ranger Knox Plugin / Knox Services

If you get permission denied error during knox start please provide required privileges to knox user. For example :

```
chown --R knox:knox /usr/local/knox/data
chown --R knox:knox /usr/local/knox/logs
chown --R knox:knox /usr/local/knox/pids
chown --R knox:hadoop /usr/local/knox/pids/*
```

Trusting Self Signed Knox Certificate

When Knox is listening on its SSL port with self signed certificate, you have to import SSL certificate of Knox into truststore used by Ranger Admin. This can be achieved by following the step:

- Log in the machine running Apache Knox
- Export knox certificate
    a. cd $GATEWAY_HOME/data/security/keystores #This is typically /usr/local/knox/data/security/keystores on Linux machine.
    b. keytool --exportcert --alias gateway--identity --keystore gateway.jks --file knox.crt
- Copy knox.crt file onto machine running Ranger Admin to a working directory, for example /usr/local/ranger--admin
- Copy the cacerts and name it cacertswithknox

```
cd /usr/local/ranger--admin
cp $JAVA_HOME/jre/lib/security/cacerts cacertswithknox
```

- Import Knox certificate into the replicated new keystore

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

- Edit /usr/local/ranger--admin/ews/ranger--admin--services.sh
    a. Add parameter **--Djavax.net.ssl.trustStore=<path to the cacertswithknox>**to the java call in the script.
- Restart Ranger Admin.

Enabling Ranger Solr Plugin

Follow these steps to configure Ranger for Solr

1. Install and configure Apache Solr in SolrCloud mode
2. Create Solr Service/Repo in Ranger Admin and add default policies
3. Install and Enable Solr Plugin by updating install.properties and calling ./enable-solr-plugin.sh
4. Add security.json with Ranger Authorizer configuration

Install and configure Apache Solr in SolrCloud mode

You can refer here for instructions.

Solr Service in Ranger Admin

1. Login to Ranger Admin
2. Give a service name. e.g. solrdev (Note this service name. You need to use this in the install.properties)
3. For Username and Password, give "ranger" and "ranger". Please note this is not used yet
4. For Solr URL, give the URL to the Solr instances. E.g. http://$HOST1:8983/solr,http://$HOST2:8983/solr
5. Test Connection might not work. Please ignore it for now
6. Click on "Add" and save the repository
7. Go to the new service and update the default (*) policy to add user "solr" to it. This should be the same user as in the JAAS file used by Solr server process. Check -Djava.security.auth.login.config for the JAAS file used by Solr. This policy is needed because Solr process within the cloud talk with each other for replication and other house keeping requests.

Install and Enable Solr Plugin

- Extract Ranger Solr plugin to a temporary location

```
cd /tmp
tar --zxvf ~/dev/incubator--ranger/target/ranger-0.5.0-solr-plugin.tar.gz
cd ranger--0.5.0--solr--plugin
```

- Edit the install.properties file. Here are the relevant lines that you should edit:
- Change the insall.properties file

PROPERTY	Sample values	Description
POLICY_MGR_URL	http://$RANGER_ADMIN_HOST:6080	URL to the RangerAdmin
REPOSITORY_NAME	solrdev	This is the service name used during creation of creation of the Solr service in Ranger
XAAUDIT.SOLR. ENABLE	true	This enables audit to Solr
XAAUDIT.SOLR.URL	http://$AUDIT_SOLR_HOST:6083/solr /ranger_audits	This is the URL to the Solr instance used for storing audits.

- set JAVA_HOME
- Now enable the solr-plugin by running the enable--solr-plugin.sh command

```
./enable--solr--plugin.sh
```

- Repeat the above process for all the Solr instances. You can reuse the same install.properties for all the servers.
- After these changes and enable-solr-plugin.sh are run on all the servers

Configuring Solr for Ranger

Solr needs to be configured to use Ranger Authorization implementation. For that, run the following command on one of the Solr host

```
$SOLR_INSTALL_HOME/server/scripts/cloud-scripts/zkcli.sh -zkhost  $ZK_HOST:2181 -cmd put /solr/security.json
'{"authentication":{"class": "org.apache.solr.security.KerberosPlugin"},"authorization":{"class": "org.apache.
ranger.authorization.solr.authorizer.RangerSolrAuthorizer"}}'
```

In addition, please do following on all the Solr hosts

```
mkdir -p /var/log/solr/audit/solr/spool
chown solr /var/log/solr/audit/solr/spool
mkdir -p /var/log/solr/audit/hdfs/spool
chown solr /var/log/solr/audit/hdfs/spool
```

- Restart all the Solr instances
- You can check the solr.log for any errors
- You can verify by logging into the Ranger Admin Web interface -> Audit > Plugins
- Make sure to create required policies for users. If users are getting denied, please check the audit logs.

Installing Apache Storm (0.10.0)

- Let's download and install apache Storm from Apache Mirrors

```
sudo tar --zxvf ~/dev/apache--storm--0.10.0--beta1.tar.gz -C /usr/local
cd /usr/local
sudo ln --s apache--storm--0.10.0--beta1 storm
cd storm
```

- Following the instructions available on the apache storm site itself (To install Apache Storm).

Enabling Ranger Storm Plugins

- We'll start by extracting our build at the appropriate place.

```
cd /usr/local
tar --zxvf ~/dev/incubator--ranger/target/ranger--0.5.0--storm--plugin.tar.gz
sudo ln --s ranger--0.5.0--storm--plugin ranger--storm--plugin
cd ranger-storm-plugin
```

- Now let's edit the install.properties file. Here are the relevant lines that you should edit:
- Change the insall.properties file

PROPERTY	VALUE
POLICY_MGR_URL	http://localhost:6080
REPOSITORY_NAME	stormdev
XAAUDIT.DB.IS_ENABLED	true
XAAUDIT.DB.HOSTNAME	localhost

XAAUDIT.DB.DATABASE_NAME	ranger
XAAUDIT.DB.USER_NAME	rangerlogger
XAAUDIT.DB.PASSWORD	rangerlogger

- Now enable the storm-plugin by running the enable--storm-plugin.sh command (Remember to set JAVA_HOME)

```
cd /usr/local/ranger--storm--plugin
./enable--storm--plugin.sh
```

- Once these changes are done Restart Storm
- This should start the association of ranger-storm-plugin with storm
- You can verify by logging into the Ranger Admin Web interface -> Audit > Agents
- You can verify the plugin is communicating to Ranger admin in Audit >plugins tab

If stormdev service is not created in Ranger Admin then storm-plugin will not able to communicate with Ranger Admin.

Installing Apache Yarn

You can run a MapReduce job on YARN in a pseudo-distributed mode by setting a few parameters and running ResourceManager daemon and NodeManager daemon in addition

The following instructions assume that hadoop installations steps mentioned in Installing Apache Hadoop are already executed.

Enabling Ranger Yarn Plugin

- We'll start by extracting our build at the appropriate place (/usr/local).

```
cd /usr/local
sudo tar zxvf ~/dev/incubator--ranger/target/ranger--0.5.0--yarn--plugin.tar.gz
sudo ln --s ranger--0.5.0--yarn--plugin ranger--yarn--plugin
cd ranger--yarn--plugin
```

- Now let's edit the install.properties file. Here are the relevant lines that you should edit:
- Change the install.properties file

PROPERTY	VALUE
POLICY_MGR_URL	http://localhost:6080
REPOSITORY_NAME	yarndev
XAAUDIT.DB.IS_ENABLED	true
XAAUDIT.DB.FLAVOUR	MYSQL
XAAUDIT.DB.HOSTNAME	localhost
XAAUDIT.DB.DATABASE_NAME	ranger_audit
XAAUDIT.DB.USER_NAME	rangerlogger
XAAUDIT.DB.PASSWORD	rangerlogger

- Now enable the yarn--plugin by running the enable--yarn--plugin.sh command.

```
cd /usr/local/ranger--yarn--plugin
./ enable--yarn--plugin.sh
```

- One more change that we need to do is copy all the jar files from ${hadoop_home}/lib

```
cp /usr/local/ranger--yarn--plugin/lib/*.jar /usr/local/hadoop/share/hadoop/yarn/lib/
```

- if you get permission denied error during yarn start please provide required privileges to yarn user in local and hdfs file system. for example :

```
mkdir /var/log/yarn
chown --R yarn:yarn /var/log/yarn
```

- Once these changes are done Start ResourceManager daemon and NodeManager daemon.
- Start the ResourceManager on ResourceManager hosts.

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

- Start the NodeManager on NodeManager hosts.

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

- Stop the ResourceManager on ResourceManager hosts.

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

- Stop the NodeManager on NodeManager hosts.

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

This should start the association of ranger--yarn--plugin with hadoop.

1. You can verify by logging into the Ranger Admin Web interface -> Audit > Agents
2. You can verify the plugin is communicating to Ranger admin in Audit-->plugins tab

If yarndev service is not created in Ranger Admin then yarn--plugin will not able to communicate with Ranger Admin.

Installing Ranger KMS (0.5.0)

Prerequisites: (Need to done for all host on which Ranger KMS needs to be installed)

- Download “Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files” zip using below link depending upon the Java version used

    a. http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html
    b. http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html

- Unzip the above downloaded zip file to java’s security folder (Depending upon the java version used)

```
unzip UnlimitedJCEPolicyJDK7.zip into $JDK_HOME/jre/lib/security
unzip jce_policy--8.zip into $JDK_HOME/jre/lib/security
```

STEPS FOR RANGER KMS

- We’ll start by extracting our build at the appropriate place(/usr/local).

```
cd /usr/local
sudo tar --zxvf ~/dev/incubator--ranger/target/ranger--0.5.0--kms.tar.gz
sudo ln --s ranger--0.5.0--kms ranger--kms
cd ranger--kms
```

- Please note that Ranger KMS plugin is integrated with Ranger KMS and will be installed automatically when KMS is installed.
- Now let’s edit the install.properties file. Here are the relevant lines that you should edit:
- Change the install.properties file

PROPERTY	VALUE
POLICY_MGR_URL	http://localhost:6080
REPOSITORY_NAME	kmsdev
KMS_MASTER_KEY_PASSWD	enter master key password
XAAUDIT.DB.IS_ENABLED	true
XAAUDIT.DB.FLAVOUR	MYSQL
XAAUDIT.DB.HOSTNAME	localhost
XAAUDIT.DB.DATABASE_NAME	ranger_audit
XAAUDIT.DB.USER_NAME	rangerlogger
XAAUDIT.DB.PASSWORD	rangerlogger
DB_FLAVOUR	MySQL/ ORACLE/ Postgres/ SQL Server
SQL_CONNECTOR_JAR	<Location of jar file>
db_root_user	root
db_root_password	<password>
db_host	localhost
db_name	rangerkms
db_user	rangerkms
db_password	<password>

- Edit “hdfs-site.xml”(Need to give provider else it will not support hadoop commands)Replace localhost with <internal host name>

    a. Go to path cd /usr/local/hadoop/conf/
    b. vim hdfs-site.xml
    c. For property “dfs.encryption.key.provider.uri” ,enter the value “kms://http@<internal host name>:9292/kms”
    d. Save and quit

- Edit “core-site.xml”(Need to give provider else it will not support hadoop commands)
- Replace localhost with <internal host name>

    a. Go to path cd /usr/local/hadoop/conf/
    b. vim core-site.xml
    c. For property “hadoop.security.key.provider.path” ,enter the value “kms://http@<internal host name>:9292/kms”
    d. Save and quit

- Once these changes are done Restart hadoop.
- Stop NameNode, SecondaryNameNode and DataNode daemon:

```
su --l hdfs --c "/usr/local/hadoop/sbin/hadoop--daemon.sh stop namenode"
su --l hdfs --c "/usr/local/hadoop/sbin/hadoop--daemon.sh start namenode"
```

- Run setup

```
./setup.sh
```

- Start the kms server

```
ranger--kms start
```

- You can verify the plugin is communicating to Ranger admin in Audit-->plugins tab.

Note

If kmsdev service is not created in Ranger Admin then kms--plugin will not able to connect to Ranger admin.

- To Create the Kms service

PROPERTY	VALUE
REPOSITORY_NAME	Name specified in installed.properties (e.g kmsdev)
KMS URL	kms://http@<internal host name>:9292/kms
Username	<username> (for e.g keyadmin)
Password	<password>

- Check Test connection

Enabling Audit Logging To HDFS

To enable Audit to HDFS for a plugin do the below

1. Depending upon Keberos or non-kerberos mode, one of the scripts from here need to be run: https://github.com/apache/incubator-ranger/tree/master/security-admin/contrib/audit_hdfs_folders
2. Set XAAUDIT.HDFS.ENABLE = true for respective component plugin in the install.properties file which may be found in /usr/local/ranger-<component>-plugin/ directory.
3. Configure NameNode host in the XAAUDIT.HDFS.HDFS_DIR.
4. Create a policy in HDFS service from Ranger Admin for individual component users (hive/hbase/knox/storm/yarn/kafka/kms) to give READ+ WRITE permission for the particular audit folder. i.e for enabling Hive component to log Audits to HDFS , we need to create a policy for hiveuser with READ+ WRITE permissions to respective audit directory
5. Audit to HDFS caches logs in local directory, which can be specified in XAAUDIT.HDFS.LOCAL_BUFFER_DIRECTORY (this can be like '/var/log/<component>/**), which is the path where audit is stored temporarily, likewise for archived logs we need to update XAAUDIT.HDFS. LOCAL_ARCHIVE_DIRECTORY value (this can be like '/var/log/<component>/**), before enabling the plugin for the component.

HDFS audit logging is for archive purposes. For seeing audit report in the Ranger Admin UI, recommended option is Solr.

Enabling Audit Logging To SOLR

- To enable Audit report from Solr, do the below change in Ranger admin

1. Set following properties in install.properties of ranger service to work audit to solr in Ranger

PROPERTIES	VALUE
audit_store	solr
audit_solr_urls	http://solr_host:6083/solr/ranger_audits
audit_solr_user	ranger_solr
audit_solr_password	NONE

2. Restart Ranger

- To enable Audit to Solr for a plugin do the below

Set following properties in install.properties of plugin to start logging audit to Solr : for eg Hbase

PROPERTY	VALUE
XAAUDIT.SOLR.IS_ENABLED	true

XAAUDIT.SOLR.ENABLE	true
XAAUDIT.SOLR.URL	http://solr_host:6083/solr/ranger_audits
XAAUDIT.SOLR.USER	ranger_solr
XAAUDIT.SOLR.PASSWORD	NONE
XAAUDIT.SOLR.FILE_SPOOL_DIR	var/log/hadoop/hdfs/audit/solr/spool

1. Enable ranger plugin for Hbase
2. Restart Hbase component.

Configuring Kerberos Authentication

Installing and Configuring the KDC

- To use Kerberos with Hadoop you can either use an existing KDC or install a new one just for Hadoop's use. The following gives a very high level description of the installation process.
- To install a new version of the server.

```
yum install krb5-server krb5-libs krb5-auth-dialog krb5-workstation
```

- Increase the entropy of the linux kernel. This can be done by installing rng-tools on CentOS:

```
yum install rng-tools -y
```

- Edit /etc/sysconfig/rngd to add EXTRAOPTIONS="-r /dev/urandom"

```
sed 's/""/"-r\/dev\/urandom"/' -i /etc/sysconfig/rngd
```

- Start rng daemon:

```
/etc/init.d/rngd start
```

- When the server is installed use a text editor to edit the configuration file, located by default here:

```
vi /etc/krb5.conf
```

- Change the [realms] section of this file by replacing the default “kerberos.example.com” setting for the kdc and admin_server properties with the Fully Qualified Domain Name of the KDC server. In this example below, “kerberos.example.com” has been replaced with “my.kdc.server”.

```
[realms]
EXAMPLE.COM = {
  kdc = my.kdc.server
  admin_server =my.kdc.server
}
```

- Set up the KDC Access Control List (ACL) by adding administrators to /var/kerberos/krb5kdc/kadm5.acl. For example, the following command grants full access to the database for users with the admin extension: */admin@EXAMPLE.COM

Creating the Kerberos Database

- Use the utility kdb5_util to create the Kerberos database. Create Kerberos database on Kerberos server host using kdb5_util utility:

```
kdb5_util create -s
```

- **Start the KDC.**

```
/etc/rc.d/init.d/krb5kdc restart
/etc/rc.d/init.d/kadmin restart
```

Installing and Configuring the Kerberos Clients

- To install the Kerberos clients, on every server in the cluster:

```
yum install krb5-workstation
```

- **Creating Service Principals and Keytab Files**

Each service and sub-service in Hadoop must have its own principal. A principal name in a given realm consists of a primary name and an instance name, which in this case is the FQDN of the host that runs that service. As services do not login with a password to acquire their tickets, their principal's authentication credentials 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 you must create the principal, using mandatory naming conventions.Then you must create the keytab file with that principal's information and copy the file to the keytab directory on the appropriate service host.

- Open the kadmin.local utility on the KDC machine:

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

- Create the service principals

```
$kadmin.local
addprinc -randkey $primary_name/$fully.qualified.domain.name@EXAMPLE.COM
```

The -randkey option is used to generate the password.Note that in the example each service principal's primary name has appended to it the instance name, the FQDN of the host on which it runs. This provides a unique principal name for services that run on multiple hosts, like DataNodes and NodeManagers. The addition of the host name serves to distinguish, for example, a request from DataNode A from a request from DataNode B. This is important for two reasons:If the Kerberos credentials for one DataNode are compromised, it does not automatically lead to all DataNodes being compromisedIf multiple DataNodes have exactly the same principal and are simultaneously connecting to the NameNode, and if the Kerberos authenticator being sent happens to have same timestamps, then the authentication would be rejected as a replay request.

- Once the principals are created in the database, you can extract the related keytab files for transfer to the appropriate host:

```
$kadmin.local
xst -norandkey -k $keytab_file_name $primary_name/fully.qualified.domain.name@EXAMPLE.COM
```

- When the keytab files have been created, on each host create a directory for them and set appropriate permissions.

```
mkdir -p /etc/security/keytabs/
chown root:hadoop /etc/security/keytabs
chmod 750 /etc/security/keytabs
```

Configure Storm to Work with a Secured Zookeeper Cluster

- open /etc/krb5.conf file in text editor and validate the following content.

```
vi /etc/krb5.conf
[logging]
default =FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
admin_server = FILE:/var/log/kadmind.log

[libdefaults]
default_realm =EXAMPLE.COM
dns_lookup_realm =false
dns_lookup_kdc =false
ticket_lifetime =24h
renew_lifetime = 7d
forwardable = true
udp_preference_limit = 1
[realms]
EXAMPLE.COM = {
kdc =mylocalhost:88
admin_server=mylocalhost:749
default_domain =hadoop.com
}
[domain_realm]
.example.com =EXAMPLE.COM
example.com =EXAMPLE.COM
.openstacklocal =EXAMPLE.COM
[appdefaults]
pam = {
debug = false
ticket_lifetime = 36000
renew_lifetime = 36000
forwardable =true
krb4_convert= false
}
```

- Open /var/kerberos/krb5kdc/kdc.conf file in text editor and validate the following content

```
vi /var/kerberos/krb5kdc/kdc.conf
[kdcdefaults]
kdc_ports = 88
kdc_tcp_ports = 88
v4_mode = nopreauth

[realms]
EXAMPLE.COM = {
acl_file =/var/kerberos/krb5kdc/kadm5.acl
dict_file =/usr/share/dict/words
admin_keytab =/var/kerberos/krb5kdc/kadm5.keytab
supported_enctypes =des3-hmac-sha1:normal arcfour-hmac:normal des-hmac-sha1:normaldes-cbc-md5:normal des-cbc-
crc:normal des-cbc-crc:v4 des-cbc-crc:afs3
}
```

- `cd /usr/local`
- Download the zookeeper from apache mirror site http://www.apache.org/dyn/closer.cgi/zookeeper/ and untar the downloaded gz file in /usr/local
- `tar -xvf zookeeper.tar.gz`
- `useradd hadoop`
- `useradd zookeeper`
- Open zookeeper/conf/zoo.cfg file in text editor and validate the following content.

```
tickTime=4000
initLimit=20
syncLimit=10
dataDir=/usr/local/zookeeper/data
clientPort=2181
server.1=mylocalhost:2888:3888
authProvider.1=org.apache.zookeeper.server.auth.SASLAuthenticationProvider
jaasLoginRenew=3600000
kerberos.removeHostFromPrincipal=true
kerberos.removeRealmFromPrincipal=true
```

- Open zookeeper/conf/zookeeper-env.sh file in text editor and validate the following content.

```
export ZOOKEEPER_USER=zookeeper
export HADOOP_GROUP=hadoop
export ZOOKEEPER_DATA_DIR=/usr/local/zookeeper/data
export ZOOKEEPER_LOG_DIR=/usr/local/zookeeper/logs
export ZOOPIDFILE=/usr/local/zookeeper/conf/zookeeper_server.pid
export ZOOKEEPER_CONF_DIR=/usr/local/zookeeper/conf
export CLIENT_JVMFLAGS="-Djava.security.auth.login.config=/usr/local/zookeeper/conf/zookeeper_client_jaas.conf"
exportSERVER_JVMFLAGS="-Xmx1024m -Djava.security.auth.login.config=/usr/local/zookeeper/conf/zookeeper_jaas.
conf"
```

- Provide permission to files and directories.

    a. mkdir -p $ZOOKEEPER_DATA_DIR
    b. chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_DATA_DIR
    c. chmod -R 755 $ZOOKEEPER_DATA_DIR
    d. mkdir -p $ZOOKEEPER_LOG_DIR
    e. chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_LOG_DIR
    f. chmod -R 755 $ZOOKEEPER_LOG_DIR;
    g. mkdir -p $ZOOPIDFILE
    h. chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOPIDFILE
    i. chmod -R 755 $ZOOPIDFILE;
    j. chmod a+x $ZOOKEEPER_CONF_DIR/;
    k. chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_CONF_DIR
    l. chmod -R 755 $ZOOKEEPER_CONF_DIR/../

- Add principal for zookeeper service.

```
kadmin.local
addprinc -randkey zookeeper/mylocalhost@EXAMPLE.COM
xst -k -norandkey /etc/security/keytabs/zookeeper.service.keytab zookeeper/mylocalhost
```

- Add principal for HTTP service.

```
addprinc -randkey HTTP/mylocalhost@EXAMPLE.COM
xst -norandkey -k /etc/security/keytabs/spnego.service.keytab HTTP/mylocalhost
```

- Restart the zookeeper service.

```
su - zookeeper -c  "/usr/local/zookeeper/bin/zkServer.sh start"
```

Configure Kerberos Authentication for Storm

- Open /usr/local/storm/conf/storm.yaml file in text editor and validate the following content.

```
storm.zookeeper.servers:-"mylocalhost"
nimbus.host:"mylocalhost"
drpc.servers:-"mylocalhost"
storm.local.dir:"/usr/local/storm/local"
storm.log.dir:"/usr/local/storm/logs"
logviewer.port:8081
storm.messaging.transport:"backtype.storm.messaging.netty.Context"
storm.messaging.netty.buffer_size:16384
storm.messaging.netty.max_retries:10
storm.messaging.netty.min_wait_ms:1000
storm.messaging.netty.max_wait_ms:5000
nimbus.childopts:"-Djava.security.auth.login.config=/usr/local/storm/conf/storm_jaas.conf"
ui.childopts:"-Djava.security.auth.login.config=/usr/local/storm/conf/storm_jaas.conf"
supervisor.childopts:"-Djava.security.auth.login.config=/usr/local/storm/conf/storm_jaas.conf"
storm.thrift.transport:"backtype.storm.security.auth.kerberos.KerberosSaslTransportPlugin"
java.security.auth.login.config:"/usr/local/storm/conf/storm_jaas.conf"
storm.zookeeper.superACL:"sasl:stormclient@EXAMPLE.COM"
nimbus.authorizer:"backtype.storm.security.auth.authorizer.SimpleACLAuthorizer"
storm.principal.tolocal:"backtype.storm.security.auth.KerberosPrincipalToLocal"
nimbus.admins:-"stormclient"
nimbus.supervisor.users:- "stormclient"
ui.filter:"org.apache.hadoop.security.authentication.server.AuthenticationFilter"
ui.filter.params:
"type":"kerberos"
"kerberos.principal":"HTTP/mylocalhost@EXAMPLE.COM"
"kerberos.keytab":"/etc/security/keytabs/spnego.service.keytab"
"kerberos.name.rules":"DEFAULT"
```

- Open /usr/local/storm/conf/storm_jaas.conf file in text editor and validate the following content.

```
StormServer
{
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/etc/security/keytabs/nimbus.service.keytab"
storeKey=true
useTicketCache=false
principal="nimbus/mylocalhost@EXAMPLE.COM";
};
StormClient
{
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/tmp/stormclient.headless.keytab"
storeKey=true
useTicketCache=false
serviceName="nimbus"
debug=true
principal="stormclient@EXAMPLE.COM";
};
Client
{
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/tmp/stormclient.headless.keytab"
storeKey=true
useTicketCache=false
serviceName="zookeeper"
debug=true
principal="stormclient@EXAMPLE.COM";
};
```

- Open /usr/local/storm/conf/client_jaas.conf file in text editor and validate the following content.

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

- Add storm headless principal.

```
addprinc storm
xst  -k /tmp/storm.headless.keytab storm
```

- Add principal for nimbus service.

```
addprinc -randkey nimbus/mylocalhost@EXAMPLE.COM
xst  -norandkey -k /etc/security/keytabs/nimbus.service.keytab nimbus/mylocalhost
```

- Add storm client headless principal.

```
addprinc stormclient
xst  -k /tmp/stormclient.headless.keytab stormclient
```

- Provide permission to logs, keytab files.

    a. mkdir -p /usr/local/storm/logs
    b. chown -R storm:storm /usr/local/storm/logs
    c. chmod +r /tmp/*.keytab
    d. chmod +x /tmp/*.keytab
    e. chmod +r //etc/security/keytabs/*.keytab
    f. chmod +x //etc/security/keytabs/*.keytab

- To initiate a session use kinit.

```
su -l storm -c "kinit -k -t /tmp/storm.headless.keytab stormclient@EXAMPLE.COM"
su -l storm -c “klist”
```

- Now enable storm plugin.
- Create stormdev repo with stormtestuser.
- Check connection.
- Check plugin communication with ranger.
- If required add user in os/ranger admin.

Installing Ranger KMS(0.5.0) on Kerberos Environment

Prerequisites Required

- Check on secure instance

    a. Checking HTTP Principle
        i. kadmin.local
        ii. list_principals
        iii. exit
    b. search for spnego.service.keytab
        i. locate spengo keytab
        ii. kadmin.local
        iii. list_principals
        iv. exit
        - The list shown should contain HTTP/<internal host name>@Realm

c. Try to using “HTTP/<internal host name>@Realm” with “spnego.service.keytab”
    i. locate spengo keytab
    ii. kinit -k -t <path>/spengo.service.keytab/<internname host name>@EXAMPLE.COM
    iii. klist
    iv. kdestroy

- Create a principle to use with Ranger Repo for logging in KMS
- Create principal

```
kadmin.local
addprinc <principal name>
Enter the password
exit
```

- Check the principal by doing kinit

```
kinit <principal name>
Enter password
klist
kdestroy
```

- Edit “hdfs--site.xml”

```
Replace localhost with <internal host name>
Go to path cd/usr/hdp/<version>/hadoop/conf/
vim hdfs-site.xml
For property “dfs.encryption.key.provider.uri” ,enter the value “kms://http@<internal host name>:9292/kms”
save and quite
```

- Edit “core--site.xml”

```
Replace        localhost with <internal host name>
Go to path cd/usr/hdp/<version>/hadoop/conf/
vim core-site.xml
```

- Restart Namenode

```
su --l hdfs --c "/usr/hdp/<version>/hadoop/sbin/hadoop--daemon.sh stop namenode"
su --l hdfs --c "/usr/hdp/<version>/hadoop/sbin/hadoop--daemon.sh start namenode"
```

- Restart krb5kdc

```
service krb5kdc restart
```

- We’ll extract our build of ranger KMS at the appropriate place (/usr/local).

```
cd/usr/local
sudo tar -zxvf ~/dev/incubator-ranger/target/ranger-0.5.0-kms.tar.gz
sudo ln -s ranger-0.5.0-kms ranger-kms
cd ranger-kms
```

Note that ranger KMS plugin is integrated with ranger KMS and will be installed automatically when KMS is installed

- Setting required for secure instance

    a. Edit following properties under ews/webapp/WEB-INF/classes/conf.dist/kms--site.xml:

```
<property>
          <name>hadoop.kms.authentication.type</name>
          <value>kerberos</value>
</property>

<property>
        <name>hadoop.kms.authentication.kerberos.keytab</name>
        <value>/etc/security/keytabs/spnego.service.keytab</value>
</property>

<property>
        <name>hadoop.kms.authentication.kerberos.principal</name>
        <value>*</value>
/property>
```

- Add following property in ews/webapp/WEB-INF/classes/conf.dist/kms--site.xml : (Replace “testkms1” with appropriate user who will be used for credential authentication):

```
<property>
        <name>hadoop.kms.proxyuser.testkms1.groups</name>
        <value>*</value>
</property>

<property>
        <name>hadoop.kms.proxyuser.testkms1.hosts</name>
        <value>*</value>
</property>

<property>
        <name>hadoop.kms.proxyuser.testkms1.users</name>
        <value>*</value>
</property>
```

- Create link to /etc/hadoop/conf/core-site.xml under /etc/ranger/kms/conf
    1. (ln -s /etc/hadoop/conf/core-site.xml /etc/ranger/kms/conf/core-site.xml)
- Install Ranger-kms with appropriate property values
    a. Go to ranger-kms folder and edit install.properties (Enter appropriat values for the below given properties)
        i. db_root user=
        ii. db_root password
        iii. db_host
        iv. db_name
        v. db_user
        vi. db_password
        vii. KMS_MASTER_KEYPASSWD
        viii. POLICY_MGR_URL
        ix. REPOSITORY_NAME
        x. XAAUDIT.DB.IS_ENABLED
        xi. XAAUDIT.DB.FLAVOUR
        xii. XAAUDIT.DB.HOSTNAME
        xiii. XAAUDIT.DB.DATABASE_NAME
        xiv. XAAUDIT.DB.USER_NAME
        xv. XAAUDIT.DB.PASSWORD
- Run setup

```
./setup.sh
```

- start the KMS server

```
rangee-kms start
```

Ranger UI setup

Check the user present in ranger which will be used for credential validation (for e.g (“testkms1”) if not then create that using “admin” login

- Create KMS service

    a. REPOSITORY_NAME: name specified in installed.properties (e.g kmsdev)
    b. KMS URL: kms://http@<internal host name>:9292/kms
    c. Username:Principle that will be used for kms (e.g.testkms1@EXAMPLE.COM)
    d. Password:Password for principle(e.g.testkms1 password)
    e. Check test connection

    - Please check the logged in user (for e.g “keyadmin”) has the permission for KMS operation You can navigate to KMS Tab and do all the operation related to the KMS.