Hortonworks Data Platform

Installing HDP Manually

(Jan 14, 2013)

docs.hortonworks.com

Hortonworks Data Platform : Installing HDP Manually

The Hortonworks Data Platform, powered by Apache Hadoop, is a massively scalable and 100% open source platform for storing, processing and analyzing large volumes of data. It is designed to deal with data from many sources and formats in a very quick, easy and cost-effective manner. The Hortonworks Data Platform consists of the essential set of Apache Hadoop projects including MapReduce, Hadoop Distributed File System (HDFS), HCatalog, Pig, Hive, HBase, Zookeeper and Ambari. Hortonworks is the major contributor of code and patches to many of these projects. These projects have been integrated and tested as part of the Hortonworks Data Platform release process and installation and configuration tools have also been included.

Unlike other providers of platforms built using Apache Hadoop, Hortonworks contributes 100% of our code back to the Apache Software Foundation. The Hortonworks Data Platform is Apache-licensed and completely open source. We sell only expert technical support, training and partner-enablement services. All of our technology is, and will remain free and open source.

Please visit the Hortonworks Data Platform page for more information on Hortonworks technology. For more information on Hortonworks services, please visit either the Support or Training page. Feel free to Contact Us directly to discuss your specific needs.

Table of Contents

List of Tables

1. Getting Ready to Install

This section describes the information and materials you need to get ready to install the Hortonworks Data Platform (HDP) manually. In general, the following instructions cover non-secure installations. For the additional information and steps needed to add security (Kerberos) to your installation, please see Setting Up Security for Manual Installs

1.1. Understand the Basics

The Hortonworks Data Platform consists of three layers.

- **Core Hadoop**:The basic components of Apache Hadoop.
  - **Hadoop Distributed File System (HDFS)**: A special purpose file system that is designed to work with the MapReduce engine. It provides high-throughput access to data in a highly distributed environment.
  - **MapReduce**: A framework for performing high volume distributed data processing using the MapReduce programming paradigm.
- **Essential Hadoop:** A set of Apache components designed to ease working with Core Hadoop.
  - **Apache Pig**: A platform for creating higher level data flow programs that can be compiled into sequences of MapReduce programs, using Pig Latin, the platform's native language.
  - **Apache Hive**: A tool for creating higher level SQL-like queries using HiveQL, the tool's native language, that can be compiled into sequences of MapReduce programs.
  - **Apache HCatalog**: A metadata abstraction layer that insulates users and scripts from how and where data is physically stored.
  - **WebHCat (Templeton)**: A component that provides a set of REST-like APIs for HCatalog and related Hadoop components.
  - **Apache HBase**: A distributed, column-oriented database that provides the ability to access and manipulate data randomly in the context of the large blocks that make up HDFS.
  - **Apache ZooKeeper**:A centralized tool for providing services to highly distributed systems. ZooKeeper is necessary for HBase installations.
- **Supporting Components**: A set of components that allow you to monitor your Hadoop installation and to connect Hadoop with your larger compute environment.
  - **Apache Oozie**:A server based workflow engine optimized for running workflows that execute Hadoop jobs.
  - **Apache Sqoop**: A component that provides a mechanism for moving data between HDFS and external structured datastores. Can be integrated with Oozie workflows.

- **Apache Flume**: A log aggregator. This component must be installed manually. See Installing and Configuring Flume NG for more information.
- **Apache Mahout**: A scalable machine learning library that implements several different approaches to machine learning. This component must be installed manually on an appropriate host, using yum for RHEL or CentOS or zypper for SLES. No configuration is needed.
- **Ganglia**: An Open Source tool for monitoring high-performance computing systems.
- **Nagios**: An Open Source tool for monitoring systems, services, and networks.

You must always install Core Hadoop, but you can select the components from the other layers based on your needs.

For more information on the structure of the HDP, see UnderstandDowning Hadoop Ecosystem.

1.2. Meet Minimum System Requirements

To run the Hortonworks Data Platform, your system must meet minimum requirements.

- Hardware Recommendations
- Operating System Requirements
- Software Requirements
- Database Requirements
- JDK Recommendations

1.2.1. Hardware Recommendations

Although there is no single hardware requirement for installing HDP, there are some basic guidelines. You can see sample setups here: Hardware Recommendations for Apache Hadoop.

1.2.2. Operating Systems Requirements

The following operating systems are supported:

- 64-bit Red Hat Enterprise Linux (RHEL) 5 or 6
- 64-bit CentOS 5 or 6
- 64-bit SUSE Linux Enterprise Server (SLES) 11, SP1

1.2.3. Software Requirements

On each of your hosts:

- yum [for RHEL or CentOS]

- zypper [for SLES]
- rpm
- scp [for multiple node installs]
- curl
- wget
- unzip
- tar
- pdsh [for multiple node installs over many hosts]

1.2.4. Database Requirements

- To use external database for Hive or Oozie metastore, ensure that a MySQL or Oracle database is deployed and available.

  (By default, Oozie uses Derby database for its metastore.)

  - For instructions on deploying and/or configuring MySQL database instance, see here [3].
  - For instructions on configuring an existing Oracle database instance, see here [4].

  

  Note

  To deploy a new Oracle instance, consult your database administrator.

- Ensure that your database administrator creates the following databases and users. (For instructions on creating users for MySQL, see here [4].):

  - If deploying Hive:

    1. hive_dbname: Required only if you are using MySQL database for Hive Metastore.
    2. hive_dbuser
    3. hive_dbpasswd

  - If deploying Oozie:

    1. oozie_dbname: Required only if you are using MySQL database for Oozie Metastore.
    2. oozie_dbuser
    3. oozie_dbpasswd

Instructions to setup MySQL database

1. Connect to the host machine where you plan to deploy MySQL instance and from a terminal window, type:

- For RHEL and CentOS:

```
yum install mysql-server
```

- For SLES:

```
zypper install mysql
```

2. Start the instance.

- For RHEL and CentOS:

```
/etc/init.d/mysqld start
```

- For SLES:

```
/etc/init.d/mysql start
```

3. Set the `root` user password and remove unnecessary information from `log` and STDOUT.

```
mysqladmin -u root password '$password'
```

```
mysqladmin -u root 2>&1 >/dev/null
```

4. Install the MySQL connector jar

```
yum install mysql-connector-java-5.0.8-1
```

Instructions to configure Oracle database

- Download the Oracle JDBC (OJDBC) driver from here and copy the downloaded JAR file to the following locations:

Note

The following instructions are for OJDBC driver for Oracle 11g.

  - **If installing Hive:** Copy the JAR file to `/usr/lib/hive/lib/`.
  - **If using external Oracle database for Oozie metastore:** Copy the JAR file to `/usr/lib/oozie/libtools/`.
  - **If installing Sqoop:** Copy the JAR file to `/usr/lib/sqoop/lib`.

- Ensure that the following SQL script is run against your Hive schema:

```
master-install-location/gsInstaller/confSupport/sql/oracle/hive-schema-0.10.0.oracle.sql
```

Instructions on manually creating users for MySQL:

- As `root`, use mysql (or other client tool) to create the "dbuser" and grant it adequate privileges.

  (For access to Hive metastore, create `hive_dbuser` and for access to Oozie metastore, create `oozie_dbuser`.

```
CREATE USER 'dbusername'@'%' IDENTIFIED BY 'dbuserpassword';
GRANT ALL PRIVILEGES ON *.* TO 'dbusername'@'%';
flush privileges;
```

- See if you can connect to the database as that user. You are prompted to enter the `dbuserpassword` password above.

```
mysql -u $dbusername -p
```

1.2.5. JDK Requirements

Your system must have the correct JDK installed on all the nodes of the cluster. HDP requires Oracle JDK 1.6 update 31. For more information, see Install the Java Development Kit

1.3. Configure the Remote Repository

The standard HDP install fetches the software from a remote yum repository over the Internet. To use this option, you must set up access to the remote repository and have an available Internet connection for each of your hosts.

Note

If your cluster does not have access to the Internet, or you are creating a large cluster and you want to conserve bandwidth, you can instead provide a local copy of the HDP repository that your hosts can access. For more information, see Deployment Strategies for Data Centers with Firewalls., a separate document in this set.

1. For each node in your cluster, download the repo configuration file `hdp.repo`. From a terminal window, type:

   - For RHEL and CentOS 5

   ```
   wget -nv http://public-repo-1.hortonworks.com/HDP-1.2.0/repos/centos5/hdp.repo  -O /etc/yum.repos.d/hdp.repo
   ```

   - For RHEL and CentOS 6

   ```
   wget -nv http://public-repo-1.hortonworks.com/HDP-1.2.0/repos/centos6/hdp.repo -O /etc/yum.repos.d/hdp.repo
   ```

   - For SLES

   ```
   wget -nv http://public-repo-1.hortonworks.com/HDP-1.2.0/repos/suse11/hdp.repo -O /etc/zypp/repos.d/hdp.repo
   ```

2. Confirm the HDP repository is configured by checking the repo list.

   - For RHEL/CentOS

   ```
   yum repolist
   ```

- For SLES

```
zypper repos
```

You should see something like this. Just make sure you have both HDP-1.2.0 and HDP-UTILS-1.1.0.15:

```
Loaded plugins: fastestmirror, security
Loading mirror speeds from cached hostfile
* base: mirrors.cat.pdx.edu
* extras: linux.mirrors.es.net
* updates: mirrors.usc.edu
repo id   repo name                                                status
HDP-1.2.0 Hortonworks Data Platform Version - HDP-1.2.0          64
HDP-UTILS-1.1.0.15 Hortonworks Data Platform Utils Version - HDP-UTILS-1.2.0
```

1.4. Decide on Deployment Type

While it is possible to deploy all of HDP on a single host, this is appropriate only for initial evaluation. In general you should use at least three hosts: one master host and two slaves.

1.5. Collect Information

To deploy your HDP installation, you need to collect the following information:

- The fully qualified domain name (FQDN) for each host in your system, and which component(s) you wish to set up on which host. You can use `hostname -f` to check for the FQDN if you do not know it.
- The hostname (for an existing instance), database name, username, and password for the MySQL/Oracle instance, if you want to use external database for Hive or Oozie metastore.

Note

If you are using an existing instance, the database user you create for HDP's use must be granted ALL PRIVILEGES on that instance.

1.6. Prepare the Environment

To deploy your HDP instance, you need to prepare your deploy environment:

- Enable NTP on the Cluster
- Check DNS
- Disable SELinux

1.6.1. Enable NTP on the Cluster

The clocks of all the nodes in your cluster must be able to synchronize with each other. If your system does not have access to the Internet, set up a master node as an NTP server.

1.6.2. Check DNS

All hosts in your system must be configured for DNS and Reverse DNS.

Note

If you are unable to configure DNS and Reverse DNS, you must edit the hosts file on every host in your cluster to contain each of your hosts.

1.6.3. Disable SELinux

SELinux can interfere with the installation process.

1.7. [Optional] Configure the Local Repositories

If your cluster does **not** have access to the Internet, or you are creating a large cluster and you want to conserve bandwidth, you need to provide access to the bits using an alternative method. For more information, see Deploying HDP In Production Data Centers with Firewalls in the Reference document of this set.

1.8. Install the Java Development Kit

The correct JDK must be installed on all the nodes in your cluster. To manually deploy the JDK:

1. Check the version. From a terminal window, type:

```
java -version
```

2. (Optional) Uninstall the Java package if the JDK version is less than v1.6 update 31.

```
rpm -qa | grep java
yum remove {java-1.x.0-jdk-1.x.0.0-1.45.1.11.1.el6.x86_64}
```

3. (Optional) Verify that the default Java package is uninstalled.

```
which java
```

4. Download the Oracle 64-bit JDK jdk-6u31-linux-x64.bin from the Oracle download site.

5. Change directory to the location where you downloaded the JDK and run the install.

```
mkdir /usr/jdk1.6.0_31
cd /usr/jdk1.6.0_31
chmod u+x $JDK_download_directory/jdk-6u31-linux-x64.
bin
$JDK_download_directory/jdk-6u31-linux-x64.bin
```

6. Create symbolic links (symlinks) to the JDK.

```
mkdir /usr/java
ln -s /usr/jdk1.6.0_31/jdk1.6.0_31 /usr/java/default
```

```
ln -s /usr/java/default/bin/java /usr/bin/java
```

7. Set up your environment to define JAVA_HOME to put the Java Virtual Machine and the Java compiler on your path.

```
export JAVA_HOME=/usr/java/default
export PATH=$JAVA_HOME/bin:$PATH
```

1.9. Create Service Users and Groups

In general Hadoop services should be owned by specific users and not by root or application users. The table below shows typical users for Hadoop services. Identify the users that you want for your Hadoop services and the common Hadoop group and create these accounts on your system.

Note

If you are considering installing your cluster in secure mode, either at installation or at a later time, you need to understand the relationship between OS system service users and Kerberos principals. Hadoop uses group memberships of users at various places, such as to determine group ownership for files or for access control. In order for Hadoop to be able to connect a Kerberos principal with its respective OS system service user, a mapping must be created. For more information on this process, see Setting Up Security for Manual Installs

Table 1.1. Typical Service Users and Groups

Hadoop Service	User	Group
HDFS	hdfs	hadoop
MapReduce	mapred	hadoop
Hive	hive	hadoop
Pig	pig	hadoop
HCatalog/WebHCat	hcat	hadoop
HBase	hbase	hadoop
ZooKeeper	zookeeper	hadoop
Oozie	oozie	hadoop

1.10. Download Companion Files

We have provided a set of companion files, including script files (`scripts.zip`) and configuration files (`configuration_files.zip`), that you should download and use throughout this process. Download and extract the files from here: http://public-repo-1.hortonworks.com/HDP-1.2.0/tools/hdp_manual_install_rpm_helper_files-1.2.0.21.tar.gz

1.11. Define Environment Parameters

You need to set up specific users and directories for your HDP installation.

1.11.1. Define Users and Groups

The following table describes system user account and groups. Use this table to define what you are going to use in setting up your environment. These users and groups should reflect the accounts you created in Create System Users and Groups.

Note

The `scripts.zip` file you downloaded in Download Companion Files includes a script, `usersAndGroups.sh`, for setting user and group environment parameters. We strongly suggest you edit and execute this script to fit your environment.

Table 1.2. Define Users and Groups for Systems

Parameter	Definition
HDFS_USER	User owning the HDFS services. For example, `hdfs`.
MAPRED_USER	User owning the MapReduce services. For example, `mapred`.
ZOOKEEPER_USER	User owning the ZooKeeper services. For example, `zookeeper`.
HIVE_USER	User owning the Hive services. For example, `hive`.
WEBHCAT_USER	User owning the WebHCat services. For example, `hcat`.
HBASE_USER	User owning the HBase services. For example, `hbase`.
OOZIE_USER	User owning the Oozie services. For example, `oozie`.
PIG_USER	User owning the Pig services. For example, `pig`.
HADOOP_GROUP	A common group shared by services. For example, `hadoop`.

1.11.2. Define Directories

The following table describes the directories for install, configuration, data, process IDs and logs based on the Hadoop Services you plan to install. Use this table to define what you are going to use in setting up your environment.

Note

The `scripts.zip` file you downloaded in Download Companion Files includes a script, `directories.sh`, for setting directory environment parameters. We strongly suggest you edit and execute this script to fit your environment.

Table 1.3. Define Directories for Core Hadoop

Hadoop Service	Parameter	Definition

HDFS	DFS_NAME_DIR	Space separated list of directories where NameNode should store the file system image.
		
		For example,
		
		`/grid/hadoop/hdfs/nn`
		
		`/grid1/hadoop/hdfs/nn`

Hadoop Service	Parameter	Definition
HDFS	DFS_DATA_DIR	Space separated list of directories where DataNodes should store the blocks. For example, /grid/hadoop/hdfs/dn /grid1/hadoop/hdfs/dn /grid2/hadoop/hdfs/dn
HDFS	FS_CHECKPOINT_DIR	Space separated list of directories where SecondaryNameNode should store the checkpoint image. For example, /grid/hadoop/hdfs/snn /grid1/hadoop/hdfs/snn /grid2/hadoop/hdfs/snn
HDFS	HDFS_LOG_DIR	Directory for storing the HDFS logs. This directory name is a combination of a directory and the $HDFS_USER. For example, /var/log/hadoop/hdfs where hdfs is the $HDFS_USER
HDFS	HDFS_PID_DIR	Directory for storing the HDFS process ID. This directory name is a combination of a directory and the $HDFS_USER. For example, /var/run/hadoop/hdfs where hdfs is the $HDFS_USER.
HDFS	HADOOP_CONF_DIR	Directory for storing the Hadoop configuration files. For example, /etc/hadoop/conf
MapReduce	MAPREDUCE_LOCAL_DIR	Space separated list of directories where MapReduce should store temporary data. For example, /grid/hadoop/mapred /grid1/hadoop/mapred /grid2/hadoop/mapred.
MapReduce	MAPRED_LOG_DIR	Directory for storing the HDFS logs. For example, /var/log/hadoop/mapred. This directory name is a combination of a directory and the $MAPRED_USER.

Hadoop Service	Parameter	Definition
		In the example `mapred` is the `$MAPRED_USER`
MapReduce	`MAPRED_PID_DIR`	Directory for storing the MapReduce process ID. For example, `/var/run/hadoop/mapred`. This directory name is a combination of a directory and the `$MAPRED_USER`. In the example, `mapred` is the `$MAPRED_USER`.

Table 1.4. Define Directories for Ecosystem Components

Hadoop Service	Parameter	Definition
Pig	`PIG_CONF_DIR`	Directory to store the Pig configuration files. For example, "`/etc/pig/conf`"
Oozie	`OOZIE_CONF_DIR`	Directory to store the Oozie configuration files. For example, "`/etc/oozie/conf`"
Oozie	`OOZIE_DATA`	Directory to store the Oozie data. For example, "`/var/db/oozie`"
Oozie	`OOZIE_LOG_DIR`	Directory to store the Oozie logs. For example, "`/var/log/oozie`"
Oozie	`OOZIE_PID_DIR`	Directory to store the Oozie process ID. For example, "`/var/run/oozie`"
Oozie	`OOZIE_TMP_DIR`	Directory to store the Oozie temporary files. For example, "`/var/tmp/oozie`"
Hive	`HIVE_CONF_DIR`	Directory to store the Hive configuration files. For example, "`/etc/hive/conf`"
Hive	`HIVE_LOG_DIR`	Directory to store the Hive logs. For example, "`/var/log/hive`"
Hive	`HIVE_PID_DIR`	Directory to store the Hive process ID. For example, "`/var/run/hive`"
WebHCat	`WEBHCAT_CONF_DIR`	Directory to store the WebHCat configuration files. For example, "`/etc/hcatalog/conf/webhcat`"
WebHCat	`WEBHCAT_LOG_DIR`	Directory to store the WebHCat logs. For example, "`/grid/0/var/log/webhcat/webhcat`"
WebHCat	`WEBHCAT_PID_DIR`	Directory to store the WebHCat process ID. For example, "`/var/run/webhcat`"
HBase	`HBASE_CONF_DIR`	Directory to store the HBase configuration files. For example, "`/etc/hbase/conf`"
HBase	`HBASE_LOG_DIR`	Directory to store the HBase logs. For example, "`/var/log/hbase`"
HBase	`HBASE_PID_DIR`	Directory to store the HBase process ID. For example, "`/var/run/hbase`"
ZooKeeper	`ZOOKEEPER_DATA_DIR`	Directory where ZooKeeper will store data. For example, `/grid1/hadoop/zookeeper/data`

Hadoop Service	Parameter	Definition
ZooKeeper	ZOOKEEPER_CONF_DIR	Directory to store the ZooKeeper configuration files. For example, "/etc/zookeeper/conf"
ZooKeeper	ZOOKEEPER_LOG_DIR	Directory to store the ZooKeeper logs. For example, "/var/log/zookeeper"
ZooKeeper	ZOOKEEPER_PID_DIR	Directory to store the ZooKeeper process ID. For example, "/var/run/zookeeper"
Sqoop	SQOOP_CONF_DIR	Directory to store the Sqoop configuration files. For example, "/usr/lib/sqoop/conf"

2. Installing HDFS and MapReduce

This section describes how to install the Hadoop Core components, HDFS and MapReduce.

2.1. Set Default File and Directory Permissions

Set the default file and directory permissions to 0022 (022). This is typically the default for most Linux distributions. Use the `umask` command to confirm and set as necessary. Be sure the correct `umask` is set for all terminal sessions that you use during installation.

2.2. Install the Hadoop RPMs

Execute the following command on all cluster nodes. From a terminal window, type:

[For RHEL and CentOS]

```
yum install hadoop hadoop-libhdfs hadoop-native hadoop-pipes hadoop-sbin
 openssl
```

[For SLES]

```
zypper install hadoop hadoop-libhdfs hadoop-native hadoop-pipes hadoop-sbin
 openssl
```

2.3. Install Compression Libraries

Make the following compression libraries available on all the cluster nodes.

2.3.1. Install Snappy

1. Install Snappy.
   - For RHEL and CentOS

     ```
     yum install snappy snappy-devel
     ```

   - For SLES

     ```
     zypper install snappy snappy-devel
     ```

2. Make the Snappy libraries available to Hadoop:

   ```
   ln -sf /usr/lib64/libsnappy.so /usr/lib/hadoop/lib/native/Linux-amd64-64/.
   ```

2.3.2. Install LZO

[For RHEL and CentOS]

```
yum install hadoop-lzo lzo lzo-devel hadoop-lzo-native
```

[For SLES]

```
zypper install lzo lzo-devel hadoop-lzo hadoop-lzo-native
```

2.4. Create Directories

Create directories and configure ownership + permissions on the appropriate hosts as described below.

Note

If any of these directories already exist, we recommend deleting and recreating them.

The `scripts.zip` file you downloaded in Download Companion Files includes two scripts, `usersAndGroups.sh` and `directories.sh`, for setting environment parameters. We strongly suggest you edit and execute these scripts to fit your environment.

2.4.1. Create the NameNode Directories

On the node that hosts the NameNode service, execute the following commands:

```
mkdir -p $DFS_NAME_DIR
chown -R $HDFS_USER:$HADOOP_GROUP $DFS_NAME_DIR
chmod -R 755 $DFS_NAME_DIR
```

2.4.2. Create the SecondaryNameNode Directories

On all that nodes that can potentially run the SecondaryNameNode service, execute the following commands:

```
mkdir -p $FS_CHECKPOINT_DIR
chown -R $HDFS_USER:$HADOOP_GROUP $FS_CHECKPOINT_DIR
chmod -R 755 $FS_CHECKPOINT_DIR
```

2.4.3. Create the DataNode and MapReduce Local Directories

On all DataNodes, execute the following commands:

```
mkdir -p $DFS_DATA_DIR
chown -R $HDFS_USER:$HADOOP_GROUP $DFS_DATA_DIR
chmod -R 750 $DFS_DATA_DIR
```

On the JobTracker and all Datanodes, execute the following commands:

```
mkdir -p $MAPREDUCE_LOCAL_DIR
chown -R $MAPRED_USER:$HADOOP_GROUP $MAPREDUCE_LOCAL_DIR
```

```
chmod -R 755 $MAPREDUCE_LOCAL_DIR
```

2.4.4. Create the Log and PID Directories

On all nodes, execute the following commands:

```
mkdir -p $HDFS_LOG_DIR
chown -R $HDFS_USER:$HADOOP_GROUP $HDFS_LOG_DIR
chmod -R 755 $HDFS_LOG_DIR
```

```
mkdir -p $MAPRED_LOG_DIR
chown -R $MAPRED_USER:$HADOOP_GROUP $MAPRED_LOG_DIR
chmod -R 755 $MAPRED_LOG_DIR
```

```
mkdir -p $HDFS_PID_DIR
chown -R $HDFS_USER:$HADOOP_GROUP $HDFS_PID_DIR
chmod -R 755 $HDFS_PID_DIR
```

```
mkdir -p $MAPRED_PID_DIR
chown -R $MAPRED_USER:$HADOOP_GROUP $MAPRED_PID_DIR
chmod -R 755 $MAPRED_PID_DIR
```

3. Setting Up the Hadoop Configuration

This section describes how to set up and edit the deployment configuration files for HDFS and MapReduce.

The `scripts.zip` file you downloaded in Download Companion Files includes two scripts, `usersAndGroups.sh` and `directories.sh`, for setting environment parameters. We strongly suggest you edit and execute these scripts to fit your environment.

3.1. Extract the Core Hadoop Configuration Files

From the file you downloaded in Download Companion Files extract the files in `configuration_files/core_hadoop` directory to a temporary location.

3.2. Modify the Configuration Files

In the temporary directory, locate the following files and modify the properties based on your environment. Search for TODO in the files for the properties to replace. See Define Environment Parameters for more information.

Table 3.1. core-site.xml

Property	Example	Description
fs.default.name	`hdfs://` *`$namenode.full.hostname`*`:8020`	Enter your NameNode hostname
fs.checkpoint.dir	`/grid/hadoop/hdfs/snn`	A comma separated list of paths. Use the list of directories from *`$FS_CHECKPOINT_DIR`*.

Table 3.2. hdfs-site.xml

Property	Example	Description
dfs.name.dir	`/grid/hadoop/hdfs/nn,/grid1/hadoop/hdfs/nn`	Comma separated list of paths. Use the list of directories from *`$DFS_NAME_DIR`*
dfs.data.dir	`/grid/hadoop/hdfs/dn,grid1/hadoop/hdfs/dn`	Comma separated list of paths. Use the list of directories from *`$DFS_DATA_DIR`*
dfs.http.address	*`$namenode.full.hostname`*`:50070`	Enter your NameNode hostname for http access
dfs.secondary.http.address	*`$secondary.namenode.full.hostname`*`:50090`	Enter your SecondaryNameNode hostname
dfs.https.address	*`$namenode.full.hostname`*`:50470`	Enter your NameNode hostname for https access.

Note

The value of NameNode new generation size should be 1/8 of maximum heap size (`-Xmx`). Please check this value, as the default setting may not be accurate. To change the default value, edit the `/etc/hadoop/conf/hadoop-env.sh` file and change the value of the `-XX:MaxnewSize` parameter to 1/8th the

value of the maximum heap size (-Xmx) parameter. Also ensure that the NameNode and Secondary NameNode have identical memory settings.

Table 3.3. mapred-site.xml

Property	Example	Description
mapred.job.tracker	*$jobtracker.full.hostname*:50300	Enter your JobTracker hostname
mapred.job.tracker.http.address	*$jobtracker.full.hostname*:50030	Enter your JobTracker hostname
mapred.local.dir	/grid/hadoop/mapred,/grid1/hadoop/mapred	Comma separated list of paths. Use the list of directories from $MAPREDUCE_LOCAL_DIR
mapreduce.tasktracker.group	hadoop	Enter your group. Use the value of *$HADOOP_GROUP*
mapreduce.history.server.http.address	*$jobtracker.full.hostname*:51111	Enter your JobTracker hostname

Table 3.4. taskcontroller.cfg

Property	Example	Description
Property	Example	Description
mapred.local.dir	/grid/hadoop/mapred,/grid1/hadoop/mapred	Comma separated list of paths. Use the list of directories from *$MAPREDUCE_LOCAL_DIR*

3.3. Copy the Configuration Files

Replace the installed Hadoop configs with the modified core_hadoop configuration files and set appropriate permissions.

```
rm -rf $HADOOP_CONF_DIR
mkdir -p $HADOOP_CONF_DIR
```

<Copy all the modified configuration files in core_hadoop to *$HADOOP_CONF_DIR* on all nodes>

```
chmod a+x $HADOOP_CONF_DIR/
chown -R $HDFS_USER:$HADOOP_GROUP $HADOOP_CONF_DIR/../
chmod -R 755 $HADOOP_CONF_DIR/../
```

4. Validating the Core Hadoop Installation

This section describes starting Core Hadoop and doing simple smoke tests.

4.1. Format and Start HDFS

1. Execute these commands on the NameNode. Login as :

```
/usr/lib/hadoop/bin/hadoop namenode -format
/usr/lib/hadoop/bin/hadoop-daemon.sh --config $HADOOP_CONF_DIR start
 namenode
```

2. Execute these commands on the SecondaryNameNode. Login as*$HDFS_USER* :

```
/usr/lib/hadoop/bin/hadoop-daemon.sh --config $HADOOP_CONF_DIR start
 secondarynamenode
```

3. Execute these commands on all DataNodes. Login as *$HDFS_USER*:

```
/usr/lib/hadoop/bin/hadoop-daemon.sh --config $HADOOP_CONF_DIR start
 datanode
```

4.2. Smoke Test HDFS

1. See if you can reach the NameNode server with your browser:

```
http://{namenode.full.hostname}:50070
```

2. Try copying a file into HDFS and listing that file. Login as *$HDFS_USER*:

```
/usr/lib/hadoop/bin/hadoop dfs -copyFromLocal /etc/passwd passwd-test
/usr/lib/hadoop/bin/hadoop dfs -ls
```

3. Test browsing HDFS:

```
http://$datanode.full.hostname:50075/browseDirectory.jsp?dir=/
```

4.3. Start MapReduce

1. Execute these commands from the JobTracker server. Login as *$HDFS_USER*:

```
/usr/lib/hadoop/bin/hadoop fs -mkdir /mapred
/usr/lib/hadoop/bin/hadoop fs -chown -R mapred /mapred
```

   Login as *$MAPRED_USER*

```
/usr/lib/hadoop/bin/hadoop-daemon.sh --config $HADOOP_CONF_DIR start
 jobtracker
```

2. Execute these commands from the JobHistory server. Login as *$MAPRED_USER*:

```
/usr/lib/hadoop/bin/hadoop-daemon.sh --config $HADOOP_CONF_DIR start
 historyserver
```

3. Execute these commands from all TaskTracker nodes. Login as *$MAPRED_USER*:

```
/usr/lib/hadoop/bin/hadoop-daemon.sh --config $HADOOP_CONF_DIR start
 tasktracker
```

4.4. Smoke Test MapReduce

1. Try browsing to the JobTracker:

```
http://{jobtracker.full.hostname}:50030/
```

2. Smoke test using Teragen to generate 10GB of data and then using Terasort to sort the data. Login as *$HDFS_USER*

```
/usr/lib/hadoop/bin/hadoop jar /usr/lib/hadoop/hadoop-examples.jar teragen
 100000000 /test/10gsort/input
/usr/lib/hadoop/bin/hadoop jar /usr/lib/hadoop/hadoop-examples.jar
 terasort /test/10gsort/input /test/10gsort/output
```

5. Installing Apache Pig

This section describes installing and testing Apache Pig, a platform for creating higher level data flow programs that can be compiled into sequences of MapReduce programs, using Pig Latin, the platform's native language.

5.1. Install the Pig RPMs

On all hosts on which Pig programs will be executed, install the RPMs.

- For RHEL/CentOS

```
yum install pig
```

- For SLES

```
zypper install pig
```

5.2. Set Up Configuration Files

There are several configuration files that need to be set up for Pig.

5.2.1. Extract the Pig Configuration Files

From the file you downloaded in Download Companion Files extract the files in `configuration_files -> pig` to a temporary directory.

5.2.2. Copy the Configuration Files

On all hosts where Pig will be executed, replace the installed Pig configs with the downloaded one and set appropriate permissions:

```
rm -rf $PIG_CONF_DIR
mkdir -p $PIG_CONF_DIR
```

<Copy the all config files to *$PIG_CONF_DIR*>

```
chmod -R 755 $PIG_CONF_DIR/../
```

5.3. Validate the Installation

Use these steps to validate your installation.

5.3.1. Smoke Test Pig

1. Use a terminal window on a machine where Pig is installed and login as *$HDFS_USER* :

```
/usr/lib/hadoop/bin/hadoop dfs -copyFromLocal /etc/passwd passwd
```

2. Create the pig script file `/tmp/id.pig` with the following contents:

```
echo "A = load 'passwd' using PigStorage(':'); " > /tmp/id.pig
echo "B = foreach A generate \$0 as id; store B into '/tmp/id.out'; " >> /
tmp/id.pig
```

3. Execute the Pig script:

```
pig -l /tmp/pig.log /tmp/id.pig
```

6. Installing Apache Hive and Apache HCatalog

This section describes installing and testing Apache Hive, a tool for creating higher level SQL-like queries using HiveQL, the tool's native language that can then be compiled into sequences of MapReduce programs. It also describes installing and testing Apache HCatalog, a metadata abstraction layer that insulates users and scripts from how and where data is physically stored.

6.1. Install the Hive and HCatalog RPMs

On all Hive client/gateway nodes (on which Hive programs will be executed), Hive Metastore Server, and HiveServer2 machine, install the Hive RPMs.

- For RHEL/CentOS:

```
yum install hive hcatalog
```

- For SLES:

```
zypper install hive hcatalog
```

6.2. Set Directories and Permissions

The `scripts.zip` file you downloaded in Download Companion Files includes two scripts, `usersAndGroups.sh` and `directories.sh`, for setting environment parameters. We strongly suggest you edit and execute these scripts to fit your environment. See Define Environment Parameters for more information.

6.2.1. Create Log Directories

Execute these commands on the Hive server machine:

```
mkdir -p $HIVE_LOG_DIR
chown -R $HIVE_USER:$HADOOP_GROUP $HIVE_LOG_DIR
chmod -R 755 $HIVE_LOG_DIR
```

6.3. Set Up the Hive/HCatalog Configuration Files

There are several configuration files that need to be set up for Hive/HCatalog.

6.3.1. Extract the Hive/HCatalog Configuration Files

From the file you downloaded in Download Companion Files, extract the files in `configuration_files -> hive` to a temporary directory.

6.3.2. Modify the Configuration Files

In the temporary directory, locate the following file and modify the properties based on your environment. Search for TODO in the files for the properties to replace.

Table 6.1. hive-site.xml

Property	Example	Description

javax.jdo.option.ConnectionURL	For MySQL database:
	`jdbc:mysql://$mysql.full.hostname:3306/$database.name?createDatabaseIfNotExist=true`
	
	For Oracle database:
	
	`jdbc:oracle:thin:@$dbhost:1521/$hive_dbname`
		Enter your JDBC connection string.

javax.jdo.option.ConnectionDriverName	For MySQL database:
	`com.mysql.jdbc.Driver`
	
	For Oracle database:
	`oracle.jdbc.driver.OracleDriver`
		JDBC Connection Driver Name.

javax.jdo.option.ConnectionUserName	`dbusername`	Enter your MySQL/Oracle user credentials from Database Requirements.
javax.jdo.option.ConnectionPassword	`dbuserpassword`	
hive.metastore.uris	`thrift://$metastore.server.full.hostname:9083`	URI for client to contact metastore server. To enable HiveServer2, leave the property value empty.

6.3.3. Copy the Configuration Files

On all Hive hosts replace the installed Hive configs with the modified configuration files and set appropriate permissions:

```
rm -rf $HIVE_CONF_DIR
mkdir -p $HIVE_CONF_DIR
```

<Copy all the modified configuration files in hive to *$HIVE_CONF_DIR* on all the nodes>

```
chown -R $HIVE_USER:$HADOOP_GROUP $HIVE_CONF_DIR/../
chmod -R 755 $HIVE_CONF_DIR/../
```

6.4. Create Directories on HDFS

1. Create Hive user home on HDFS. Login as *$HDFS_USER*

   ```
   hadoop fs -mkdir /user/$HIVE_USER
   hadoop fs -chown $HIVE_USER:$HIVE_USER/user/$HIVE_USER
   ```

2. Create warehouse directory on HDFS.

   ```
   hadoop fs -mkdir /apps/hive/warehouse
    hadoop fs -chown -R $HIVE_USER:users /apps/hive/warehouse
    hadoop fs -chmod -R 775 /apps/hive/warehouse
   ```

6.5. Download The Database Connector

By default, Hive uses embedded Derby database for its metastore. However, you can choose to enable remote database (either MySQL or Oracle) for Hive metastore.

Use the following instructions to download the appropriate database connector for your cluster:

1. Ensure that you complete the instructions provided here.
2. Unzip and copy the downloaded JAR file the `/usr/lib/hive/lib/` directory on your Hive host machine.
3. Ensure that the JAR file has appropriate permissions.

6.6. Validate the Installation

Use these steps to validate your installation.

1. Start Hive Metastore service. Login as *$HIVE_USER*

```
nohup hive --service metastore>$HIVE_USER/hive.out 2>$HIVE_USER/hive.log &
```

2. Smoke Test Hive.
   a. Open Hive command line shell.

```
hive
```

   b. Run sample commands.

```
show databases;
create table test(col1 int, col2 string);
show tables;
```

3. Start HiveServer2.

```
 /usr/lib/hive/bin/hiveserver2 -hiveconf hive.metastore.uris=" " >
$HIVE_LOG_DIR/hiveserver2.out 2> $HIVE_LOG_DIR/hiveserver2.log &
```

4. Smoke Test HiveServer2.
   a. Open Beeline command line shell to interact with HiveServer2.

```
/usr/lib/hive/bin/beeline
```

   b. Establish connection to server.

```
!connect jdbc:hive2://$FQDN_Hive_hostmachine:10000 $HIVE_USER password
 org.apache.hive.jdbc.HiveDriver
```

   c. Run sample commands.

```
show databases;
create table test2(a int, b string);
show tables;
```

7. Installing WebHCat

This section describes installing and testing WebHCat, which provides a REST interface to Apache HCatalog services like job submission and eventing.

7.1. Install the WebHCat RPMs

On the WebHCat server machine, install the necessary RPMs.

- For RHEL/CentOS:

```
yum install hcatalog webhcat-tar-hive webhcat-tar-pig
```

- For SLES:

```
zypper install hcatalog webhcat-tar-hive webhcat-tar-pig
```

7.2. Set Directories and Permissions

The `scripts.zip` file you downloaded in Download Companion Files includes two scripts, `usersAndGroups.sh` and `directories.sh`, for setting environment parameters. We strongly suggest you edit and execute these scripts to fit your environment. See Define Environment Parameters for more information.

Execute these commands on your WebHCat server machine to create log and pid directories.

```
mkdir -p $WEBHCAT_LOG_DIR
chown -R $WEBHCAT_USER:$HADOOP_GROUP $WEBHCAT_LOG_DIR
chmod -R 755 $WEBHCAT_LOG_DIR
```

```
mkdir -p $WEBHCAT_PID_DIR
chown -R $WEBHCAT_USER:$HADOOP_GROUP $WEBHCAT_PID_DIR
chmod -R 755 $WEBHCAT_PID_DIR
```

7.3. Modify WebHCat Config Files

Use the following instructions to modify the WebHCat config files:

1. Extract the WebHCat configuration files

   From the file you downloaded in Download Companion Files extract the files in `configuration_files/templeton` directory to a temporary location.

2. Modify the configuration files

   In the temporary directory, locate the following files and modify the properties based on your environment. Search for TODO in the files for the properties to replace. See Define Environment Parameters for more information.

Table 7.1. webhcat-env.xml

Property	Example	Description
templeton.hive.properties	`hive.metastore.local=false, hive.metastore.uris=thrift://` *`$metastore.server.full.hostname`*`:9083, hive.metastore.sasl.enabled=no,hive.metastore.execute.setugi=true`	Properties to set when running Hive.
templeton.zookeeper.hosts	*`$zookeeper1.host.FQDN`*`:2181,` *`$zookeeper1.host.FQDN`*`:2181,`	ZooKeeper servers, as comma separated HOST:PORT pairs.

7.4. Set Up the WebHCat Configuration Files

There are several configuration files that need to be set up for WebHCat.

7.4.1. Extract the WebHCat Configuration Files

From the file you downloaded in Download Companion Files, extract the files in `configuration_files/webhcat` directory to a temporary location.

7.4.2. Copy the Configuration Files

Copy the modified WebHCat configuration files and set appropriate permissions:

```
rm -rf $WEBHCAT_CONF_DIR/*
```

Copy all the config files to *$WEBHCAT_CONF_DIR*

```
chown -R $WEBHCAT_USER:$HADOOP_GROUP $WEBHCAT_CONF_DIR
chmod -R 755 $WEBHCAT_CONF_DIR
```

7.5. Set Up the HDFS User and Prepare WebHCat Directories On HDFS

1. Set up the HDFS user. Login as *$HDFS_USER*

```
 hadoop fs -mkdir /user/$WEBHCAT_USER
hadoop fs -chown -R $WEBHCAT_USER:$WEBHCAT_USER /user/$WEBHCAT_USER
hadoop fs -mkdir /apps/webhcat
```

2. Prepare WebHCat directories on HDFS.

```
hadoop dfs -copyFromLocal /usr/share/HDP-webhcat/pig.tar.gz /apps/webhcat/
hadoop dfs -copyFromLocal /usr/share/HDP-webhcat/hive.tar.gz /apps/webhcat/
hadoop dfs -copyFromLocal /usr/lib/hadoop/contrib/streaming/hadoop-
streaming*.jar /apps/webhcat/
```

3. Set appropriate permissions for the HDFS user and the webhcat directory.

```
hadoop fs -chown -R $WEBHCAT_USER:users /apps/webhcat
hadoop fs -chmod -R 755 /apps/webhcat
```

7.6. Validate the Installation

1. Start the WebHCat server.

```
<login as $WEBHCAT_USER>
/usr/lib/hcatalog/sbin/webhcat_server.sh start
```

2. From the browser, type:

```
http://$FQDN_of_WebHCat_host_machine:50111/templeton/v1/status
```

   You should see the following output:

```
{"status":"ok","version":"v1"}
```

8. Installing Apache Oozie

This section describes installing and testing Apache Oozie, a server based workflow engine optimized for running workflows that execute Hadoop jobs.

8.1. Install the Oozie RPMs

1. On Oozie server machine, install the necessary RPMs.
   - For RHEL/CentOS:

     ```
     yum install oozie extjs
     ```

   - For SLES:

     ```
     zypper install oozie extjs
     ```

2. Add the ExtJS library to the Oozie application.

   ```
   /usr/lib/oozie/bin/oozie-setup.sh -hadoop 0.20.200 /usr/lib/hadoop -extjs /
   usr/share/HDP-oozie/ext-2.2.zip
   ```

3. Add LZO JAR file.

   ```
   /usr/lib/oozie/bin/oozie-setup.sh -hadoop 0.20.200 /usr/lib/hadoop -extjs /
   usr/share/HDP-oozie/ext-2.2.zip -jars /usr/lib/hadoop/lib/hadoop-lzo-0.5.0.
   jar
   ```

8.1.1. Optional: Add Database Connector JAR Files

If you are using external database (MySQL/Oracle) for Oozie metastore, you must add appropriate database connector JAR files using the instructions provided below:

1. Complete the instructions provided here.
2. Add the downloaded JAR files.
   - For MySQL:

     ```
     /usr/lib/oozie/bin/oozie-setup.sh -hadoop 0.20.200 /usr/lib/hadoop -
     extjs /usr/share/HDP-oozie/ext-2.2.zip -jars /usr/lib/hadoop/lib/hadoop-
     lzo-0.5.0.jar:/usr/lib/oozie/libtools/mysql-connector-java-5.1.18-bin.jar
     ```

   - For Oracle:

     ```
     /usr/lib/oozie/bin/oozie-setup.sh -hadoop 0.20.200 /usr/lib/hadoop -
     extjs /usr/share/HDP-oozie/ext-2.2.zip -jars /usr/lib/hadoop/lib/hadoop-
     lzo-0.5.0.jar:/usr/lib/oozie/libtools/odbc6.jar
     ```

8.2. Set Directories and Permissions

The `scripts.zip` file you downloaded in Download Companion Files includes two scripts, `usersAndGroups.sh` and `directories.sh`, for setting environment

parameters. We strongly suggest you edit and execute these scripts to fit your environment. See Define Environment Parameters for more information.

8.2.1. Create Log Directories

Execute these commands on your Oozie server machine.

```
mkdir -p $OOZIE_DATA
chown -R $OOZIE_USER:$HADOOP_GROUP $OOZIE_DATA
chmod -R 755 $OOZIE_DATA
```

```
mkdir -p $OOZIE_LOG_DIR
chown -R $OOZIE_USER:$HADOOP_GROUP $OOZIE_LOG_DIR
chmod -R 755 $OOZIE_LOG_DIR
```

```
mkdir -p $OOZIE_PID_DIR
chown -R $OOZIE_USER:$HADOOP_GROUP $OOZIE_PID_DIR
chmod -R 755 $OOZIE_PID_DIR
```

```
mkdir -p $OOZIE_TMP_DIR
chown -R $OOZIE_USER:$HADOOP_GROUP $OOZIE_TMP_DIR
chmod -R 755 $OOZIE_TMP_DIR
```

8.3. Set Up the Oozie Configuration Files

There are several configuration files that need to be set up for Oozie.

8.3.1. Extract the Oozie Configuration Files

From the file you downloaded in Download Companion Files, extract the files in `configuration_files/oozie` directory to a temporary location.

8.3.2. Modify the Configuration Files

In the temporary directory, locate the following file and modify the properties based on your environment. Search for TODO in the files for the properties to replace.

Table 8.1. oozie-site.xml

Property	Example	Description
oozie.base.url	`http://{oozie.full.hostname}:11000/oozie`	Enter your Oozie server hostname
oozie.service.JPAService.jdbc.driver	For Derby: `org.apache.derby.jdbc.EmbeddedDriver` For MySQL: `com.mysql.jdbc.Driver` For Oracle: `oracle.jdbc.driver.OracleDriver`	JDBC Driver class.
oozie.service.JPAService.jdbc.url	For Derby: `jdbc:derby:$oozie.data.dir/$oozie.db.schema.name-db;create=true`	JDBC URL.

Property	Example	Description

	For MySQL:
	`jdbc:mysql://$dbhost:3306/$dbname`
	
	For Oracle:
	`jdbc:oracle:thin:@$dbhost:1521:$dbname`	

oozie.service.JPAService.jdbc.username	`oozie_dbuser`	Oozie database user credentials created using the instructions provided here [3].
oozie.service.JPAService.jdbc.password	`oozie_dbpasswd`	

Table 8.2. oozie-env.sh

Variable	Example	Description
OOZIE_LOG_DIR	`/var/log/oozie`	Use value from *$OOZIE_LOG_DIR*
OOZIE_PID_DIR	`/var/run/oozie`	Use value from *$OOZIE_PID_DIR*
OOZIE_DATA_DIR	`/var/db/oozie`	Use value from *$OOZIE_DATA_DIR*

8.3.3. Copy the Configuration Files

On your Oozie server machine, replace the installed Oozie configs with the modified Oozie configuration files and set appropriate permissions.

Also create a new config directory, copy the config files, and set the permissions:

```
rm -rf $OOZIE_CONF_DIR
mkdir -p $OOZIE_CONF_DIR
```

<Copy all the config files to *$OOZIE_CONF_DIR*>

```
chown -R $OOZIE_USER:$HADOOP_GROUP $OOZIE_CONF_DIR/../
chmod -R 755 $OOZIE_CONF_DIR/../
```

8.4. Initialize Database Schema

On the Oozie host machine, execute the following command. Login as *$OOZIE_USER*

```
/usr/lib/oozie/bin/ooziedb.sh create -run
```

8.5. Install the Oozie Sharelib in Hadoop HDFS

On the Oozie host machine, execute the following commands:

```
cd /usr/lib/oozie
tar -xzf oozie-sharelib.tar.gz
```

Login as *$HDFS_USER*

```
hadoop fs -mkdir /user/$OOZIE_USER
hadoop fs -copyFromLocal /usr/lib/oozie/share /user/$OOZIE_USER/.
hadoop fs -chown $OOZIE_USER:hadoop /user/$OOZIE_USER
```

```
hadoop fs -chmod -R 755 /user/$OOZIE_USER
```

8.6. Validate the Installation

Use these steps to validate your installation.

1. Start Oozie. Login as *$OOZIE_USER*

   ```
   /usr/lib/oozie/bin/oozie-start.sh
   ```

2. Smoke Test Oozie.

   a. Confirm that you can browse to the Oozie server:

   ```
   http://$oozie.full.hostname:11000/oozie
   ```

   b. Access the Oozie Server with the Oozie client.

   ```
   oozie admin -oozie http://$oozie.full.hostname:11000/oozie -status
   ```

   You should see the following output:

   ```
   System mode: NORMAL
   ```

9. Installing HBase and ZooKeeper

This section describes installing and testing Apache HBase, a distributed, column-oriented database that provides the ability to access and manipulate data randomly in the context of the large blocks that make up HDFS. It also describes installing and testing Apache ZooKeeper, a centralized tool for providing services to highly distributed systems.

9.1. Install the HBase and ZooKeeper RPMs

1. Execute the following command on Zookeeper nodes and gateway node:
   - For RHEL/CentOS:

     ```
     yum install zookeeper
     ```

   - For SLES:

     ```
     zypper install zookeeper
     ```

2. Execute the following command on HBaseMaster node, RegionServer nodes and the gateway node:
   - For RHEL/CentOS:

     ```
     yum install hbase
     ```

   - For SLES:

     ```
     zypper install hbase
     ```

9.2. Set Directories and Permissions

The `scripts.zip` file you downloaded in Download Companion Files includes two scripts, `usersAndGroups.sh` and `directories.sh`, for setting environment parameters. We strongly suggest you edit and execute these scripts to fit your environment. See Define Environment Parameters for more information.

9.2.1. Create Log and PID Directories

1. Execute these commands on the HBaseMaster node and RegionServer nodes:

   ```
   mkdir -p $HBASE_LOG_DIR
   chown -R $HBASE_USER:$HADOOP_GROUP $HBASE_LOG_DIR
   chmod -R 755 $HBASE_LOG_DIR
   ```

   ```
   mkdir -p $HBASE_PID_DIR
   chown -R $HBASE_USER:$HADOOP_GROUP $HBASE_PID_DIR
   chmod -R 755 $HBASE_PID_DIR
   ```

2. Execute these commands on the ZooKeeper 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
```

9.3. Set Up the Configuration Files

There are several configuration files that need to be set up for HBase and ZooKeeper.

9.3.1. Extract the HBase/ZooKeeper Configuration Files

From the file you downloaded in Download Companion Files, extract the files in `configuration_files/hbase` and `configuration_files/ zookeeper` directories to two temporary directories.

9.3.2. Modify the Configuration Files

In the respective temporary directories, locate the following files and modify the properties based on your environment. Search for TODO in the files for the properties to replace.

Table 9.1. zoo.cfg

Variable	Example	Description
server.1	*$zookeeper.server1.full.hostname* :2888:3888	Enter the 1st ZooKeeper hostname
server.2	*$zookeeper.server2.full.hostname*:2888:3888	Enter the 2nd ZooKeeper hostname
server.3	*$zookeeper.server3.full.hostname*:2888:3888	Enter the 3rd ZooKeeper hostname

Table 9.2. hbase-site.xml

Variable	Example	Description
hbase.rootdir	hdfs:// *$namenode.full.hostname* :8020/apps/hbase/data	Enter the NameNode server FQDN.
hbase.master.info.bindAddress	*$hbase.master.full.hostname*	Enter the HBase Master server FQDN
hbase.zookeeper.quorum	server1.full.hostname, server2.full.hostname, server3.full.hostname	Comma separated list of Zookeeper servers (match to what is specified in zoo.cfg but without portnumbers)

9.3.3. Copy the Configuration Files

1. Replace the installed ZooKeeper configs with the modified ZooKeeper configuration files and set appropriate permissions.

```
rm -rf $ZOOKEEPER_CONF_DIR/*
```

<Copy all the config files to *$ZOOKEEPER_CONF_DIR*>

```
chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_CONF_DIR/../
chmod -R 755 $ZOOKEEPER_CONF_DIR/../
```

2. Replace the installed HBase configs with the modified HBase configuration files and set appropriate permissions.

```
rm -rf $HBASE_CONF_DIR
```

<Copy all the config files to *$HBASE_CONF_DIR* >

```
chown -R $HBASE_USER:$HADOOP_GROUP $HBASE_CONF_DIR/../  ;
chmod -R 755 $HBASE_CONF_DIR/../
```

9.4. Validate the Installation

Use these steps to validate your installation.

9.4.1. Start HBase and ZooKeeper

1. Start ZooKeeper. Execute this command from each ZooKeeper node. Login as *$ZOOKEEPER_USER*:

```
/usr/lib/zookeeper/bin/zkServer.sh start $ZOOKEEPER_CONF_DIR/zoo.cfg
```

2. Create HBase user home on HDFS from any host. Login as *$HDFS_USER*:

```
hadoop fs -mkdir /user/$HBASE_USER
hadoop fs -chown $HBASE_USER:$HBASE_USER/user/$HBASE_USER
```

3. Create HBase root directories on HDFS from any host. Login as *$HDFS_USER*:

```
hadoop fs -mkdir /apps/hbase/data
hadoop fs -chown -R $HBASE_USER:$HBASE_USER/apps/hbase
```

4. Start the HBase Master. Execute this command from the HBase Master node. Login as *$HBASE_USER*:

```
 /usr/lib/hbase/bin/hbase-daemon.sh --config $HBASE_CONF_DIRstart master
```

5. Start the RegionServers. Execute this command from each HBase RegionServer node. Login as *$HBASE_USER*:

```
/usr/lib/hbase/bin/hbase-daemon.sh --config $HBASE_CONF_DIR start
 regionserver
```

9.4.2. Smoke Test HBase and ZooKeeper

Use the following instructions to smoke test HBase and ZooKeeper:

1. Open the HBase command line tool

```
hbase shell
```

2. On the HBase shell prompt execute the following commands:

```
status
help
create 'usertable', 'family'
put 'usertable', 'row01', 'family:col01', 'value1'
scan 'usertable'
disable 'usertable'
drop 'usertable'
```

10. Installing Apache Sqoop

This section describes installing and testing Apache Sqoop, a component that provides a mechanism for moving data between HDFS and external structured datastores.

10.1. Install the Sqoop RPMs

On all nodes where you plan to use the Sqoop client, install the RPMs:

- For RHEL/CentOS:

```
yum install sqoop
```

- For SLES:

```
zypper install sqoop
```

10.2. Optional: Download Database Connector

If you plan to migrate data from HDFS/Hive/HBase to database, you must have appropriate database connector (MySQL/Oracle) JAR file.

Use the following instructions to add appropriate database connector:

1. Complete the instructions listed here: Minimum requirements - Database requirements
2. Copy the JAR file to `/usr/lib/sqoop/lib`.

10.3. Set Up the Sqoop Configuration

There are several configuration files that need to be set up for Sqoop.

10.3.1. Extract the Sqoop Configuration Files

From the file you downloaded in Download Companion Files extract the files in `configuration_files/sqoop` directory to a temporary location.

10.3.2. Copy the Configuration Files

Copy all the config files to the Sqoop conf dir.

```
<copy the config files to $SQOOP_CONF_DIR >
```

10.4. Validate the Installation

Use this step to validate your installation.

Execute the following command. You should see the Sqoop version information displayed.

```
sqoop version
```

11. Installing Ganglia

This section describes installing and testing Ganglia, a system for monitoring and capturing metrics from services and components of the Hadoop cluster.

11.1. Install the Ganglia RPMs

On the host you have chosen to be the Ganglia server, install the server RPMs:

- For RHEL/CentOS:

```
yum install ganglia-gmond-3.2.0-99 ganglia-gmetad-3.2.0-99 gweb-2.2.0-99
 hdp_mon_ganglia_addons
```

- For SLES:

```
zypper install ganglia-gmond-3.2.0-99 ganglia-gmetad-3.2.0-99 gweb-2.2.0-99
 hdp_mon_ganglia_addons
```

On each host in the cluster, install the client RPMs:

- For RHEL/CentOS:

```
yum install ganglia-gmond-3.2.0-99
```

- For SLES:

```
zypper install ganglia-gmond-3.2.0-99
```

11.2. Install the Configuration Files

There are several configuration files that need to be set up for Ganglia.

11.2.1. Extract the Ganglia Configuration Files

From the file you downloaded in Download Companion Files, open the `configuration_files.zip` and copy the files in the `ganglia` folder to a temporary directory. The `ganglia` folder contains two sub-folders, `objects` and `scripts`.

11.2.2. Copy the Configuration Files

On the Ganglia server host:

1. Create the directory for the `objects` folder:

```
mkdir -p /usr/libexec/hdp/ganglia
```

2. Copy the objects files:

```
cp <tmp-directory>/ganglia/objects /usr/libexec/hdp/ganglia
```

3. Copy the contents of the scripts folder to `init.d`

```
cp <tmp-directory>/ganglia/scripts /etc/init.d
```

On each host in the cluster:

1. Copy the Ganglia monitoring init script to `init.d`

```
cp <tmp-directory>/ganglia/scripts/hdp-gmond /etc/init.d
```

11.2.3. Set Up Ganglia Hosts

1. On the Ganglia server, to configure the `gmond` collector:

```
/usr/libexec/hdp/ganglia/setupGanglia.sh -c HDPJobTracker -m
/usr/libexec/hdp/ganglia/setupGanglia.sh -c HDPNameNode -m
/usr/libexec/hdp/ganglia/setupGanglia.sh -c HDPSlaves -m
/usr/libexec/hdp/ganglia/setupGanglia.sh -t
```

2. If HBase is installed, on the HBase Master:

```
/usr/libexec/hdp/ganglia/setupGanglia.sh -c HDPHBaseMaster -m
```

3. On the NameNode and SecondaryNameNode servers, to configure the `gmond` emitters:

```
/usr/libexec/hdp/ganglia/setupGanglia.sh -c HDPNameNode
```

4. On the JobTracker server, to configure the `gmond` emitters:

```
/usr/libexec/hdp/ganglia/setupGanglia.sh -c HDPJobTracker
```

5. On all hosts, to configure the `gmond` emitters:

```
/usr/libexec/hdp/ganglia/setupGanglia.sh -c HDPSlaves
```

6. If HBase is installed, on the HBase Master, to configure the `gmond` emitter:

```
/usr/libexec/hdp/ganglia/setupGanglia.sh -c HDPHBaseMaster
```

11.2.4. Set Up Configurations

1. On the Ganglia server, use a text editor to open the following master configuration files:

```
/etc/ganglia/hdp/HDPNameNode/conf.d/gmond.master.conf
/etc/ganglia/hdp/HDPJobTracker/conf.d/gmond.master.conf
/etc/ganglia/hdp/HDPSlaves/conf.d/gmond.master.conf
```

   And if HBase is installed:

```
/etc/ganglia/hdp/HDPHBaseMaster/conf.d/gmond.master.conf
```

2. Confirm that the “bind” property in each of these files is set to the Ganglia server hostname.

3. On the Ganglia server, use a text editor to open the `gmetad` configuration file:

   `/etc/ganglia/hdp/gmetad.conf`

4. Confirm the "data_source" properties are set to the Ganglia server hostname. For example:

```
data_source "HDPSlaves" my.ganglia.server.hostname:8660
data_source "HDPNameNode" my.ganglia.server.hostname:8661
data_source "HDPJobTracker" my.ganglia.server.hostname:8662
```

And if HBase is installed:

```
data_source "HDPHBaseMaster" my.ganglia.server.hostname:8663
```

5. On all hosts except the Ganglia server, use a text editor to open the slave configuration files:

```
/etc/ganglia/hdp/HDPNameNode/conf.d/gmond.slave.conf
/etc/ganglia/hdp/HDPJobTracker/conf.d/gmond.slave.conf
/etc/ganglia/hdp/HDPSlaves/conf.d/gmond.slave.conf
```

And if HBase is installed

```
/etc/ganglia/hdp/HDPHBaseMaster/conf.d/gmond.slave.conf
```

6. Confirm that the `host` property is set to the Ganglia Server hostname.

11.2.5. Set Up Hadoop Metrics

On each host in the cluster:

1. Stop the Hadoop services.
2. Change to the Hadoop configuration directory.

```
cd $HADOOP_CONF_DIR
```

3. Copy the Ganglia metrics properties file into place.

```
mv hadoop-metrics2.properties-GANGLIA  hadoop-metrics2.properties
```

4. Edit the metrics properties file and set the Ganglia server hostname.

```
namenode.sink.ganglia.servers=my.ganglia.server.hostname:8661
datanode.sink.ganglia.servers=my.ganglia.server.hostname:8660
jobtracker.sink.ganglia.servers=my.ganglia.server.hostname:8662
tasktracker.sink.ganglia.servers=my.ganglia.server.hostname:8660
maptask.sink.ganglia.servers=my.ganglia.server.hostname:8660
reducetask.sink.ganglia.servers=my.ganglia.server.hostname:8660
```

5. Restart the Hadoop services.

11.3. Validate the Installation

Use these steps to validate your installation.

11.3.1. Start the Ganglia Server

On the Ganglia server:

```
service httpd restart
/etc/init.d/hdp-gmetad start
```

11.3.2. Start Ganglia Monitoring on All Hosts

On all hosts:

```
/etc/init.d/hdp-gmond start
```

11.3.3. Confirm that Ganglia is Running

Browse to the Ganglia server:

```
http://{ganglia.server}/ganglia
```

12. Installing Nagios

This section describes installing and testing Nagios, a system that monitors Hadoop cluster components and issues alerts on warning and critical conditions.

12.1. Install the Nagio RPMs

On the host you have chosen to be the Nagios server, install the RPMs:

- For RHEL and CentOS

```
yum install net-snmp net-snmp-utils php-pecl-json
yum install wget httpd php net-snmp-perl perl-Net-SNMP fping nagios-3.2.3
 nagios-plugins-1.4.9 hdp_mon_nagios_addons
```

- For SLES

```
zypper install net-snmp
zypper install wget apache2 php php-curl perl-SNMP perl-Net-SNMP fping
 nagios-3.2.3-2.1 nagios-plugins-1.4.9 hdp_mon_nagios_addons
```

12.2. Install the Configuration Files

There are several configuration files that need to be set up for Nagios.

12.2.1. Extract the Nagios Configuration Files

From the file you downloaded in Download Companion Files, open the `configuration_files.zip` and copy the files in the nagios folder to a temporary directory. The `nagios` folder contains two sub-folders, `objects` and `plugins`.

12.2.2. Copy the Configuration Files

1. Copy the contents of the `objects` folder into place:

```
cp $tmp-directory/nagios/objects /etc/nagios/objects
```

2. Copy the contents of the `plugins` folder into place:

```
cp $tmp-directory/nagios/plugins /usr/lib64/nagios/plugins
```

12.2.3. Set the Nagios Admin Password

1. Choose a Nagios administrator password, for example, “admin”
2. Use the following command to set the password:

```
htpasswd -c -b  /etc/nagios/htpasswd.users nagiosadmin admin
```

12.2.4. Set the Nagios Admin Email Contact Address

1. Open `/etc/nagios/objects/contacts.cfg` with a text editor.

2. Change the `nagios@localhost` value to the admin email address for recceiving alerts.

12.2.5. Register the Hadoop Configuration Files

1. Open `/etc/nagios/nagios.cfg` with a text editor.

2. In the section "`OBJECT CONFIGURATION FILE(S)`", add the following:

```
   # Definitions for hadoop servers
cfg_file=/etc/nagios/objects/hadoop-commands.cfg
cfg_file=/etc/nagios/objects/hadoop-hosts.cfg
cfg_file=/etc/nagios/objects/hadoop-hostgroups.cfg
cfg_file=/etc/nagios/objects/hadoop-services.cfg
cfg_file=/etc/nagios/objects/hadoop-servicegroups.cfg
```

12.2.6. Set Hosts

1. Open `/etc/nagios/objects/hadoop-hosts.cfg` with a text editor.

2. Create a "`define host { ... }`" entry for each host in your cluster using the following format:

```
define host {
       alias @HOST@
       host_name @HOST@
       use linux-server
       address @HOST@
       check_interval 0.25
       retry_interval 0.25
       max_check_attempts 4
       notifications_enabled 1
       first_notification_delay 0 # Send notification soon after
                                  change in the hard state
       notification_interval 0    # Send the notification once
       notification_options       d,u,r
  }
```

3. Replace the "@HOST@" with the hostname.

12.2.7. Set Host Groups

1. Open `/etc/nagios/objects/hadoop-hostsgroups.cfg` with a text editor.

2. Create host groups based on all the hosts and services you have installed in your cluster. Each host group entry should follow this format:

```
define hostgroup {
       hostgroup_name  @NAME@
       alias           @ALIAS@
       members         @MEMBERS@
   }
```

Where

Table 12.1. Host Group Parameters

Parameter	Description
@NAME@	The host group name
@ALIAS@	The host group alias
@MEMBERS@	A comma-separated list of hosts in the group

3. The following table lists the core and monitoring host groups:

Table 12.2. Core and Monitoring Hosts

Service	Component	Name	Alias	Members
All servers in the cluster		all-servers	All Servers	List all servers in the cluster
HDFS	NameNode	namenode	namenode	The NameNode host
HDFS	SecondaryNameNode	snamenode	snamenode	The Secondary NameNode host
MapReduce	JobTracker	jobtracker	jobtracker	The Job Tracker host
HDFS, MapReduce	Slaves	slaves	slaves	List all hosts running DataNode and TaskTrackers
Nagios		nagios-server	nagios-server	The Nagios server host
Ganglia		ganglia-server	ganglia-server	The Ganglia server host

4. The following table lists the ecosystem project host groups:

Table 12.3. Ecosystem Hosts

Service	Component	Name	Alias	Members
HBase	Master	hbasemaster	hbasemaster	List the master server
HBase	Region	regions-servers	region-servers	List all region servers
ZooKeeper		zookeeper-servers	zookeeper-servers	List all ZooKeeper servers
Oozie		oozie-server	oozie-server	The Oozie server
Hive		hiveserver	hiverserver	The Hive metastore server
WebHCat		twebhcat-server	webhcat-server	The WebHCat server

12.2.8. Set Services

1. Open `/etc/nagios/objects/hadoop-services.cfg` with a text editor.

   This file contains service definitions for the following services: Ganglia, HBase (Master and Region), ZooKeeper, Hive, Templeton and Oozie

2. Remove any services definitions for services you have not installed.

3. Replace the parameter `@NAGIOS_BIN@` and `@STATUS_DAT@` parameters based on the operating system.

- For RHEL and CentOS

```
@STATUS_DAT@ = /var/nagios/status.dat
@NAGIOS_BIN@ = /usr/bin/nagios
```

- For SLES

```
@STATUS_DAT@ = /var/lib/nagios/status.dat
@NAGIOS_BIN@ = /usr/sbin/nagios
```

4. If you have installed Hive or Oozie services, replace the parameter `@JAVA_HOME@` with the path to the Java home. For example, `/usr/java/default`.

12.2.9. Set Status

1. Open `/etc/nagios/objects/hadoop-commands.cfg` with a text editor.
2. Replace the @STATUS_DAT@ parameter with the location of the Nagios status file. The file is located:

- For RHEL and CentOS

```
/var/nagios/status.dat
```

- For SLES

```
 /var/lib/nagios/status.dat
```

12.3. Validate the Installation

Use these steps to validate your installation.

12.3.1. Start Nagios

Start the Nagios server

```
/etc/init.d/nagios start
```

12.3.2. Confirm Nagios is Running

Confirm the server is running

```
/etc/init.d/nagios status
```

This should return:

```
nagios (pid #) is running...
```

12.3.3. Test Nagios Services

Run the following command:

```
/usr/lib64/nagios/plugins/check_hdfs_capacity.php -h namenode_hostname -p
 50070 -w 80% -c 90%
```

This should return:

```
OK: DFSUsedGB:<some#>, DFSTotalGB:<some#>
```

12.3.4. Test Nagios Access

1. Browse to the Nagios server:

   ```
   http://$nagios.server/nagios
   ```

2. Login using the Nagios admin username (nagiosadmin) and password (see Set the Nagios Admin Password).

3. Click on **hosts** to validate that all the hosts in the cluster are listed.

4. Click on **services** to validate all the Hadoop services are listed for each host.

12.3.5. Test Nagios Alerts

1. Login to one of your cluster DataNodes.

2. Stop the TaskTracker service.

   ```
   su -l mapred -c "/usr/lib/hadoop/bin/hadoop-daemon.sh --config /etc/hadoop/
   conf stop tasktracker"
   ```

3. Validate that you received an alert at the admin email address and that you have critical state showing on the console.

4. Start the TaskTracker service.

   ```
   su -l mapred -c "/usr/lib/hadoop/bin/hadoop-daemon.sh --config /etc/hadoop/
   conf start tasktracker"
   ```

5. Validate that you received an alert at the admin email address and that critical state is cleared on the console.

13. Setting Up Security for Manual Installs

This section provides information on enabling security for a manually installed version of HDP. These steps must be taken in addition to the normal set-up procedure.

13.1. Preparing Kerberos

This section provides information on setting up Kerberos for an HDP installation.

13.1.1. Kerberos Overview

To create secure communication among its various components, HDP uses Kerberos. Kerberos is a third party authentication mechanism, in which users and services that users wish to access rely on a third party - the Kerberos server - to authenticate each to the other. This mechanism also can be used to encrypt all traffic between the user and the service, although this has a significant performance impact. The Kerberos server itself is known as the *Key Distribution Center*, or KDC. At a high level, it has three parts:

- A database of the users and services (known as *principals*) that it knows about and their respective Kerberos passwords
- An *authentication server* (AS) which performs the initial authentication and issues a *Ticket Granting Ticket* (TGT)
- A *Ticket Granting Server* (TGS) that issues subsequent service tickets based on the initial TGT.

A user principal requests authentication from the AS. The AS returns a TGT that is encrypted using the user principal's Kerberos password, which is known only to the user principal and the AS. The user principal decrypts the TGT locally using its Kerberos password, and from that point forward, until the ticket expires, the user principal can use the TGT to get service tickets from the TGS.

Because a service principal cannot provide a password each time to decrypt the TGT, it uses a special file, called a *keytab,* which contains its authentication credentials.

The service tickets are what allow the principal to access various services. The set of hosts, users, and services over which the Kerberos server has control is called a *realm*.

Note

Because Kerberos is a time-sensitive protocol, all hosts in the realm must be time-synchronized, for example, by using the Network Time Protocol (NTP). If the local system time of a client differs from that of the KDC by as little as 5 minutes (the default), the client will not be able to authenticate.

13.1.2. Installing and Configuring the KDC

To use Kerberos with HDP you can either use an existing KDC or install a new one just for HDP's use. The following gives a very high level description of the installation process. To get more information see RHEL documentation or CentOS documentation or SLES documentation.

To install a new version of the server:

- For RHEL or CentOS

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

- For SLES

```
zypper install krb5 krb5-server krb5-client
```

Note

The host on which you install the KDC must itself be secure.

When the server is installed you must edit the two main configuration files, located by default here:

For RHEL or CentOS

- `/etc/krb5.conf`
- `/var/kerberos/krb5kdc/kdc.conf`.

For SLES

- `/etc/krb5.conf`
- `/var/lib/kerberos/krb5kdc/kdc.conf`

Use these files to specify the realm by changing EXAMPLE.COM and example.com to case-matched version of the domain name for the realm and changing the KDC value from `kerberos.example.com` to the fully qualified name of the Kerberos server host.

The updated version of `/etc/krb5.conf` should be copied to every node in your cluster.

13.1.3. Creating the Database and Setting Up the First Administrator

1. Use the utility `kdb5_util` to create the Kerberos database.

   - For all OSes

   ```
   kdb5_util create -s
   ```

   The -s option allows you to store the master server key for the database in a *stash* file. If the stash file is not present, you will need to log into the KDC with the master

password (specified during installation) each time it starts. This will automatically regenerate the master server key.

2. Edit the Access Control List (`/var/kerberos/krb5kdc/kadm5.acl` in RHEL or CentOS and `/var/lib/kerberos/krb5kdc/kadm5.acl` in SLES) to define the principals that have admin (modifying) access to the database. A simple example would be a single entry:

```
*/admin@EXAMPLE.COM  *
```

   This specifies that all principals with the `/admin` *instance* extension have full access to the database. You must restart `kadmin` for the change to take effect.

3. Create the first user principal. This must be done at a terminal window on the KDC machine itself, while you are logged in as `root`. Notice the `.local`. Normal `kadmin` usage requires that a principal with appropriate access already exist.The `kadmin.local` command can be used even if no principals exist.

```
/usr/sbin/kadmin.local -q "addprinc <username>/admin"
```

   Other principals can now be created either on the KDC machine itself or through the network, using this principal. The following instruction assume you are using the KDC machine.

4. Start Kerberos.

   - For RHEL and CentOS

```
/sbin/service krb5kdc start
/sbin/service kadmin start
```

   - For SLES

```
rckrb5kdc start
rckadmind start
```

13.1.4. Creating Service Principals and Keytab Files for HDP

Each service in HDP must have its own principal. 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. 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.

Step 1: Create a service principal using the `kadmin` utility:

```
kadmin: addprinc -randkey $principal_name fully.qualified.domain.name>@YOUR-
REALM.COM
```

You must have a principal with administrative permissions to use this command. The randkey is used to generate the password.

Note that in the example each service principal's name has appended to it the fully qualified domain name of the host on which it is running. This is to provide a unique principal name for services that run on multiple hosts, like DataNodes and TaskTrackers.

The addition of the hostname 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 compromised
- If 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 timestamp, then the authentication would be rejected as a replay request.

In general the *`$principal name`* part of the name should match the values in the table below. It is possible to use different names (except you **must** use HTTP) but this requires customizing many of the steps below. **Note** that the NameNode, Secondary NameNode, and Oozie require two principals each:

Table 13.1. Service Principal Names

Service Name	Default Principal Name
NameNode	nn and HTTP
Secondary NameNode	nn and HTTP
JobTracker	jt
TaskTracker	tt
DataNode	dn
HBase Master	hm
HBase RegionServer	rs
Hive Metastore/HiveServer2	hive
Oozie	oozie and HTTP
WebHCat	HTTP

For example: To create the principal for a DataNode service, issue this command:

```
kadmin: addprinc -randkey dn/$datanode-host@EXAMPLE.COM
```

Step 2: Extract the related keytab file and place it in the keytab directory (by default `/etc/krb5.keytab`) of the appropriate respective components:

```
kadmin:  xst -norandkey -k $keytab_file_name/fully.qualified.domain.name
```

Important

The Service Keytab filenames must match the principal name whose information they contain. HTTP **must** use the spnego filename.

If you use the default names, these names would be as follows:

Table 13.2. Service Keytab File Names

Service Name	Default Keytab File Name
NameNode	nn.service.keytab AND spnego.service.keytab
Secondary NameNode	nn.service.keytab

Service Name	Default Keytab File Name
	AND spnego.service.keytab
JobTracker	jt.service.keytab
TaskTracker	tt.service.keytab
DataNode	dn.service.keytab
HBase Master	hm.service.keytab
HBase RegionServer	rs.service.keytab
Hive Metastore/HiveServer2	hive.service.keytab
Oozie	oozie.service.keytab AND spnego.service.keytab
WebHCat	spnego.service.keytab

For example: To create the keytab files for the NameNode, issue these commands:

```
kadmin: xst -k nn.service.keytab nn/$namenode-host
kadmin: xst -k spnego.service.keytab HTTP/$namenode-host
```

When you have created the keytab files, copy them to the `keytab` directory of the respective service hosts.

Step 3: Verify that the correct keytab files and principals are associated with the correct service using the `klist` command. For example, on the NameNode:

```
klist –k -t /etc/security/nn.service.keytab
```

Do this on each respective service in your cluster.

Step 4: Make the keytab file readable for other users:

```
chmod -R 644 /etc/security/keytabs
```

Do this on each respective service in your cluster.

13.1.5. Providing the jce-6 Security Jars

You must have a copy of the jce-6 security policy jars available in the `$JAVA_HOME/jre/lib/security/` directory of each of your hosts. You can download them from here:

```
http://www.oracle.com/technetwork/java/javase/downloads/jce-6-download-429243.html
```

13.2. Configuring HDP

This section provides information on configuring HDP for Kerberos.

- Configuration Overview
- Creating Mappings Between Principals and OS Service Usernames
- Creating the Database and Setting Up the First Administrator

- Creating Principals and Keytab Files for HDP

13.2.1. Configuration Overview

Configuring HDP for Kerberos has two parts:

- Creating a mapping between service principals and OS service usernames.

  Hadoop uses group memberships of users at various places, such as to determine group ownership for files or for access control.

  A user is mapped to the groups it belongs to using an implementation of the `GroupMappingServiceProvider` interface. The implementation is pluggable and is configured in `core-site.xml`.

  By default Hadoop uses `ShellBasedUnixGroupsMapping`, which is an implementation of `GroupMappingServiceProvider`. It fetches the group membership for a username by executing a UNIX shell command. In secure clusters, since the usernames are actually Kerberos principals, `ShellBasedUnixGroupsMapping` will work only if the Kerberos principals map to valid UNIX usernames. Hadoop provides a feature that lets administrators specify mapping rules to map a Kerberos principal to a local UNIX username .

- Adding information to various service configuration files.

  There are several optional entries in service configuration files that must be added to enable security on HDP.

13.2.2. Creating Mappings Between Principals and OS Service Usernames

HDP uses a rule-based system to create mappings between service principals and their related OS service usernames. The rules are specified in the `core-site.xml` configuration file as the value to the optional key `hadoop.security.auth_to_local`.

The default rule is simply named `DEFAULT`. It translates all principals in your default domain to their first component. For example, `myusername@APACHE.ORG` and `myusername/admin@APACHE.ORG` both become `myusername`, assuming your default domain is APACHE.ORG. So if the service principal and the OS service username are the same, the default rule suffices. If the two names are not identical, you must create rules to do the mapping.

13.2.2.1. Creating Rules

To accomodate more complex translations, you can create a hierarchical set of rules to add to the default. Each rule is divided into three parts: base, filter, and substitution.

13.2.2.1.1. The Base

The base begins with the number of components in the principal name (excluding the realm), followed by a colon, and the pattern for building the username from the sections of

the principal name. In the pattern section `$0` translates to the realm, `$1` translates to the first component and `$2` to the second component.

For example:

`[1:$1@$0]` translates `myusername@APACHE.ORG` to `myusername@APACHE.ORG`

`[2:$1]` translates `myusername/admin@APACHE.ORG` to `myusername`

`[2:$1%$2]` translates `myusername/admin@APACHE.ORG` to "`myusername%admin`

13.2.2.1.2. The Filter

The filter consists of a regex in a parentheses that must match the generated string for the rule to apply.

For example:

`(.*%admin)` matches any string that ends in `%admin`

`(.*@SOME.DOMAIN)` matches any string that ends in `@SOME.DOMAIN`

13.2.2.1.3. The Substitution

The substitution is a sed rule that translates a regex into a fixed string.

For example:

`s/@ACME\.COM//` removes the first instance of `@SOME.DOMAIN`.

`s/@[A-Z]*\.COM//` removes the first instance of `@` followed by a name followed by `COM`.

`s/X/Y/g` replaces all of the `X` in the name with `Y`

13.2.2.2. Examples

- If your default realm was `APACHE.ORG`, but you also wanted to take all principals from `ACME.COM` that had a single component `joe@ACME.COM`, you would create this rule:

```
RULE:[1:$1@$0](.@ACME.COM)s/@.//
DEFAULT
```

- To also translate names with a second component, you would use these rules:

```
RULE:[1:$1@$0](.@ACME.COM)s/@.//
RULE:[2:$1@$0](.@ACME.COM)s/@.//
DEFAULT
```

- To treat all principals from `APACHE.ORG` with the extension `/admin` as `admin`, your rules would look like this:

```
RULE[2:$1%$2@$0](.%admin@APACHE.ORG)s/./admin/
DEFAULT
```

13.2.3. Adding Security Information to Configuration Files

To enable security on HDP, you must add optional information to various configuration files.

13.2.3.1. core-site.xml

To the `core-site.xml` file on *every* host in your cluster, you must add the following information:

Table 13.3. core-site.xml

Property Name	Property Value	Description
hadoop.security.authentication	`kerberos`	Set the authentication type for the cluster. Valid values are: simple or kerberos.
hadoop.rpc.protection	`authentication; integrity; privacy`	This is an [OPTIONAL] setting. If not set, defaults to `authentication`. `authentication`= authentication only; the client and server mutually authenticate during connection setup. `integrity` = authentication and integrity; guarantees the integrity of data exchanged between client and server as well as authentication. `privacy` = authentication, integrity, and confidentiality; guarantees that data exchanged between client and server is encrypted and is not readable by a "man in the middle".
hadoop.security.authorization	`true`	Enable authorization for different protocols.

hadoop.security.auth_to_local	The mapping rules. For example
	`RULE:[2:$1@$0]([jt]t@.*EXAMPLE.COM)s/.*/$MAPRED_USER/`
	`RULE:[2:$1@$0]([nd]n@.*EXAMPLE.COM)s/.*/$HDFS_USER/`
	`RULE:[2:$1@$0](hm@.*EXAMPLE.COM)s/.*/$HBASE_USER/`
	`RULE:[2:$1@$0](rs@.*EXAMPLE.COM)s/.*/$HABSE_USER/`
	`DEFAULT`
		The mapping from Kerberos principal names to local OS user names. See Creating Mappings Between Principals and OS Service Usernames for more information.

hadoop.proxyuser.hive. groups	$HIVE_USER	Allows the Hive superuser to impersonate any member of the group users. This is required only when installing Hive on the cluster.
hadoop.proxyuser.hive. hosts	$HIVE_USER_Hostname_FQDN	The name of the host from which the Hive superuser can connect. This is required only when installing Hive on the cluster.
hadoop.proxyuser.oozie. groups	$OOZIE_USER	Allows the Oozie superuser to impersonate any member of the group users. This is required only when installing Oozie on the cluster.
hadoop.proxyuser.oozie. hosts	$OOZIE_USER_Hostname_FQDN	The name of the host from which the Oozie superuser can connect. This is

Property Name	Property Value	Description
		required only when installing Oozie on the cluster.
hadoop.proxyuser.HTTP. groups	users	Allows the HTTP superuser to impersonate any member of the group users.
hadoop.proxyuser.HTTP. hosts	WebHCat_Hostname_FQDN	The name of the host from which the HTTP superuser can connect.

The XML for these entries:

```
  <property>
        <name>hadoop.security.authentication</name>
        <value>kerberos</value>
        <description>Set the authentication for the cluster. Valid values are:
 simple or
        kerberos.
        </description>
</property>

<property>
        <name>hadoop.security.authorization</name>
        <value>true</value>
        <description>Enable authorization for different protocols.
        </description>
</property>

<property>

        <name>hadoop.security.auth_to_local</name>
        <value>
        RULE:[2:$1@$0]([jt]t@.*EXAMPLE.COM)s/.*/$MAPRED_USER/
        RULE:[2:$1@$0]([nd]n@.*EXAMPLE.COM)s/.*/$HDFS_USER/
        RULE:[2:$1@$0](hm@.*EXAMPLE.COM)s/.*/$HBASE_USER/
        RULE:[2:$1@$0](rs@.*EXAMPLE.COM)s/.*/$HBASE_USER/
        DEFAULT</value>
        <description>The mapping from kerberos principal names
        to local OS user names.</description>
</property>

<property>
  <name>hadoop.proxyuser.hive.groups</name>
  <value>users</value>
  <description>
     Allow the superuser hive to impersonate any members of the group users.
 This
     is required only when installing hive on the cluster.
  </description>
</property>

<property>
  <name>hadoop.proxyuser.hive.hosts</name>
  <value>Hive_Hostname_FQDN</value>
  <description>
     Hostname from where superuser hive can connect. This
     is required only when installing hive on the cluster.
  </description>
</property>
```

```
<property>
  <name>hadoop.proxyuser.oozie.groups</name>
 <value>users</value>
  <description>
     Allow the superuser oozie to impersonate any members of the group users.
 This
     is required only when installing oozie on the cluster.
 </description>
</property>

<property>
  <name>hadoop.proxyuser.oozie.hosts</name>
  <value>Oozie_Hostname_FQDN</value>
 <description>
     Hostname from where superuser oozie can connect. This
     is required only when installing oozie on the cluster.
  </description>
</property>

<property>
 <name>hadoop.proxyuser.hcat.hosts</name>
  <value>Webhcat_Hostname_FQDN</value>
  <description>
     Hostname from where superuser hcat can connect. This
     is required only when installing webhcat on the cluster.
  </description>
</property>

<property>
  <name>hadoop.proxyuser.HTTP.groups</name>
 <value>users</value>
 <description>
     Allow the superuser HTTP to impersonate any members of the group users.
  </description>
</property>

<property>
  <name>hadoop.proxyuser.HTTP.hosts</name>
  <value>Webhcat_Hostname_FQDN</value>
 <description>
     Hostname from where superuser HTTP can connect.
 </description>
</property>
```

13.2.3.2. hdfs-site.xml

To the `hdfs-site.xml` file on *every* host in your cluster, you must add the following information:

Table 13.4. hdfs-site.xml

Property Name	Property Value	Description
dfs.block.access.token.enable	`true`	If `true`, access tokens are used as capabilities for accessing datanodes. If `false`, no access tokens are checked on accessing datanodes.
dfs.namenode.kerberos.principal	nn/_HOST@EXAMPLE.COM	Kerberos principal name for the NameNode.

Property Name	Property Value	Description
dfs.secondary.namenode.kerberos.principal	nn/_HOST@EXAMPLE.COM	Kerberos principal name for the secondary NameNode.
dfs.secondary.http.address **Note**: cluster variant	Example: ip-10-72-235-178.ec2.internal:50090	Address of secondary namenode web server.
dfs.secondary.https.port	50490	The https port to which the secondary-namenode binds
dfs.web.authentication.kerberos.principal	HTTP/_HOST@EXAMPLE.COM	The HTTP Kerberos principal used by Hadoop-Auth in the HTTP endpoint. The HTTP Kerberos principal MUST start with 'HTTP/' per Kerberos HTTP SPNEGO specification.
dfs.web.authentication.kerberos.keytab	`/etc/security/keytabs/spnego.service.keytab`	The Kerberos keytab file with the credentials for the HTTP Kerberos principal used by Hadoop-Auth in the HTTP endpoint.
dfs.datanode.kerberos.principal	dn/_HOST@EXAMPLE.COM	The Kerberos principal that the DataNode runs as. "_HOST" is replaced by the real host name .
dfs.namenode.keytab.file	`/etc/security/keytabs/nn.service.keytab`	Combined keytab file containing the NameNode service and host principals.
dfs.secondary.namenode.keytab.file	`/etc/security/keytabs/nn.service.keytab`	Combined keytab file containing the NameNode service and host principals. <question?>
dfs.datanode.keytab.file	`/etc/security/keytabs/dn.service.keytab`	The filename of the keytab file for the DataNode.
dfs.https.port	50470	The https port to which the NameNode binds
dfs.https.address	Example: ip-10-111-59-170.ec2.internal:50470	The https address to which the NameNode binds
dfs.namenode.kerberos.internal.spnego.principal	${dfs.web.authentication.kerberos.principal}	
dfs.secondary.namenode.kerberos.internal.spnego.principal	${dfs.web.authentication.kerberos.principal}	
dfs.datanode.address	The address, with a privileged port - any port number under 1023. Example: 0.0.0.0:1019	
dfs.datanode.http.address	The address, with a privileged port - any port number under 1023. Example: 0.0.0.0:1022	

The XML for these entries:

```
<property>
        <name>dfs.block.access.token.enable</name>
        <value>true</value>
        <description> If "true", access tokens are used as capabilities
        for accessing datanodes. If "false", no access tokens are checked on
        accessing datanodes. </description>
</property>

<property>
        <name>dfs.namenode.kerberos.principal</name>
        <value>nn/_HOST@EXAMPLE.COM</value>
        <description> Kerberos principal name for the
        NameNode </description>
```

```
</property>

<property>
        <name>dfs.secondary.namenode.kerberos.principal</name>
        <value>nn/_HOST@EXAMPLE.COM</value>
        <description>Kerberos principal name for the secondary NameNode.
        </description>
</property>

<property>
        <!--cluster variant -->
        <name>dfs.secondary.http.address</name>
        <value>ip-10-72-235-178.ec2.internal:50090</value>
        <description>Address of secondary namenode web server</description>
</property>

<property>
        <name>dfs.secondary.https.port</name>
        <value>50490</value>
        <description>The https port where secondary-namenode
        binds</description>
</property>

<property>
        <name>dfs.web.authentication.kerberos.principal</name>
        <value>HTTP/_HOST@EXAMPLE.COM</value>
        <description> The HTTP Kerberos principal used by Hadoop-Auth in the
 HTTP endpoint.
        The HTTP Kerberos principal MUST start with 'HTTP/' per Kerberos HTTP
        SPNEGO specification.
        </description>
</property>

<property>
        <name>dfs.web.authentication.kerberos.keytab</name>
        <value>/etc/security/keytabs/spnego.service.keytab</value>
        <description>The Kerberos keytab file with the credentials for the
 HTTP
        Kerberos principal used by Hadoop-Auth in the HTTP endpoint.
        </description>
</property>

<property>
        <name>dfs.datanode.kerberos.principal</name>
        <value>dn/_HOST@EXAMPLE.COM</value>
        <description>
        The Kerberos principal that the DataNode runs as. "_HOST" is replaced
 by the real
        host name.
        </description>
</property>

<property>
        <name>dfs.namenode.keytab.file</name>
        <value>/etc/security/keytabs/nn.service.keytab</value>
        <description>
        Combined keytab file containing the namenode service and host
        principals.
        </description>
</property>
```

```
<property>
        <name>dfs.secondary.namenode.keytab.file</name>
        <value>/etc/security/keytabs/nn.service.keytab</value>
        <description>
        Combined keytab file containing the namenode service and host
        principals.
        </description>
</property>

<property>
        <name>dfs.datanode.keytab.file</name>
        <value>/etc/security/keytabs/dn.service.keytab</value>
        <description>
        The filename of the keytab file for the DataNode.
        </description>
</property>

<property>
        <name>dfs.https.port</name>
        <value>50470</value>
        <description>The https port where namenode
        binds</description>
</property>

<property>
        <name>dfs.https.address</name>
        <value>ip-10-111-59-170.ec2.internal:50470</value>
        <description>The https address where namenode binds</description>
</property>

<property>
        <name>dfs.namenode.kerberos.internal.spnego.principal</name>
        <value>${dfs.web.authentication.kerberos.principal}</value>
</property>

<property>
        <name>dfs.secondary.namenode.kerberos.internal.spnego.principal</name>

        <value>${dfs.web.authentication.kerberos.principal}</value>
</property>

<property>
        <name>dfs.datanode.address</name>
        <value>The address, with a privileged port - any port number under
 1023. Example: 0.0.0.0:1019</value>
</property>

<property>
        <name>def.datanode.http.address</name>
        <value>The address, with a privileged port - any port number under
 1023.  For example: 0.0.0.0:1022</value>
</property>
```

On all secure DataNodes, you must set the user to run the DataNode *as* after dropping privileges. For example:

```
export HADOOP_SECURE_DN_USER=$HDFS_USER
```

Note

The DataNode daemon must be started as `root`.

Optionally, you can allow that user to access the directories where pid and log files are kept. For example:

```
export HADOOP_SECURE_DN_PID_DIR=/var/run/hadoop/$HADOOP_SECURE_DN_USER
export HADOOP_SECURE_DN_LOG_DIR=/var/run/hadoop/$HADOOP_SECURE_DN_USER
```

13.2.3.3. mapred-site.xml

To the `mapred-site.xml` file on *every* host in your cluster, you must add the following information:

Table 13.5. mapred-site.xml

Property Name	Property Value	Description	Final
mapreduce.jobtracker.kerberos.principal	jt/_HOST@EXAMPLE.COM	Kerberos principal name for the JobTracker	
mapreduce.tasktracker.kerberos.principal	tt/_HOST@EXAMPLE.COM	Kerberos principal name for the TaskTracker. _HOST is replaced by the host name of the task tracker.	
mapreduce.jobtracker.keytab.file	`/etc/security/keytabs/jt.service.keytab`	The keytab for the JobTracker principal	
mapreduce.tasktracker.keytab.file	`/etc/security/keytabs/tt.service.keytab`	The keytab for the Tasktracker principal	
mapreduce.jobhistory.kerberos.principal **Note**: cluster variant	jt/_HOST@EXAMPLE.COM	Kerberos principal name for JobHistory. This must map to the same user as the JT user.	`true`
mapreduce.jobhistory.keytab.file **Note**: cluster variant	`/etc/security/keytabs/jt.service.keytab`	The keytab for the JobHistory principal	

The XML for these entries:

```
<property>
        <name>mapreduce.jobtracker.kerberos.principal</name>
        <value>jt/_HOST@EXAMPLE.COM</value>
        <description>Kerberos principal name for the JobTracker   </
description>
</property>

<property>
        <name>mapreduce.tasktracker.kerberos.principal</name>
        <value>tt/_HOST@EXAMPLE.COM</value>
        <description>Kerberos principal name for the TaskTracker.
        "_HOST" is replaced by the host name of the task tracker.
        </description>
</property>

<property>
```

```
        <name>mapreduce.jobtracker.keytab.file</name>
        <value>/etc/security/keytabs/jt.service.keytab</value>
        <description>
        The keytab for the jobtracker principal.
        </description>
</property>

<property>
        <name>mapreduce.tasktracker.keytab.file</name>
        <value>/etc/security/keytabs/tt.service.keytab</value>
        <description>The filename of the keytab for the task
        tracker</description>
</property>

<property>
        <name>mapreduce.jobhistory.kerberos.principal</name>
        <!--cluster variant -->
        <value>jt/_HOST@EXAMPLE.COM</value>
        <description> Kerberos principal name for JobHistory. This must map
 to
               the same user as the JT user. )</description>
</property>

<property>
        <name>mapreduce.jobhistory.keytab.file</name>
        <!--cluster variant -->
        <value>/etc/security/keytabs/jt.service.keytab</value>
        <description>The keytab for the JobHistory principal
        principal.</description>
</property>
```

13.2.3.4. hbase-site.xml

For Hbase to run on a secured cluster, Hbase must be able to authenticate itself to HDFS. To the `hbase-site.xml` file on your HBase server, you must add the following information. There are no default values; the following are all only examples:

Table 13.6. hbase-site.xml

Property Name	Property Value	Description
hbase.master.keytab.file	`/etc/security/keytabs/hm.service.keytab`	The keytab for the HMaster service principal
hbase.master.kerberos.principal	hm/_HOST@EXAMPLE.COM	The Kerberos principal name that should be used to run the HMaster process. If _H0ST is used as the hostname portion, it will be replaced with the actual hostname of the running instance.
hbase.regionserver.keytab.file	`/etc/security/keytabs/rs.service.keytab`	The keytab for the HRegionServer service principal
hbase.regionserver.kerberos.principal	rs/_HOST@EXAMPLE.COM	The Kerberos principal name that should be used to run the HRegionServer process. If _H0ST is used as the hostname portion, it will be replaced with the actual hostname of the running instance.
hbase.superuser	hbase	Comma-separated List of users or groups that are allowed full privileges, regardless of stored ACLs, across the

Property Name	Property Value	Description
		cluster. Only used when HBase security is enabled.

The XML for these entries:

```
  <property>
        <name>hbase.master.keytab.file</name>
        <value>/etc/security/keytabs/hm.service.keytab</value>
        <description>Full path to the kerberos keytab file to use for logging
        in the configured HMaster server principal.
        </description>
</property>

<property>
        <name>hbase.master.kerberos.principal</name>
        <value>hm/_HOST@EXAMPLE.COM</value>
        <description>Ex. "hbase/_HOST@EXAMPLE.COM".
        The kerberos principal name that
        should be used to run the HMaster process.  The
        principal name should be in
        the form: user/hostname@DOMAIN.  If "_HOST" is used
        as the hostname portion, it will be replaced with the actual hostname
 of the running
        instance.
        </description>
</property>

<property>
        <name>hbase.regionserver.keytab.file</name>
        <value>/etc/security/keytabs/rs.service.keytab</value>
        <description>Full path to the kerberos keytab file to use for logging
        in the configured HRegionServer server principal.
        </description>
</property>

<property>
        <name>hbase.regionserver.kerberos.principal</name>
        <value>rs/_HOST@EXAMPLE.COM</value>
        <description>Ex. "hbase/_HOST@EXAMPLE.COM".
        The kerberos principal name that
        should be used to run the HRegionServer process. The
        principal name should be in the form:
        user/hostname@DOMAIN.  If _HOST
        is used as the hostname portion, it will be replaced
        with the actual hostname of the running
        instance.  An entry for this principal must exist
        in the file specified in hbase.regionserver.keytab.file
        </description>
</property>

<!--Additional configuration specific to HBase security -->

<property>
        <name>hbase.superuser</name>
        <value>hbase</value>
        <description>List of users or groups (comma-separated), who are
        allowed full privileges, regardless of stored ACLs, across the
 cluster. Only
        used when HBase security is enabled.
```

```
        </description>
</property>
```

13.2.3.5. hive-site.xml

Hive Metastore supports Kerberos authentication for Thrift clients only. HiveServer does not support Kerberos authentication for any clients:

Table 13.7. hive-site.xml

Property Name	Property Value	Description
hive.metastore.sasl.enabled	`true`	If `true`, the Metastore Thrift interface will be secured with SASL and clients must authenticate with Kerberos
hive.metastore.kerberos.keytab.file	`/etc/security/keytabs/hive.service.keytab`	The keytab for the Metastore Thrift service principal
hive.metastore.kerberos.principal	hive/_HOST@EXAMPLE.COM	The service principal for the Metastore Thrift server. If _H0ST is used as the hostname portion, it will be replaced with the actual hostname of the running instance.
hive.server2.authentication	KERBEROS	Authentication type
hive.server2.authentication.kerberos.principal	hive/_HOST@EXAMPLE.COM	The service principal for the HiveServer2. If _HOST is used as the hostname portion, it will be replaced with the actual hostname of the running instance.
hive.server2.authentication.kerberos.keytab	`/etc/security/keytabs/hive.service.keytab`	The keytab for the HiveServer2 service principal

The XML for these entries:

```
<property>
        <name>hive.metastore.sasl.enabled</name>
        <value>true</value>
        <description>If true, the metastore thrift interface will be secured
 with
        SASL.
        Clients must authenticate with Kerberos.</description>
</property>

<property>
        <name>hive.metastore.kerberos.keytab.file</name>
        <value>/etc/security/keytabs/hive.service.keytab</value>
        <description>The path to the Kerberos Keytab file containing the
        metastore thrift server's service principal.</description>
</property>

<property>
        <name>hive.metastore.kerberos.principal</name>
        <value>hive/_HOST@EXAMPLE.COM</value>
        <description>The service principal for the metastore thrift server.
 The
        special string _HOST will be replaced automatically with the correct
        hostname.</description>
</property>

< property>
   <name>hive.server2.authentication</name>
```

```
    <value>KERBEROS</value>
    <description>Authentication type </description>
</property>

<property>
   <name>hive.server2.authentication.kerberos.principal</name>
    <value>hive/_HOST@EXAMPLE.COM</value>
    <description>The service principal for the HiveServer2. If _HOST is
           used as the hostname portion, it will be replaced with the actual
           hostname of the running instance.</description>
</property>

<property>
    <name>hive.server2.authentication.kerberos.keytab</name>
    <value>/etc/security/keytabs/hive.service.keytab</value>
    <description>The keytab for the HiveServer2 service principal</
description>
</property>
```

13.2.3.6. oozie-site.xml

To the `oozie-site.xml` file, you must add the following information:

Table 13.8. oozie-site.xml

Property Name	Property Value	Description
oozie.service.AuthorizationService.security.enabled	`true`	Specifies whether security (user name/admin role) is enabled or not. If it is disabled any user can manage the Oozie system and manage any job.
oozie.service.HadoopAccessorService.kerberos.enabled	`true`	Indicates if Oozie is configured to use Kerberos
local.realm	EXAMPLE.COM	Kerberos Realm used by Oozie and Hadoop. Using 'local.realm' to be aligned with Hadoop configuration.
oozie.service.HadoopAccessorService.keytab.file	`/etc/security/keytabs/oozie.service.keytab`	The keytab for the Oozie service principal.
oozie.service.HadoopAccessorService.kerberos.principal	$OOZIE_PRINCIPAL/_HOST@EXAMPLE.COM	Kerberos principal for Oozie service
oozie.authentication.type	kerberos	
oozie.authentication.kerberos.principal	HTTP/_HOST@EXAMPLE.COM	Whitelisted job tracker for Oozie service
oozie.authentication.kerberos.keytab	`/etc/security/keytabs/spnego.service.keytab`	Location of the Oozie user keytab file.
oozie.service.HadoopAccessorService.nameNode.whitelist		

oozie.authentication.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/`
	`RULE:[2:$1@$0](hm@.*EXAMPLE.COM)s/.*/$HBASE_USER/`
	`RULE:[2:$1@$0](rs@.*EXAMPLE.COM)s/.*/$HBASE_USER/`
	`DEFAULT`
		The mapping from Kerberos principal names to local service user names. See Creating Mappings Between Principals and UNIX Usernames for more information.

The XML for these entries:

```
<property>
    <name>oozie.service.AuthorizationService.security.enabled</name>
```

```
    <value>true</value>
    <description>Specifies whether security (user name/admin role) is enabled
 or not.
          If it is disabled any user can manage the Oozie system and manage
 any job.</description>
</property>

<property>
    <name>oozie.service.HadoopAccessorService.kerberos.enabled</name>
    <value>true</value>
    <description>Indicates if Oozie is configured to use Kerberos</
description>
</property>

<property>
    <name>local.realm </name>
    <value>EXAMPLE.COM </value>
    <description>Kerberos Realm used by Oozie and Hadoop. Using 'local.realm'
 to be
          aligned with Hadoop configuration</description>
</property>

<property>
    <name>oozie.service.HadoopAccessorService.keytab.file </name>
    <value>/etc/security/keytabs/oozie.service.keytab</value>
    <description>The keytab for the Oozie service principal.</description>
</property>

<property>
    <name>oozie.service.HadoopAccessorService.kerberos.principal</name>
    <value>$OOZIE_PRINCIPAL/_HOSTl@EXAMPLE.COM </value>
    <description>Kerberos principal for Oozie service</description>
</property>

<property>
    <name>oozie.authentication.type</name>
    <value>kerberos</value>
    <description>Authentication type</description>
</property>


<property>
    <name>oozie.authentication.kerberos.principal</name>
    <value>$HTTP_USER/_HOST@EXAMPLE.COM</value>
    <description>Whitelisted job tracker for Oozie service</description>
</property>

<property>
    <name> oozie.authentication.kerberos.keytab</name>
    <value>/etc/security/keytabs/spnego.service.keytab</value>
    <description>Location of the Oozie user keytab file.</description>
</property>

<property>
    <name>oozie.service.HadoopAccessorService.nameNode.whitelist</name>
    <value/>
    <description/>
</property>

<property>
```

```
    <name>oozie.authentication.kerberos.name.rules</name>
    <value><value>
        RULE:[2:$1@$0]([jt]t@.*EXAMPLE.COM)s/.*/$MAPRED_USER/
        RULE:[2:$1@$0]([nd]n@.*EXAMPLE.COM)s/.*/$HDFS_USER/
        RULE:[2:$1@$0](hm@.*EXAMPLE.COM)s/.*/$HBASE_USER/
        RULE:[2:$1@$0](rs@.*EXAMPLE.COM)s/.*/$HBASE_USER/
        DEFAULT</value>
    <description>The mapping from Kerberos principal names to local service
 user names.</description>
</property>
```

13.2.3.7. webhcat-site.xml

To the `webhcat-site.xml` file, you must add the following information:

Table 13.9. webhcat-site.xml

Property Name	Property Value	Description
templeton.kerberos.principal	HTTP/_HOST@EXAMPLE.COM	
templeton.kerberos.keytab	`/etc/security/keytabs/spnego.service.keytab`	
templeton.kerberos.secret	secret	
templeton.kerberos.properties	hive.metastore.local=false, hive.metastore.uris=thrift://MetastoreHost_FQDN:9083, hive.metastore.sasl.enabled=true, hive.metastore.execute.setugi= true, hive.exec.mode.local.auto=false, hive.metastore.kerberos.principal=$HIVE_PRINCIPAL/_HOST@EXAMPLE.COM"	

The XML for these entries:

```
</property>
    <name>templeton.kerberos.principal</name>
    <value>HTTP/_HOST@EXAMPLE.COM</value>
    <description/>
<property>

<property>
    <name>templeton.kerberos.keytab</name>
    <value>/etc/security/keytabs/spnego.service.keytab</value>
    <description/>
</property>

<property>
    <name>templeton.kerberos.secret</name>
    <value>secret</value>
    <description/>
</property>

<property>
    <name>templeton.kerberos.properties</name>
    <value>hive.metastore.local=false, hive.metastore.uris=thrift://
MetastoreHost_FQDN:9083,
         hive. metastore.sasl.enabled=true, hive.metastore.execute.setugi=
 true,
         hive.exec.mode.local.auto=false,
```

```
         hive.metastore.kerberos.principal=$HIVE_PRINCIPAL/_HOST@EXAMPLE.
COM"</value>
         <description/>
</property>
```