Hortonworks Data Platform

Upgrading HDP Manually

(September 2, 2016)

docs.cloudera.com

Hortonworks Data Platform: Upgrading 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. Upgrade from HDP 2.3 to HDP 2.4.3 Manually

Important

Upgrading from HDP 2.3.6 to 2.4.0 or 2.4.2 is not supported; you can only upgrade HDP 2.3.6 to 2.4.3+.

Important

If you installed and manage HDP 2.3 with Ambari, **you must use the Ambari Upgrade Guide** to perform the HDP 2.3 to HDP 2.4.3 upgrade.

Note

These instructions cover the upgrade between two minor releases. If you need to upgrade between two maintenance releases, follow the upgrade instructions in the HDP Release Notes.

Starting with HDP 2.2, HDP supports side-by-side installation of HDP 2.2 and subsequent releases, which lets you perform rolling upgrades on your cluster and improve execution times for in-place upgrade. To support side-by-side installation, the HDP package version naming convention for both RPMs and Debs has changed to include the HDP product version. For example, hadoop-hdfs is now hadoop-2.4.3-hdfs. HDP 2.2 marked the first release where HDP rpms, debs, and directories contained versions in the names to permit side-by-side installations of later HDP releases. To select from the releases you have installed side-by-side, Hortonworks provided hdp-select, a command that lets you select the active version of HDP from the versions you have installed.

Now with HDP 2.4.3, you can take advantage of this versioned installation to perform Rolling Upgrade from 2.3 to 2.4.3. However, Rolling Upgrade involves complex orchestration as well as side-by-side installation. It is too complex for a manual procedure, and is therefore supported only as an Ambari feature. If you wish to perform a Rolling Upgrade, refer to the Ambari Install instructions to install Ambari, then follow the Ambari Rolling Upgrade instructions, see Ambari Upgrade Guide.

This chapter provides instructions on how to manually upgrade to HDP 2.4.3 from the HDP 2.3 release. It assumes the existing HDP 2.3 was also installed manually.

The HDP packages for a complete installation of HDP 2.4.3 will occupy about 2.5 GB of disk space.

Warning

Until the upgrade is finalized, no HDFS data is deleted from the cluster. Be sure to review your capacity and ensure that you have extra space available during the upgrade window.

The following provides an overview of steps for upgrading to the latest release of HDP 2.4.3 from HDP 2.3:

Important

Upgrading from HDP 2.3.6 to 2.4.0 or 2.4.2 is not supported; you can only upgrade HDP 2.3.6 to 2.4.3+.

1. Get ready to upgrade
2. Upgrade HDP 2.3 Components
3. Use hdp-select to symlink the HDP 2.4.3 components into "current," in place of the former HDP 2.3 components
4. Configure and Start Apache ZooKeeper
5. Configure and Start Hadoop
6. Start HDFS
7. Configure and Start YARN/MapReduce
8. Configure and Start Apache HBase
9. Configure and Start Apache Phoenix
10. Configure and Start Apache Accumulo
11. Configure and Start Apache Tez
12. Configure and Start Apache Hive and Apache HCatalog
13. Configure and Start Apache Oozie
14. Configure and Start Apache WebHCat (Templeton)
15. Configure and Start Apache Pig
16. Configure and Start Apache Sqoop
17. Configure and Start Apache Flume
18. Configure and Start Apache Mahout
19. Configure and Start Hue
20. Configure and Start Apache Knox
21. Configure and Start Apache Falcon
22. Configure and Start Apache Storm
23. Configure and Start Apache Ranger
24. Configure and Start Apache Spark
25. Upgrade Apache Slider
26. Upgrade Apache Kafka

27. Upgrade Apache Atlas

28. Finalize Upgrade

1.1. Getting Ready to Upgrade

HDP Stack upgrade involves upgrading from HDP 2.3 to HDP 2.4.3 versions and adding the new HDP 2.4.3 services. These instructions change your configurations.

Note

In case if you are running a Kerberos secured installation you must use **kinit** before running the commands as any particular user.

Hardware recommendations

Although there is no single hardware requirement for installing HDP, there are some basic guidelines. The HDP packages for a complete installation of HDP 2.4.3 will take up about 2.5 GB of disk space.

The first step is to ensure you keep a backup copy of your HDP 2.3 configurations.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Important

Upgrading from HDP 2.3.6 to 2.4.0 or 2.4.2 is not supported; you can only upgrade HDP 2.3.6 to 2.4.3+.

1. Back up the HDP directories for all Hadoop components you have installed.

   The following is a list of all HDP directories:

   - `/etc/hadoop/conf`
   - `/etc/hbase/conf`
   - `/etc/hive-hcatalog/conf`
   - `/etc/hive-webhcat/conf`
   - `/etc/accumulo/conf`
   - `/etc/hive/conf`
   - `/etc/pig/conf`
   - `/etc/sqoop/conf`

- /etc/flume/conf
- /etc/mahout/conf
- /etc/oozie/conf
- /etc/hue/conf
- /etc/knox/conf
- /etc/zookeeper/conf
- /etc/tez/conf
- /etc/falcon/conf
- /etc/storm/conf
- /etc/slider/conf/
- /etc/spark/conf/
- /etc/phoenix/conf/
- /etc/ranger/admin/conf, /etc/ranger/usersync/conf (If Ranger is installed, also take a backup of install.properties for all the plugins, ranger admin & ranger usersync.)
- Optional - Back up your userlogs directories, ${mapred.local.dir}/userlogs.

2. Oozie runs a periodic purge on the shared library directory. The purge can delete libraries that are needed by jobs that started before the upgrade began and that finish after the upgrade. To minimize the chance of job failures, you should extend the oozie.service.ShareLibService.purge.interval and oozie.service.ShareLibService.temp.sharelib.retention.days settings.

   Add the following content to the the oozie-site.xml file prior to performing the upgrade:

```
<property>
<name>oozie.service.ShareLibService.purge.interval</name>
<value>1000</value><description>
How often, in days, Oozie should check for old ShareLibs and LauncherLibs to
 purge from HDFS.
</description>
</property>

<property>
<name>oozie.service.ShareLibService.temp.sharelib.retention.days</name>
<value>1000</value>
<description>
ShareLib retention time in days.</description>
</property>
```

3. Stop all long-running applications deployed using Slider.

First list the running applications managed by Slider:

```
su - yarn -c "/usr/hdp/current/slider-client/bin/slider list"
```

For all applications returned in previous command run:

```
su - yarn -c "/usr/hdp/current/slider-client/bin/slider stop <app_name>"
```

4. Stop all services (including MapReduce) except HDFS, ZooKeeper, and Ranger, and client applications deployed on HDFS on their respective machine. Some components (HDFS, ZooKeeper, etc) might reside in multiple machines. It is required to stop the running instances on all the machines.

   See Stopping HDP Services for more information.

Component	Command
Accumulo	`cd /usr/hdp/current/accumulo-client/bin && /usr/hdp/current/accumulo-client/bin/stop-all.sh`
Atlas	`/usr/hdp/current/atlas-server/bin/atlas_stop.py`
Knox	`cd $GATEWAY_HOME && su - knox -c "bin/gateway.sh stop"` Where $GATEWAY_HOME is the Knox installation directory, typically `/usr/hdp/current/knox-server`.
Falcon	`su - $FALCON_USER -c "/usr/hdp/current/falcon-server/bin/falcon-stop"` Where $FALCON_USER is the user owning the Falcon Services. For example: falcon.
Oozie	`su - oozie -c "/usr/hdp/current/oozie-server/bin/oozie-stop.sh"`
WebHCat	`su - $WEBHCAT_USER -c "/usr/hdp/hive-webhcat/sbin/webhcat_server.sh stop"` Where $WEBCAT_USER is the user owning the WebHCat services. For example: hcat.
Hive	Run this command on the Hive Metastore and Hive Server2 host machine: `pkill -u hive -f hive`
HBase RegionServers	`su - hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-daemon.sh --config /etc/hbase/conf stop regionserver"`
HBase Master host machine	`su - hbase -c "/usr/hdp/current/hbase-master/bin/hbase-daemon.sh --config /etc/hbase/conf stop master"`

YARN & Mapred Histro	Run this command on all NodeManagers:
	
	`su - yarn -c "/usr/hdp/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh --config /etc/hadoop/conf stop nodemanager"`
	
	Run this command on the History Server host machine:
	
	`su - mapred -c "/usr/hdp/current/hadoop-mapreduce-historyserver/sbin/`

Component	Command

	`mr-jobhistory-daemon.sh --config /etc/hadoop/conf stop historyserver"`
	
	Run this command on the ResourceManager host machine(s):
	
	`su - yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.sh --config /etc/hadoop/conf stop resourcemanager"`
	
	Run this command on the ResourceManager host machine:
	
	`su - yarn -c "/usr/hdp/current/hadoop-yarn-timelineserver/sbin/yarn-daemon.sh --config /etc/hadoop/conf stop timelineserver"`
	
	Run this command on the YARN Timeline Server node:
	
	`su - yarn -c "/usr/hdp/current/hadoop-yarn-timelineserver/sbin/yarn-daemon.sh --config /etc/hadoop/conf stop timelineserver"`

Storm	List the currently running topologies:
	
	`su - storm -c "storm list"`
	
	Kill all the topologies returned by the previous command:
	
	`su - storm -c "storm kill <topology_name>"`
	
	Stop Storm's supervisord:
	
	`sudo service supervisord stop`

Spark (History server)	`su - spark -c "/usr/hdp/current/spark-client/sbin/stop-history-server.sh"`

5. If you have the Hive component installed, back up the Hive Metastore database.

   The following instructions are provided for your convenience. For the latest backup instructions, see your database documentation.

Table 1.1. Hive Metastore Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sql For example: mysqldump hive > /tmp/mydir/backup_hive.sql	mysql $dbname < $inputfilename.sql For example: mysql hive < /tmp/mydir/backup_hive.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql For example: sudo -u postgres pg_dump hive > /tmp/mydir/backup_hive.sql	sudo -u $username psql $databasename < $inputfilename.sql For example: sudo -u postgres psql hive < /tmp/mydir/backup_hive.sql

Database Type	Backup	Restore
Oracle	Export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database file=input_file.dmp

6. If you have the Oozie component installed, back up the Oozie metastore database.

   These instructions are provided for your convenience. Check your database documentation for the latest backup instructions.

Table 1.2. Oozie Metastore Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sql For example: mysqldump oozie > /tmp/mydir/backup_oozie.sql	mysql $dbname < $inputfilename.sql For example: mysql oozie < /tmp/mydir/backup_oozie.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql For example: `sudo -u postgres pg_dump oozie > /tmp/mydir/backup_oozie.sql`	sudo -u $username psql $databasename < $inputfilename.sql For example: `sudo -u postgres psql oozie < /tmp/mydir/backup_oozie.sql`
Oracle	Export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database file=input_file.dmp

7. **Optional**: Back up the Hue database.

   The following instructions are provided for your convenience. For the latest backup instructions, please see your database documentation. For database types that are not listed below, follow your vendor-specific instructions.

Table 1.3. Hue Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sql For example: mysqldump hue > /tmp/mydir/backup_hue.sql	mysql $dbname < $inputfilename.sql For example: mysql hue < /tmp/mydir/backup_hue.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql For example: sudo -u postgres pg_dump hue > /tmp/mydir/backup_hue.sql	sudo -u $username psql $databasename < $inputfilename.sql For example: sudo -u postgres psql hue < /tmp/mydir/backup_hue.sql
Oracle	Connect to the Oracle database using sqlplus. Export the database.	Import the database:

Database Type	Backup	Restore
	For example: exp username/password@database full=yes file=output_file.dmp	For example: imp username/password@database file=input_file.dmp
SQLite	/etc/init.d/hue stop su $HUE_USER mkdir ~/hue_backup sqlite3 desktop.db .dump > ~/hue_backup/desktop.bak /etc/init.d/hue start	/etc/init.d/hue stop cd /var/lib/hue mv desktop.db desktop.db.old sqlite3 desktop.db < ~/hue_backup/desktop.bak /etc/init.d/hue start

8. Back up the Knox data/security directory.

```
cp -RL /etc/knox/data/security ~/knox_backup
```

9. Save the namespace by executing the following commands:

```
su - hdfs

hdfs dfsadmin -safemode enter

hdfs dfsadmin -saveNamespace
```

Note

In secure mode, you must have Kerberos credentials for the hdfs user.

10. Run the `fsck` command as the HDFS Service user and fix any errors. (The resulting file contains a complete block map of the file system.)

```
su - hdfs -c "hdfs fsck / -files -blocks -locations > dfs-old-fsck-1.log"
```

Note

In secure mode, you must have Kerberos credentials for the hdfs user.

11. Use the following instructions to compare status before and after the upgrade.

    The following commands must be executed by the user running the HDFS service (by default, the user is hdfs).

    a. Capture the complete namespace of the file system. (The following command does a recursive listing of the root file system.)

Important

Make sure the namenode is started.

```
su - hdfs -c "hdfs dfs -ls -R / > dfs-old-lsr-1.log"
```

Note

In secure mode you must have Kerberos credentials for the hdfs user.

b. Run the report command to create a list of DataNodes in the cluster.

```
su - hdfs -c "hdfs dfsadmin -report > dfs-old-report-l.log"
```

c. **Optional:** You can copy all or unrecoverable only data storelibext-customer directory in HDFS to a local file system or to a backup instance of HDFS.

d. **Optional:** You can also repeat the steps 3 (a) through 3 (c) and compare the results with the previous run to ensure the state of the file system remained unchanged.

12. Finalize any prior HDFS upgrade, if you have not done so already.

```
su - hdfs -c "hdfs dfsadmin -finalizeUpgrade"
```

Note

In secure mode, you must have Kerberos credentials for the hdfs user.

13. Stop remaining services (HDFS, ZooKeeper, and Ranger).

See Stopping HDP Services for more information.

Component	Command

HDFS	On all DataNotes:
	
	If you are running secure cluster, run following command as root:
	
	`/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop datanode`
	
	Else:
	
	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop datanode"`
	
	If you are not running a highly available HDFS cluster, stop the Secondary NameNode by executing this command on the Secondary NameNode host machine:
	
	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop secondarynamenode"`
	
	On the NameNode host machine(s)
	
	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop namenode"`
	
	If you are running NameNode HA, stop the ZooKeeper Failover Controllers (ZKFC) by executing this command on the NameNode host machine:

Component	Command

	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop zkfc"`
	
	If you are running NameNode HA, stop the JournalNodes by executing these command on the JournalNode host machines:
	
	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop journalnode"`

ZooKeeper Host machines	`su - zookeeper -c "/usr/hdp/current/zookeeper-server/bin/zookeeper-server stop"`

Ranger (XA Secure)	`service ranger-admin stop`
	
	`service ranger-usersync stop`

14. Back up your NameNode metadata.

Note

It's recommended to take a backup of the full `${dfs.namenode.name.dir}` path.

a. Copy the following checkpoint files into a backup directory.

The NameNode metadata is stored in a directory specified in the hdfs-site.xml configuration file under the configuration value "dfs.namenode.dir".

For example, if the configuration value is:

```
<property>
  <name>dfs.namenode.name.dir</name>
  <value>/hadoop/hdfs/namenode</value>
</property>
```

Then, the NameNode metadata files are all housed inside the directory `/hadooop.hdfs/namenode`.

b. Backup the layoutVersion of the namenode:

`${dfs.namenode.name.dir}/current/VERSION`

15. Verify that edit logs in `${dfs.namenode.name.dir}/current/edits*` are empty.

a. Run: `hdfs oev -i ${dfs.namenode.name.dir}/current/edits_inprogress_* -o edits.out`

b. Verify the edits.out file. It should only have OP_START_LOG_SEGMENT transaction. For example:

```
<?xml version="1.0" encoding="UTF-8"?><EDITS>
<EDITS_VERSION>-56</EDITS_VERSION>
<RECORD>
<OPCODE>OP_START_LOG_SEGMENT</OPCODE>
<DATA>
```

```
<TXID>5749</TXID>
</DATA>
</RECORD>
```

c. If edits.out has transactions other than OP_START_LOG_SEGMENT, run the following steps and then verify edit logs are empty.

- Start the existing version NameNode.
- Ensure there is a new FS image file.
- Shut the NameNode down:

```
hdfs dfsadmin –saveNamespace
```

16. Rename or delete any paths that are reserved in the new version of HDFS.

When upgrading to a new version of HDFS, it is necessary to rename or delete any paths that are reserved in the new version of HDFS. If the NameNode encounters a reserved path during upgrade, it will print an error such as the following:

```
/.reserved is a reserved path and .snapshot is a reserved path
component in this version of HDFS.

Please rollback and delete or rename this path, or upgrade with the
-renameReserved key-value pairs option to automatically rename these
paths during upgrade.
```

Specifying `-upgrade -renameReserved` optional key-value pairs causes the NameNode to automatically rename any reserved paths found during startup.

For example, to rename all paths named `.snapshot` to `.my-snapshot` and change paths named `.reserved` to `.my-reserved`, specify `-upgrade -renameReserved .snapshot=.my-snapshot,.reserved=.my-reserved`.

If no key-value pairs are specified with `-renameReserved`, the NameNode will then suffix reserved paths with:

```
.<LAYOUT-VERSION>.UPGRADE_RENAMED
```

For example: `.snapshot.-51.UPGRADE_RENAMED`.

Note

We recommend that you perform a `-saveNamespace` before renaming paths (running `-saveNamespace` appears in a previous step in this procedure). This is because a data inconsistency can result if an edit log operation refers to the destination of an automatically renamed file.

Also note that running `-renameReserved` will rename all applicable existing files in the cluster. This may impact cluster applications.

17. If you are on JDK 1.6, upgrade the JDK on all nodes to JDK 1.7 or JDK 1.8 before upgrading HDP.

1.2. Upgrade HDP 2.3 Components

Important

See the Release Notes for the HDP 2.4.3.0 repo information.

The upgrade process to HDP 2.4.3 involves the following steps.

Select your OS:

RHEL/CentOS/Oracle 6

1. On all hosts, clean the yum repository.

   ```
   yum clean all
   ```

2. Remove your old HDP 2.3 components. This command uninstalls the HDP 2.3 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for RHEL/CentOS/Oracle6 table for the names of the files that need to be removed for each component and use the following format:

   ```
   yum erase "accumulo_$VERSION_$BUILD*" "atlas_metadata_$VERSION_$BUILD*"
    "datafu_$VERSION_$BUILD*" "falcon_$VERSION_$BUILD*" "flume_$VERSION_
   $BUILD*" "hadoop_$VERSION_$BUILD*" "hadoop_$VERSION_$BUILD-client*" "hadoop_
   $VERSION_$BUILD-hdfs*" "hadooplzo_$VERSION_$BUILD*" ...
   ```

   Where:

   - $VERSION is the HDP version number in the following format: 2_4_3_0
   - $HUE_VERSION is the Hue version number in the following format : 2.6.1.2.4.3.0
   - $BUILD is the HDP build number in the following format : 258
   - $HDP_VERSION is the old installed HDP version (e.g.: 2.3.6.0-3796)

   So, for example, the file name will look like: accumulo-2_4_3_0_258*

Table 1.4. HDP Component Files for RHEL/CentOS/Oracle6

Component	Associated Files
Accumulo	accumulo_$VERSION_$BUILD*
Atlas	atlas_metadata_$VERSION_$BUILD*
Datafu	datafu_$VERSION_$BUILD*
Falcon	falcon_$VERSION_$BUILD*
Flume	flume_$VERSION_$BUILD*
Hadoop (includes hadoop_client and hadoop_hdfs)	hadoop_$VERSION_$BUILD* hadoop_$VERSION_$BUILD-client* hadoop_$VERSION_$BUILD-hdfs* hadooplzo_$VERSION_$BUILD*
HBase	hbase_$VERSION_$BUILD*

Component	Associated Files
Hive	hive_$VERSION_$BUILD* hive_hbase_$VERSION_$BUILD* hive_hcatalog_$VERSION_$BUILD* hive_jdbc_$VERSION_$BUILD* hive_metastore_$VERSION_$BUILD* hive_server_$VERSION_$BUILD* hive_webhcathive_hbase_$VERSION_$BUILD* ranger_hive_$VERSION_$BUILD*
Hue	hue-$HUE_VERSION-$BUILD* hue-beeswax-$HUE_VERSION-$BUILD* hue-common-$HUE_VERSION-$BUILD* hue-hcatalog-$HUE_VERSION-$BUILD* hue-oozie-$HUE_VERSION-$BUILD* hue-pig-$HUE_VERSION-$BUILD* hue-server-$HUE_VERSION-$BUILD*
Kafka	kafka_$VERSION_$BUILD*
Knox	knox_$VERSION_$BUILD*
Mahout	mahout_$VERSION_$BUILD*
MapReduce	hadoop_$VERSION_$BUILD-mapreduce*
Oozie	oozie_$VERSION_$BUILD*
Phoenix	phoenix_$VERSION_$BUILD
Pig	pig_$VERSION_$BUILD*
Ranger	ranger_$VERSION_$BUILD*
Slider	slider_$VERSION_$BUILD*
Spark	spark_$VERSION_$BUILD*
Sqoop	sqoop_$VERSION_$BUILD*
Storm	storm_$VERSION_$BUILD*
Tez	tez_$VERSION_$BUILD*
YARN	hadoop_$VERSION_$BUILD-yarn*
ZooKeeper	zookeeper_$VERSION_$BUILD*

3. Validate that all HDP 2.3 component binaries are uninstalled:

   ```
   yum list installed | grep @HDP2.3
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/yum.repos.d/hdp.repo
   ```

5. Install the HDP 2.4.3 repo:

   - Download the hdp.repo file:

     ```
     wget -nv http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.4.3.0/hdp.repo -O /etc/yum.repos.d/hdp.repo
     ```

- Confirm the HDP repository is configured.

  ```
  yum repolist
  ```

  You should see something like this. Verify that you have the HDP-2.4.3 directory:

  ```
  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 namestatus
  HDP-2.4.3 Hortonworks Data Platform Version - HDP-2.4.3
  ```

6. Install the HDP 2.4.3 versions of the components that you want to upgrade. Refer to the HDP Component Files for RHEL/CENTOS/Oracle6 table for the names of the files that need to be installed for each component and use the following format:

   ```
   yum install "accumulo_$VERSION_$BUILD*" "datafu_$VERSION_$BUILD*" "falcon_
   $VERSION_$BUILD*" "flume_$VERSION_$BUILD*" "hadoop_$VERSION_$BUILD*"
    "hadooplzo_$VERSION_$BUILD*" ...
   ```

   

   Note

   If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded.

   ```
   yum list installed | grep HDP-<old.stack.version.number>
   ```

   No component file names should appear in the returned list.

RHEL/CentOS/Oracle 5 (Deprecated)

1. On all hosts, clean the yum repository.

   ```
   yum clean all
   ```

2. Remove your old HDP 2.3 components. This command uninstalls the HDP 2.3 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for RHEL/CENTOS/Oracle5 table for the names of the files that need to be removed for each component and use the following format:

   ```
   yum erase "accumulo_$VERSION_$BUILD*" "datafu_$VERSION_$BUILD*" "falcon_
   $VERSION_$BUILD*" "flume_$VERSION_$BUILD*" "hadoop_$VERSION_$BUILD*"
    "hadooplzo_$VERSION_$BUILD*" ...
   ```

   Where:

   - $VERSION is the HDP version number in the following format: 2_4_3_0
   - $HUE_VERSION is the Hue version number in the following format : 2.6.1.2.4.3.0
   - $BUILD is the HDP build number in the following format : 258

So, for example, the file name will look like: accumulo-2_4_3_0_258*

Table 1.5. HDP Component Files for RHEL/CENTOS/Oracle5

Component	Associated Files
Accumulo	accumulo_$VERSION_$BUILD*
Atlas	atlas_metadata_$VERSION_$BUILD*
Datafu	datafu_$VERSION_$BUILD*
Falcon	falcon_$VERSION_$BUILD*
Flume	flume_$VERSION_$BUILD*
Hadoop (includes hadoop_client and hadoop_hdfs)	hadoop_$VERSION_$BUILD* hadoop_$VERSION_$BUILD-client* hadoop_$VERSION_$BUILD-hdfs* hadooplzo_$VERSION_$BUILD*
HBase	hbase_$VERSION_$BUILD*
Hive	hive_$VERSION_$BUILD* hive_hbase_$VERSION_$BUILD* hive_hcatalog_$VERSION_$BUILD* hive_jdbc_$VERSION_$BUILD* hive_metastore_$VERSION_$BUILD* hive_server_$VERSION_$BUILD* hive_webhcathive_hbase_$VERSION_$BUILD* ranger_hive_$VERSION_$BUILD*
Hue	hue-$HUE_VERSION-$BUILD* hue-beeswax-$HUE_VERSION-$BUILD* hue-common-$HUE_VERSION-$BUILD* hue-hcatalog-$HUE_VERSION-$BUILD* hue-oozie-$HUE_VERSION-$BUILD* hue-pig-$HUE_VERSION-$BUILD* hue-server-$HUE_VERSION-$BUILD*
Kafka	kafka_$VERSION_$BUILD*
Knox	knox_$VERSION_$BUILD*
Mahout	mahout_$VERSION_$BUILD*
MapReduce	hadoop_$VERSION_$BUILD-mapreduce*
Oozie	oozie_$VERSION_$BUILD*
Phoenix	phoenix_$VERSION_$BUILD
Pig	pig_$VERSION_$BUILD*
Ranger	ranger_$VERSION_$BUILD*
Slider	slider_$VERSION_$BUILD*
Spark	spark_$VERSION_$BUILD*
Sqoop	sqoop_$VERSION_$BUILD*

Component	Associated Files
Storm	storm_$VERSION_$BUILD*
Tez	tez_$VERSION_$BUILD*
YARN	hadoop_$VERSION_$BUILD-yarn*
ZooKeeper	zookeeper_$VERSION_$BUILD*

3. Validate that all HDP 2.3 component binaries are uninstalled:

   ```
   yum list installed | grep @HDP2.3
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/yum.repos.d/hdp.repo
   ```

5. Install the HDP 2.4.3 repo:

   - Download the hdp.repo file:

     ```
     wget -nv http://public-repo-1.hortonworks.com/HDP/centos5/2.x/updates/2.4.
     3.0/hdp.repo -O /etc/yum.repos.d/hdp.repo
     ```

   - Confirm the HDP repository is configured.

     ```
     yum repolist
     ```

     You should see something like this. Verify that you have the HDP-2.4.3 directory:

     ```
     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 namestatus
     HDP-2.4.3 Hortonworks Data Platform Version - HDP-2.4.3
     ```

6. Install the HDP 2.4.3 versions of the components that you want to upgrade. Refer to the HDP Component Files for RHEL/CENTOS/Oracle5table for the names of the files that need to be installed for each component and use the following format:

   ```
   yum install "accumulo_$VERSION_$BUILD*" "datafu_$VERSION_$BUILD*" "falcon_
   $VERSION_$BUILD*" "flume_$VERSION_$BUILD*" "hadoop_$VERSION_$BUILD*"
    "hadooplzo_$VERSION_$BUILD*" ...
   ```

   

   Note

   If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded.

   ```
   yum list installed | grep HDP-<old.stack.version.number>
   ```

   No component file names should appear in the returned list.

SLES 11 SP 1 (Deprecated)

1. On all hosts, clean the zypper repository.

   ```
   zypper clean --all
   ```

2. Remove your old HDP 2.3 components. These commands uninstall the HDP 2.3 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for SLES11 SP1 table for the names of the files that need to be removed for each component and use the following format:

   ```
   INSTALLED_PACKAGES=zypper search -s --installed-only | grep $OLD_HDP_VERSION
    | cut -d "|" -f2 | awk NF=NF RS= OFS=' '
   zypper --non-interactive remove ${INSTALLAD_PACKAGES}
   ```

   Where $OLD_HDP_VERSION is the old installed HDP version in the following format: 2.3.6.0-3796

   The following table lists the files that need to be deleted for each of the HDP components. Variables are used to indicate:

   - $VERSION - HDP version number in the following format: 2_4_3_0
   - $HUE_VERSION - Hue version number in the following format : 2.6.1.2.3.0.0
   - $BUILD - the HDP build number in the following format : 258

   So, for example, the file name will look like: accumulo-2_4_3_0_258*

Table 1.6. HDP Component Files for SLES11 SP1

Component	Associated Files
Accumulo	accumulo_$VERSION_$BUILD*
Atlas	atlas_metadata_$VERSION_$BUILD*
Datafu	datafu_$VERSION_$BUILD*
Falcon	falcon_$VERSION_$BUILD*
Flume	flume_$VERSION_$BUILD*
Hadoop (includes hadoop_client and hadoop_hdfs)	hadoop_$VERSION_$BUILD* hadoop_$VERSION_$BUILD-client* hadoop_$VERSION_$BUILD-hdfs* hadooplzo_$VERSION_$BUILD*
HBase	hbase_$VERSION_$BUILD*
Hive	hive_$VERSION_$BUILD* hive_hbase_$VERSION_$BUILD* hive_hcatalog_$VERSION_$BUILD* hive_jdbc_$VERSION_$BUILD* hive_metastore_$VERSION_$BUILD* hive_server_$VERSION_$BUILD* hive_webhcathive_hbase_$VERSION_$BUILD* ranger_hive_$VERSION_$BUILD*
Hue	hue-$HUE_VERSION-$BUILD*

Component	Associated Files
	hue-beeswax-$HUE_VERSION-$BUILD* hue-common-$HUE_VERSION-$BUILD* hue-hcatalog-$HUE_VERSION-$BUILD* hue-oozie-$HUE_VERSION-$BUILD* hue-pig-$HUE_VERSION-$BUILD* hue-server-$HUE_VERSION-$BUILD*
Kafka	kafka_$VERSION_$BUILD*
Knox	knox_$VERSION_$BUILD*
Mahout	mahout_$VERSION_$BUILD*
MapReduce	hadoop_$VERSION_$BUILD-mapreduce*
Oozie	oozie_$VERSION_$BUILD*
Phoenix	phoenix_$VERSION_$BUILD
Pig	pig_$VERSION_$BUILD*
Ranger	ranger_$VERSION_$BUILD*
Slider	slider_$VERSION_$BUILD*
Spark	spark_$VERSION_$BUILD*
Sqoop	sqoop_$VERSION_$BUILD*
Storm	storm_$VERSION_$BUILD*
Tez	tez_$VERSION_$BUILD*
YARN	hadoop_$VERSION_$BUILD-yarn*
ZooKeeper	zookeeper_$VERSION_$BUILD*

3. Validate that all HDP 2.3 component binaries are uninstalled:

```
zypper search --installed-only --repo HDP-2.3.6.0
```

4. Remove your old hdp.repo file:

```
rm /etc/zypp/repos.d/hdp.repo
```

5. Download the HDP 2.4.3 hdp.repo file:

```
wget -nv http://public-repo-1.hortonworks.com/HDP/sles11sp1/2.x/updates/2.4.
3.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
```

6. Install the HDP 2.4.3 versions of the components that you want to upgrade. You can install all components with the following commands

```
zypper update --repo HDP-2.4.3.0
UNINSTALLED_PACKAGES=zypper search --uninstalled-only --repo HDP-2.4.3.0 |
 awk -F '|' 'NR > 5 {print $2}' | tr '\n' ' '
zypper --non-interactive install ${UNINSTALLED_PACKAGES}
```

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

```
zypper up -r HDP-2.4.3
```

```
zypper install oozie-client
```

Note

If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded. For example, to verify hdfs, hive, and hcatalog:

   ```
   rpm -qa | grep hdfs, && rpm -qa | grep hive && rpm -qa | grep hcatalog
   ```

   No component files names should appear in the returned list.

SLES 11 SP3/SP4

1. On all hosts, clean the zypper repository.

   ```
   zypper clean --all
   ```

2. Remove your old HDP 2.3 components. These commands uninstall the HDP 2.3 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for SLES11 SP3/SP4 table for the names of the files that need to be removed for each component and use the following format:

   ```
   INSTALLED_PACKAGES=zypper search -s --installed-only | grep $OLD_HDP_VERSION
    | cut -d "|" -f2 | awk NF=NF RS= OFS=' '
   zypper --non-interactive remove ${INSTALLAD_PACKAGES}
   ```

   Where $OLD_HDP_VERSION is the old installed HDP version in the following format: 2.3.6.0-3796

   The following table lists the files that need to be deleted for each of the HDP components. Variables are used to indicate:

   - $VERSION - HDP version number in the following format: 2_4_3_0
   - $HUE_VERSION - Hue version number in the following format : 2.6.1.2.4.3.0
   - $BUILD - the HDP build number in the following format : 258

   So, the file name will look like: accumulo-2_4_3_0_258*

Table 1.7. HDP Component Files for SLES11 SP3/SP4

Component	Associated Files
Accumulo	accumulo_$VERSION_$BUILD*
Atlas	atlas_metadata_$VERSION_$BUILD*
Datafu	datafu_$VERSION_$BUILD*
Falcon	falcon_$VERSION_$BUILD*
Flume	flume_$VERSION_$BUILD*
Hadoop (includes hadoop_client and hadoop_hdfs)	hadoop_$VERSION_$BUILD* hadoop_$VERSION_$BUILD-client*

Component	Associated Files
	hadoop_$VERSION_$BUILD-hdfs* hadooplzo_$VERSION_$BUILD*
HBase	hbase_$VERSION_$BUILD*
Hive	hive_$VERSION_$BUILD* hive_hbase_$VERSION_$BUILD* hive_hcatalog_$VERSION_$BUILD* hive_jdbc_$VERSION_$BUILD* hive_metastore_$VERSION_$BUILD* hive_server_$VERSION_$BUILD* hive_webhcathive_hbase_$VERSION_$BUILD* ranger_hive_$VERSION_$BUILD*
Hue	hue-$HUE_VERSION-$BUILD* hue-beeswax-$HUE_VERSION-$BUILD* hue-common-$HUE_VERSION-$BUILD* hue-hcatalog-$HUE_VERSION-$BUILD* hue-oozie-$HUE_VERSION-$BUILD* hue-pig-$HUE_VERSION-$BUILD* hue-server-$HUE_VERSION-$BUILD*
Kafka	kafka_$VERSION_$BUILD*
Knox	knox_$VERSION_$BUILD*
Mahout	mahout_$VERSION_$BUILD*
MapReduce	hadoop_$VERSION_$BUILD-mapreduce*
Oozie	oozie_$VERSION_$BUILD*
Phoenix	phoenix_$VERSION_$BUILD
Pig	ranger_$VERSION_$BUILD*
Ranger	ranger_$VERSION_$BUILD*
Slider	slider_$VERSION_$BUILD*
Spark	spark_$VERSION_$BUILD*
Sqoop	sqoop_$VERSION_$BUILD*
Storm	storm_$VERSION_$BUILD*
Tez	tez_$VERSION_$BUILD*
YARN	hadoop_$VERSION_$BUILD-yarn*
ZooKeeper	zookeeper_$VERSION_$BUILD*

3. Validate that all HDP 2.3 component binaries are uninstalled:

   ```
   zypper search --installed-only --repo HDP-2.3.6.0
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/zypp/repos.d/hdp.repo
   ```

5. Download the HDP 2.4.3 hdp.repo file:

```
http://public-repo-1.hortonworks.com/HDP/suse11sp3/2.x/
updates/2.4.3.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
```

6. Install the HDP 2.4.3 versions of the components that you want to upgrade. You can install all components with the following commands:

```
zypper update --repo HDP-2.4.3.0
UNINSTALLED_PACKAGES=zypper search --uninstalled-only --repo HDP-2.4.3.0 |
 awk -F '|' 'NR > 5 {print $2}' | tr '\n' ' '
zypper --non-interactive install ${UNINSTALLED_PACKAGES}
```

```
zypper up -r HDP-2.4.3
```

```
zypper install oozie-client
```

Note

If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded. For example, to verify hdfs, hive, and hcatalog:

```
rpm -qa | grep hdfs, && rpm -qa | grep hive && rpm -qa | grep
hcatalog
```

No component files names should appear in the returned list.

Ubuntu 12

1. On all hosts, clean the apt-get repository.

```
apt-get clean -&-all
```

2. Remove your old HDP 2.3 components. This command uninstalls the HDP 2.3 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for Ubuntu 12 table for the names of the files that need to be removed for each component and use the following format:

```
apt-get remove "accumulo-$VERSION-$BUILD*" "datafu-$VERSION-$BUILD*"
 "falcon-$VERSION-$BUILD*" "flume-$VERSION-$BUILD*" "hadoop-$VERSION-
$BUILD*" "hadooplzo-$VERSION-$BUILD*" ...
```

The following table lists the files that need to be deleted for each of the HDP components. Variables are used to indicate:

- $VERSION - HDP version in the following format: 2-4-2-0
- $HUE_VERSION - Hue version in the following format : 2.6.1.2.4.3.0
- $BUILD - version of the HDP build in the following format : 258

So, for example, the file name will look like: accumulo-2-4-2-0-258*

Table 1.8. HDP Component Files for Ubuntu 12

Component	HDP Files to Re-install for Upgrade
Accumulo	accumulo-$VERSION-$BUILD*
Atlas	atlas-metadata-$VERSION-$BUILD*
Datafu	datafu-$VERSION-$BUILD*
Falcon	falcon-$VERSION-$BUILD*
Flume	flume-$VERSION-$BUILD*
Hadoop (includes hadoop-client and hadoop-hdfs)	hadoop-$VERSION-$BUILD* hadoop-$VERSION-$BUILD-client* hadoop-$VERSION-$BUILD-hdfs* hadooplzo-$VERSION-$BUILD*
HBase	hbase-$VERSION-$BUILD*
Hive	hive-$VERSION-$BUILD* hive-hbase-$VERSION-$BUILD* hive-hcatalog-$VERSION-$BUILD* hive-jdbc-$VERSION-$BUILD* hive-metastore-$VERSION-$BUILD* hive-server-$VERSION-$BUILD* hive-webhcathive-hbase-$VERSION-$BUILD* ranger-hive-$VERSION-$BUILD*
Hue	hue-$HUE_VERSION-$BUILD* hue-beeswax-$HUE_VERSION-$BUILD* hue-common-$HUE_VERSION-$BUILD* hue-hcatalog-$HUE_VERSION-$BUILD* hue-oozie-$HUE_VERSION-$BUILD* hue-pig-$HUE_VERSION-$BUILD* hue-server-$HUE_VERSION-$BUILD*
Kafka	kafka-$VERSION-$BUILD*
Knox	knox-$VERSION-$BUILD*
Mahout	mahout-$VERSION-$BUILD*
MapReduce	hadoop-$VERSION-$BUILD-mapreduce*
Oozie	oozie-$VERSION-$BUILD*
Phoenix	phoenix-$VERSION-$BUILD
Pig	ranger-$VERSION-$BUILD*
Ranger	ranger-$VERSION-$BUILD*
Slider	slider-$VERSION-$BUILD*
Spark	spark-$VERSION-$BUILD*
Sqoop	sqoop-$VERSION-$BUILD*
Storm	storm-$VERSION-$BUILD*
Tez	tez-$VERSION-$BUILD*

Component	HDP Files to Re-install for Upgrade
YARN	hadoop-$VERSION-$BUILD-yarn*
ZooKeeper	zookeeper-$VERSION-$BUILD*

3. Validate that all HDP 2.3 component binaries are uninstalled:

```
yum list installed | grep @HDP2.3
```

4. Remove your old hdp.repo file:

```
rm /etc/apt/sources.list.d/hdp.list
```

5. Download the HDP 2.4.3 hdp.repo file:

```
wget http://public-repo-1.hortonworks.com/HDP/ubuntu12/2.x/
updates/2.4.3.0/hdp.list - O /etc/apt/sources.list.d/hdp.list
```

6. Run an update:

```
apt-get update
```

7. Install the HDP 2.4.3 versions of the components that you want to upgrade. Refer to the HDP Component Files for Ubuntu 12 table for the names of the files that need to be installed for each component and use the following format:

```
apt-get install "accumulo-$VERSION-$BUILD*" "datafu-$VERSION-$BUILD*"
 "falcon-$VERSION-$BUILD*" "flume-$VERSION-$BUILD*" "hadoop-$VERSION-
$BUILD*" "hadooplzo-$VERSION-$BUILD*" ...
```

Note

If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

Debian 6 (Deprecated)/Debian 7

HDP support for Debian 6 is deprecated with HDP 2.4.3. Future versions of HDP will no longer be supported on Debian 6.

1. On all hosts, clean the apt-get repository.

```
apt-get clean -&-all
```

2. Remove your old HDP 2.3 components. This command uninstalls the HDP 2.3 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for Debian 6 (Deprecated)/Debian 7 table for the names of the files that need to be removed for each component and use the following format:

```
apt-get remove "accumulo-$VERSION-$BUILD*" "datafu-$VERSION-$BUILD*"
 "falcon-$VERSION-$BUILD*" "flume-$VERSION-$BUILD*" "hadoop-$VERSION-
$BUILD*" "hadooplzo-$VERSION-$BUILD*" ...
```

   The following table lists the files that need to be deleted for each of the HDP components. Variables are used to indicate:

- $VERSION - HDP version in the following format: 2-4-2-0
- $HUE_VERSION - Hue version in the following format : 2.6.1.2.3.0.0
- $BUILD - version of the HDP build in the following format : 258

So, for example, the file name will look like: accumulo-2-4-2-0-258*

Table 1.9. HDP Component Files for Debian 6 (Deprecated)/Debian 7

Component	HDP Files to Re-install for Upgrade
Accumulo	accumulo-$VERSION-$BUILD*
Atlas	atlas-metadata-$VERSION-$BUILD*
Datafu	datafu-$VERSION-$BUILD*
Falcon	falcon-$VERSION-$BUILD*
Flume	flume-$VERSION-$BUILD*
Hadoop (includes hadoop-client and hadoop-hdfs)	hadoop-$VERSION-$BUILD* hadoop-$VERSION-$BUILD-client* hadoop-$VERSION-$BUILD-hdfs* hadooplzo-$VERSION-$BUILD*
HBase	hbase-$VERSION-$BUILD*
Hive	hive-$VERSION-$BUILD* hive-hbase-$VERSION-$BUILD* hive-hcatalog-$VERSION-$BUILD* hive-jdbc-$VERSION-$BUILD* hive-metastore-$VERSION-$BUILD* hive-server-$VERSION-$BUILD* hive-webhcathive-hbase-$VERSION-$BUILD* ranger-hive-$VERSION-$BUILD*
Hue	hue-$HUE_VERSION-$BUILD* hue-beeswax-$HUE_VERSION-$BUILD* hue-common-$HUE_VERSION-$BUILD* hue-hcatalog-$HUE_VERSION-$BUILD* hue-oozie-$HUE_VERSION-$BUILD* hue-pig-$HUE_VERSION-$BUILD* hue-server-$HUE_VERSION-$BUILD*
Kafka	kafka-$VERSION-$BUILD*
Knox	knox-$VERSION-$BUILD*
Mahout	mahout-$VERSION-$BUILD*
MapReduce	hadoop-$VERSION-$BUILD-mapreduce*
Oozie	oozie-$VERSION-$BUILD*
Phoenix	phoenix-$VERSION-$BUILD
Pig	ranger-$VERSION-$BUILD*

Component	HDP Files to Re-install for Upgrade
Ranger	ranger-$VERSION-$BUILD*
Slider	slider-$VERSION-$BUILD*
Spark	spark-$VERSION-$BUILD*
Sqoop	sqoop-$VERSION-$BUILD*
Storm	storm-$VERSION-$BUILD*
Tez	tez-$VERSION-$BUILD*
YARN	hadoop-$VERSION-$BUILD-yarn*
ZooKeeper	zookeeper-$VERSION-$BUILD*

3. Validate that all HDP 2.3 component binaries are uninstalled:

   ```
   yum list installed | grep @HDP2.3
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/apt/sources.list.d/hdp.list
   ```

5. Download the HDP 2.4.3 hdp.repo file:

   ```
   wget -nv http://public-repo-1.hortonworks.com/HDP/debian6/2.x/
   updates/2.4.3.0/hdp.list - O /etc/apt/sources.list.d/hdp.list
   ```

   or

   ```
   wget -nv http://public-repo-1.hortonworks.com/HDP/debian7/2.x/
   updates/2.4.3.0/hdp.list - O /etc/apt/sources.list.d/hdp.list
   ```

6. Run an update:

   ```
   apt-get update
   ```

7. Install the HDP 2.4.3 versions of the components that you want to upgrade. Refer to the HDP Component Files for Debian 6 (Deprecated)/Debian 7 table for the names of the files that need to be installed for each component and use the following format:

   ```
   apt-get install "accumulo-$VERSION-$BUILD*" "datafu-$VERSION-$BUILD*"
    "falcon-$VERSION-$BUILD*" "flume-$VERSION-$BUILD*" "hadoop-$VERSION-
   $BUILD*" "hadooplzo-$VERSION-$BUILD*" ...
   ```

Note

If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

1.3. Symlink Directories with hdp-select

Warning

HDP 2.4.3 installs hdp-select automatically with the installation or upgrade of the first HDP component. If you have not already upgraded ZooKeeper, hdp-select has not been installed.

To prevent version-specific directory issues for your scripts and updates, Hortonworks provides hdp-select, a script that symlinks directories to hdp-current and modifies paths for configuration directories.

1. Before you run hdp-select, remove one link:

   ```
   rm /usr/bin/oozie
   ```

2. Run hdp-select set all on your NameNode and all your DataNodes:

   ```
   hdp-select set all 2.4.3.0-$BUILD
   ```

   For example:

   ```
   /usr/bin/hdp-select set all 2.4.3.0-$BUILD
   ```

1.4. Configure and Start Apache ZooKeeper

Tip

If you are running a highly available HDFS cluster, configure and restart Apache ZooKeeper **before** you upgrade HDFS. This best practice lets the upgraded ZKFC work with your primary NameNode and your Standby NameNode.

Before you can upgrade the ZooKeeper service, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

1. Replace your configuration after upgrading on all the ZooKeeper nodes. Replace the ZooKeeper template configuration in `/etc/zookeeper/conf`.

2. Start ZooKeeper.

   On all ZooKeeper server host machines, run the following command to start ZooKeeper and the ZKFC:

   ```
   su - zookeeper -c "/usr/hdp/current/zookeeper-server/bin/zookeeper-server
    start"
   ```

1.5. Configure Hadoop

Overwrite original configuration files with new ones:

```
cp -r $YOUR_BACKUP_FOLDER/ /etc/hadoop/conf
```

Where $YOUR_BACKUP_FOLDER is the directory where you saved your configuration files.

RHEL/CentOS/Oracle Linux

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml.

2. Paths have changed in HDP 2.3. Make sure you remove old path specifications from hadoop-env.sh, such as:

   ```
   export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-amd64-64
   ```

   If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

SLES

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml.

2. Paths have changed since HDP 2.3. Make sure you remove old path specifications from hadoop-env.sh, such as:

   ```
   export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-amd64-64
   ```

   If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

Ubuntu/Debian

HDP support for Debian 6 is deprecated with HDP 2.4.3. Future versions of HDP will no longer be supported on Debian 6.

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml

2. Paths have changed in HDP 2.4.3. Make sure you remove old path specifications from hadoop-env.sh, such as:

   ```
   export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-amd64-64
   ```

   If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

1.6. Start Hadoop Core

Warning

Before you start HDFS on a highly available HDFS cluster, you must start the ZooKeeper service. If you do not start the ZKFC, there can be failures.

To start HDFS, run commands as the $HDFS_USER.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using

another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

1. Replace your configuration after upgrading on all the HDFS nodes. Replace the HDFS template configuration in `/etc/hdfs/conf`.

2. If you are upgrading from a highly available HDFS cluster configuration, start all JournalNodes. On each JournalNode host, run the following commands:

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

   

   Important

   All JournalNodes must be running when performing the upgrade, rollback, or finalization operations. If any JournalNodes are down when running any such operation, the operation fails.

3. If you are running HDFS on a highly available namenode, you must first start the ZooKeeper service

   

   Note

   Perform this step only if you are on a highly available HDFS cluster.

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

4. Start the NameNode.

   Because the file system version has now changed you must start the NameNode manually.

   On the active NameNode host, run the following commands:

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

   On a large system, this can take a long time to complete.

   

   Note

   Run this command with the -upgrade option only once. After you have completed this step, you can bring up the NameNode using this command without including the -upgrade option.

   To check if the Upgrade is in progress, check that the "\previous" directory has been created in the \NameNode and \JournalNode directories. The "\previous" directory contains a snapshot of the data before upgrade.

   In a highly available HDFS cluster configuration, this NameNode will not enter the standby state as usual. Rather, this NameNode will immediately enter the active state, perform an upgrade of its local storage directories, and also perform an upgrade of the

shared edit log. At this point, the standby NameNode in the HA pair is still down. It will be out of sync with the upgraded active NameNode.

To synchronize the active and standby NameNode, re-establishing HA, re-bootstrap the standby NameNode by running the NameNode with the '-bootstrapStandby' flag. Do NOT start this standby NameNode with the '-upgrade' flag.

```
su - hdfs -c "hdfs namenode -bootstrapStandby -force"
```

The bootstrapStandby command will download the most recent fsimage from the active NameNode into the $dfs.name.dir directory of the standby NameNode. You can enter that directory to make sure the fsimage has been successfully downloaded. After verifying, start the ZKFailoverController, then start the standby NameNode. You can check the status of both NameNodes using the Web UI.

5. Verify that the NameNode is up and running:

   ```
   ps -ef|grep -i NameNode
   ```

6. If you do not have a highly available HDFS cluster configuration (non_HA namenode), start the Secondary NameNode.

   

   Note

   Do not perform this step if you have a highly available HDFS cluster configuration.

   On the Secondary NameNode host machine, run the following commands:

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

7. Verify that the Secondary NameNode is up and running.

   

   Note

   Do not perform this step if you have a highly available HDFS cluster environment.

   ```
   ps -ef|grep SecondaryNameNode
   ```

8. Start DataNodes.

   On each of the DataNodes, enter the following command. Note: If you are working on a `non-secure` DataNode, use $HDFS_USER. For a secure DataNode, use root.

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

9. Verify that the DataNode process is up and running:

   ```
   ps -ef|grep DataNode
   ```

10. Verify that NameNode can go out of safe mode.

```
>su - hdfs -c "hdfs dfsadmin -safemode wait"
```

You should see the following result: `Safe mode is OFF`

In general, it takes 5-10 minutes to get out of safemode. For thousands of nodes with millions of data blocks, getting out of safemode can take up to 45 minutes.

1.7. Verify HDFS Filesystem Health

Analyze if the filesystem is healthy.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Important

If you have a secure server, you will need Kerberos credentials for hdfs user access.

1. Run the fsck command on namenode as $HDFS_USER:

   ```
   su - hdfs -c "hdfs fsck / -files -blocks -locations > dfs-new-fsck-1.log"
   ```

   You should see feedback that the filesystem under path / is HEALTHY.

2. Run `hdfs` namespace and report.

   a. List directories.

   ```
   su - hdfs -c "hdfs dfs -ls -R / > dfs-new-lsr-1.log"
   ```

   b. Open the `dfs-new-lsr-l.log` and confirm that you can see the file and directory listing in the namespace.

   c. Run report command to create a list of DataNodes in the cluster.

   ```
   su - hdfs -c "hdfs dfsadmin -report > dfs-new-report-1.log"
   ```

   d. Open the `dfs-new-report` file and validate the admin report.

3. Compare the namespace report before the upgrade and after the upgrade. Verify that user files exist after upgrade.

   The file names are listed below:

   ```
   dfs-old-fsck-1.log < -- > dfs-new-fsck-1.log
   ```

   ```
   dfs-old-lsr-1.log < -- > dfs-new-lsr-1.log
   ```

Note

You must do this comparison manually to catch all errors.

4. From the NameNode WebUI, determine if all DataNodes are up and running.

   ```
   http://<namenode>:<namenodeport
   ```

5. If you are on a highly available HDFS cluster, go to the StandbyNameNode web UI to see if all DataNodes are up and running:

   ```
   http://<standbynamenode>:<namenodeport>
   ```

6. If you are **not** on a highly available HDFS cluster, go to the SecondaryNameNode web UI to see if it the secondary node is up and running:

   ```
   http://<secondarynamenode>:<secondarynamenodeport>
   ```

7. Verify that read and write to hdfs works successfully.

   ```
   hdfs dfs -put [input file] [output file]

   hdfs dfs -cat [output file]
   ```

1.8. Configure YARN and MapReduce

After you upgrade Hadoop, complete the following steps to update your configs.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Important

In secure mode, you must have Kerberos credentials for the hdfs user.

1. Upload the MapReduce tarball to HDFS. As the HDFS user, for example 'hdfs':

   ```
   su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/
   mapreduce/"

   su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/hadoop/
   mapreduce.tar.gz /hdp/apps/2.4.3.0-$BUILD/mapreduce/"

   su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp"

   su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/
   mapreduce"
   ```

```
su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/
mapreduce/mapreduce.tar.gz"
```

2. Make sure that the following properties are in `/etc/hadoop/conf/mapred-site.xml`:

   - Make sure mapreduce.application.framework.path exists in mapred-site.xml:

```
<property>
 <name>mapreduce.application.framework.path</name>
 <value>/hdp/apps/${hdp.version}/mapreduce/mapreduce.tar.gz#mr-framework</
value>
</property>

<property>
 <name>yarn.app.mapreduce.am.admin-comand-opts</name>
 <value>-Dhdp.version=${hdp.version}</value>
</property>
```

   Note

   You do not need to modify ${hdp.version}.

   - Modify the following existing properties to include ${hdp.version}:

```
<property>
 <name>mapreduce.admin.user.env</name>
 <value>LD_LIBRARY_PATH=/usr/hdp/${hdp.version}/hadoop/lib/native:/usr/
hdp/${hdp.version}/hadoop/
     lib/native/Linux-amd64-64</value>
</property>

<property>
 <name>mapreduce.admin.map.child.java.opts</name>
 <value>-server -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.
version}</value>
 <final>true</final>
</property>

<property>
 <name>mapreduce.admin.reduce.child.java.opts</name>
 <value>-server -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.
version}</value>
 <final>true</final>
</property>

<property>
 <name>mapreduce.application.classpath</name>
 <value>$PWD/mr-framework/hadoop/share/hadoop/mapreduce/*,
   $PWD/mr-framework/hadoop/share/hadoop/mapreduce/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/common/*,
   $PWD/mr-framework/hadoop/share/hadoop/common/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/yarn/*,
   $PWD/mr-framework/hadoop/share/hadoop/yarn/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/hdfs/*,
   $PWD/mr-framework/hadoop/share/hadoop/hdfs/lib/*,
  /usr/hdp/${hdp.version}/hadoop/lib/hadoop-lzo-0.6.0.${hdp.version}.jar,
  /etc/hadoop/conf/secure</value>
```

```
</property>
```

Note

You do not need to modify ${hdp.version}.

Note

If you are planning to use Spark in yarn-client mode, make Spark work in yarn-client mode 2.4.3.0-$BUILD.

3. Make sure the following property is in `/etc/hadoop/conf/yarn-site.xml`:

```
<property>
 <name>yarn.application.classpath</name>
 <value>$HADOOP_CONF_DIR,/usr/hdp/${hdp.version}/hadoop-client/*,
  /usr/hdp/${hdp.version}/hadoop-client/lib/*,
  /usr/hdp/${hdp.version}/hadoop-hdfs-client/*,
  /usr/hdp/${hdp.version}/hadoop-hdfs-client/lib/*,
  /usr/hdp/${hdp.version}/hadoop-yarn-client/*,
  /usr/hdp/${hdp.version}/hadoop-yarn-client/lib/*</value>
</property>
```

4. On secure clusters only, add the following properties to `/etc/hadoop/conf/yarn-site.xml`:

```
 <property>
  <name>yarn.timeline-service.recovery.enabled:</name>
  <value>TRUE</value>
  </property>

  <property>
  <name>yarn.timeline-service.state-store.class: org.apache.hadoop.yarn.
server.timeline.recovery:</name>
  <value>LeveldbTimelineStateStore</value>
  </property>

  <property>
  <name>yarn.timeline-service.leveldb-state-store.path:</name>
  <value><the same as the default of "yarn.timeline-service-leveldb-
timeline-store.path</value>
  </property>
```

5. For secure clusters, you must create and configure the container-executor.cfg configuration file:

   - Create the container-executor.cfg file in /usr/hdp/2.4.3.0-$BUILD/hadoop-yarn/bin/container-executor.

   - Insert the following properties:

```
<property>
yarn.nodemanager.linux-container-executor.group=hadoop
banned.users=hdfs,yarn,mapred
min.user.id=1000
</property>
```

- yarn.nodemanager.linux-container-executor.group - Configured value of yarn.nodemanager.linux-container-executor.group. This must match the value of yarn.nodemanager.linux-container-executor.group in yarn-site.xml.
- banned.users - Comma-separated list of users who can not run container-executor.
- min.user.id - Minimum value of user id. This prevents system users from running container-executor.
- allowed.system.users - Comma-separated list of allowed system users.

- Set the file /etc/hadoop/conf/container-executor.cfg file permissions to only be readable by root:

```
chown root:hadoop /etc/hadoop/conf/container-executor.cfg
chmod 400 /etc/hadoop/conf/container-executor.cfg
```

- Set the container-executor program so that only root or hadoop group users can run it:

```
chown root:hadoop /usr/hdp/${hdp.version}/hadoop-yarn-server-nodemanager/
bin/container-executor

chmod 6050 /usr/hdp/${hdp.version}/hadoop-yarn-server-nodemanager/bin/
container-executor
```

1.9. Start YARN/MapReduce Services

To start YARN, run commands as a YARN user. To start MapReduce, run commands as a MapReduce user.

Note

The `su` commands in this section use "yarn" to represent the YARN Service user and mapreduce to represent the MAPREDUCE Service user. If you are using another name for these Service users, you will need to substitute your Service user name for "yarn" or "mapreduce" in each of the `su` commands.

1. Manually clear the ResourceManager state store.

```
su - yarn -c "yarn resourcemanager -format-state-store"
```

2. Start the ResourceManager on all your ResourceManager hosts.

```
su - yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.
sh start resourcemanager"

ps -ef | grep -i resourcemanager
```

3. Start the TimelineServer on your TimelineServer host.

```
su - yarn -c "/usr/hdp/current/hadoop-yarn-tinelineserver/sbin/yarn-daemon.
sh start timelineserver"

ps -ef | grep -i timelineserver
```

4. Start the NodeManager on all your NodeManager hosts.

```
su - yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh
 start nodemanager"

ps -ef | grep -i nodemanager
```

5. To start MapReduce, run the following commands:

```
su - mapreduce -c "/usr/hdp/current/hadoop-mapreduce-historyserver/sbin/mr-
jobhistory-daemon.sh start historyserver"

ps -ef | grep -i jobhistoryserver
```

1.10. Run Hadoop Smoke Tests

To smoke test your Hadoop upgrade, you can run the following MapReduce job as a regular user.

The job uses MapReduce to write 100MB of data into HDFS with RandomWriter

```
hadoop jar /usr/hdp/current/hadoop-mapreduce-client/hadoop-mapreduce-examples.
jar
            randomwriter -Dtest.randomwrite.total_bytes=10000000 test-after-
upgrade.
```

You should see messages similar to:

```
map 0% reduce 0%
…map 100% reduce 100%
Job …. completed successfully
```

MapReduce upgraded successfully. You can now upgrade your other components.

Basic Troubleshooting

To find the number of active nodes and NodeManagers, access the ResourceManager web UI:

`http://<resource manager host>:8088/cluster/nodes`

The number of active nodes should be equal to the number of nodemanagers.

Accessing error messages:

1. Access the ApplicationMaster WebUI to view the container logs.

2. At your console logs for MapReduce job, look for a line with this format:

   13/10/02 17:57:21 INFO mapreduce.Job: The url to track the job: http://<resource manager host>:8088/proxy/application_1380673658357_0007/

3. Select the logs link under ApplicationMaster table. It will redirect you to the container logs. Error messages display here.

1.11. Configure and Start Apache HBase

Before you can upgrade Apache HBase, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "hbase" to represent the HBASE Service user. If you are using another name for your HBASE Service user, you will need to substitute your HBASE Service user name for "hbase" in each of the `su` commands.

The hbase.bucketcache.percentage.in.combinedcache is removed in HDP 2.4.3. This simplifies the configuration of block cache. BucketCache configurations from HDP 2.3 will need to be recalculated to attain identical memory allotments in HDP 2.4.3. The L1 LruBlockCache will be whatever hfile.block.cache.size is set to and the L2 BucketCache will be whatever hbase.bucketcache.size is set to.

1. Replace your configuration after upgrading. Replace the HBase template configuration in `/etc/hbase/conf`.
2. Start services. From root, assuming that $HBASE_USER=hbase:

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

   ```
   su - hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-
   daemon.sh start regionserver"
   ```

3. Check processes.

   ```
   ps -ef | grep -i hmaster
   ```

   ```
   ps -ef | grep -i hregion
   ```

1.12. Configure Apache Phoenix

Before you can upgrade Apache Phoenix, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

To configure Phoenix, complete the following steps:

1. Add the following property to the `/etc/hbase/hbase-site.xml` file on all HBase nodes, the MasterServer, and all RegionServers to prevent deadlocks from occurring during maintenance on global indexes:

```
<property>
  <name>hbase.regionserver.wal.codec</name>
  <value>org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec</
value>
</property>
```

2. To enable user-defined functions, configure the following property in `/etc/hbase/conf` on all Hbase nodes.

```
<property>
 <name>phoenix.functions.allowUserDefinedFunctions</name>
 <value>true</value>
 <description>enable UDF functions</description>
</property>
```

3. Ensure the client side hbase-site.xml matches the server side configuration.

4. If the folder specified in `hbase.tmp.dir` property on `hbase-site.xml` does not exist, create that directory with adequate permissions.

5. Set the following porperty in the hbase-site.xml file for all RegionServers, but not on the client side:

```
<property>
  <name>hbase.rpc.controllerfactory.class</name>
  <value>org.apache.hadoop.hbase.ipc.controller.ServerRpcControllerFactory</
value>
</property>
```

6. Restart the HBase Master and RegionServers.

Configuring Phoenix to Run in a Secure Cluster

Perform the following additional steps to configure Phoenix to run in a secure Hadoop cluster:

1. To link the HBase configuration file with the Phoenix libraries:

   ```
   ln -sf HBASE_CONFIG_DIR/hbase-site.xml PHOENIX_HOME/bin/hbase-
   site.xml
   ```

2. To link the Hadoop configuration file with the Phoenix libraries:

   ```
   ln -sf HADOOP_CONFIG_DIR/core-site.xml PHOENIX_HOME/bin/core-
   site.xml
   ```

Note

When running the pssql.py and sqlline.py Phoenix scripts in secure mode, you can safely ignore the following warnings.

```
14/04/19 00:56:24 WARN util.NativeCodeLoader:
Unable to load native-hadoop library for your platform...
  using builtin-java classes where applicable

14/04/19 00:56:24 WARN util.DynamicClassLoader: Failed to identify the fs of
dir hdfs://<HOSTNAME>:8020/apps/hbase/data/lib, ignored java.io.IOException:
No FileSystem for scheme: hdfs
```

1.13. Configure and Start Apache Accumulo

Before you can upgrade Apache Accumulo, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "accumulo" to represent the Accumulo Service user. If you are using another name for your Accumulo Service user, you will need to substitute your Accumulo Service user name for "accumulo" in each of the `su` commands.

1. You must replace your configuration after upgrading. Copy `/etc/accumulo/conf` from the template to the conf directory in Accumulo hosts.

2. Start the services:

```
su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` master"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` tserver"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` gc"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` tracer"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` monitor"
```

3. Check that the processes are running

   ```
   ps -ef | grep accumulo
   ```

   or visit http://<hostname>:50095 in your browser

1.14. Configure and Start Apache Tez

Before you can upgrade Apache Tez, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using

another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

To upgrade Tez:

1. Copy your previously backed-up copy of tez-site.xml into the /etc/tez/conf directory.
2. Upload the Tez tarball to HDFS.

```
su - hdfs
hdfs dfs -mkdir -p /hdp/apps/<hdp_version>/tez/
hdfs dfs -put /usr/hdp/<hdp_version>/tez/lib/tez.tar.gz /hdp/apps/
<hdp_version>/tez/
hdfs dfs -chown -R hdfs:hadoop /hdp
hdfs dfs -chmod -R 555 /hdp/apps/<hdp_version>/tez
hdfs dfs -chmod -R 444 /hdp/apps/<hdp_version>/tez/tez.tar.gz
```

   Where <hdp_version> is the current HDP version, for example 2.4.3.0-258.

3. Edit the tez.lib.uris property in the tez-site.xml file to point to /hdp/apps/<hdp_version>/tez/tez.tar.gz

```
...
<property>
 <name>tez.lib.uris</name>
 <value>/hdp/apps/<hdp_version>/tez/tez.tar.gz</value>
</property>
...
```

   Where `<hdp_version>` is the current HDP version, for example `2.4.3.0-258`.

4. **Optional** Earlier releases of Tez did not have access control. In the current version of Tez, the default behavior restricts the ability to view the Tez history to only the owner of the job. To retain unrestricted access for non-secure clusters, set `tez.am.view-acls` set to "*".
5. Change the value of the tez.tez-ui.history-url.base property to the url for the upgraded Tez View. For information on setting up the Tez view, see Deploying the Tez View in the HDP Ambari Views Guide.

1.15. Configure and Start Apache Hive and Apache HCatalog

Before you can upgrade Apache Hive and Apache HCatalog, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "hive" to represent the Hive Service user. If you are using another name for your Hive Service user, you will need to substitute your Hive Service user name for "hive" in each of the `su` commands.

1. Prior to starting the upgrade process, set the following in your hive configuration file:

```
datanucleus.autoCreateSchema=false
```

2. Copy the jdbc connector jar from OLD_HIVE_HOME/lib to CURRENT_HIVE_HOME/lib.

3. Upgrade the Hive Metastore database schema. Restart the Hive Metastore database and run:

```
su - hive -c "/usr/hdp/current/hive-metastore/bin/schematool -upgradeSchema
 -dbType" <$databaseType>
```

The value for $databaseType can be derby, mysql, oracle, or postgres.

Note

If you are using Postgres 8 and Postgres 9, you should reset the Hive Metastore database owner to <HIVE_USER>:

```
psql -U <POSTGRES_USER> -c
ALTER DATABASE <HIVE-METASTORE-DB-NAME> OWNER TO <HIVE_USER>
```

Note

If you are using Oracle 11, you may see the following error message:

```
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
optimize.mapjoin.mapreduce does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
heapsize does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
server2.enable.impersonation does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
semantic.analyzer.factory.impl does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.auto.
convert.sortmerge.join.noconditionaltask does not exist
Metastore connection URL: jdbc:oracle:thin:@//ip-172-31-42-1.ec2.
internal:1521/XE
Metastore Connection Driver : oracle.jdbc.driver.OracleDriver
Metastore connection User: hiveuser
Starting upgrade metastore schema from version 0.13.0 to 0.14.0
Upgrade script upgrade-0.13.0-to-0.14.0.oracle.sql
Error: ORA-00955: name is already used by an existing object
 (state=42000,code=955)
Warning in pre-upgrade script pre-0-upgrade-0.13.0-to-0.14.0.
oracle.sql: Schema script failed, errorcode 2
Completed upgrade-0.13.0-to-0.14.0.oracle.sql
schemaTool completed
```

You can safely ignore this message. The error is in the pre-upgrade script and can be ignored; the schematool succeeded.

4. Edit the hive-site.xml file and modify the properties based on your environment. Search for TODO in the file for the properties to replace.

   a. Edit the following properties in the hive-site.xml file:

```
<property>
```

```
 <name>fs.file.impl.disable.cache</name>
 <value>false</value>
 <description>Set to false or remove fs.file.impl.disable.cache</
description>
</property>

<property>
 <name>fs.hdfs.impl.disable.cache</name>
 <value>false</value>
 <description>Set to false or remove fs.hdfs.impl.disable.cache
 </description>
<property>
```

b. **Optional:** To enable the Hive buildin authorization mode, make the following changes. If you want to use the advanced authorization provided by Ranger, refer to the Ranger instructions.

Set the following Hive authorization parameters in the hive-site.xml file:

```
<property>
 <name>hive.server2.enable.doAs</name>
 <value>false</value>
</property>

<property>
 <name>hive.security.metastore.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.
   StorageBasedAuthorizationProvider,org.apache.hadoop.hive.ql.security.
authorization.MetaStoreAuthzAPIAuthorizeEmbedOnly</value>
</property>

<property>
 <name>hive.security.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.
SQLStdConfOnlyAuthorizeFactory</value>
</property>
```

Also set hive.users.in.admin.role to the list of comma-separated users who need to be added to admin role. A user who belongs to the admin role needs to run the "set role" command before getting the privileges of the admin role, as this role is not in the current roles by default.

Set the following in the hiveserver2-site.xml file.

```
<property>
 <name>hive.security.authenticator.manager</name>
 <value>org.apache.hadoop.hive.ql.security.
SessionStateUserAuthenticator</value>
</property>

<property>
 <name>hive.security..authorization.enabled</name>
 <value>true</value>
</property>

<property>
 <name>hive.security.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.
SQLStdHiveAuthorizeFactory/value>
```

```
</property>
```

c. For a remote Hive metastore database, set the IP address (or fully-qualified domain name) and port of the metastore host using the following hive-site.xml property value.

```
<property>
 <name>hive.metastore.uris</name>
 <value>thrift://$metastore.server.full.hostname:9083</value>
 <description>URI for client to contact metastore server.
   To enable HiveServer2, leave the property value empty.
   </description>
</property>
```

You can further fine-tune your configuration settings based on node hardware specifications, using the HDP utility script.

5. Start Hive Metastore.

   On the Hive Metastore host machine, run the following command:

   ```
   su - hive -c "nohup /usr/hdp/current/hive-metastore/bin/hive
   --service metastore -hiveconf hive.log.file=hivemetastore.log
   >/var/log/hive/hivemetastore.out 2>/var/log/hive/
   hivemetastoreerr.log &"
   ```

6. Start Hive Server2.

   On the Hive Server2 host machine, run the following command:

   ```
   su - hive

   nohup /usr/hdp/current/hive-server2/bin/hiveserver2 -hiveconf
   hive.metastore.uris=" " -hiveconf hive.log.file=hiveserver2.log
   >/var/log/hive/hiveserver2.out 2> /var/log/hive/
   hiveserver2err.log &
   ```

1.16. Configure and Start Apache Oozie

Before you can upgrade Apache Oozie, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The duration of the Oozie upgrade is dependent on the amount of job history stored in ooziedb. This history must be backed up and restored during the upgrade process. Best practice when planning for upgrade is to backup ooziedb from your production oozie server and restore it to a test or development oozie server. This can help you estimate the time that will be required to upgrade Oozie during your production upgrade.

Note

The `su` commands in this section use "hdfs" to represent the HDFS Service user and "oozie" to represent the Oozie Service user. If you are using another name for your HDFS Service user or your Oozie Service user, you will need to substitute your Service user names for "hdfs" or "oozie" in each of the `su` commands.

Upgrading Oozie is a complex process. Although the instructions are straightforward, set aside a dedicated block of time to upgrade oozie clients and servers.

Perform the following preparation steps on each oozie server host:

1. You must restore oozie-site.xml from your backup to the conf directory on each oozie server and client.
2. Copy the JDBC jar to libext-customer:
   a. Create the /usr/hdp/current/oozie/libext-customer directory.

      ```
      cd /usr/hdp/current/oozie-server

      mkdir libext-customer
      ```

   b. Grant read/write/execute access to all users for the libext-customer directory.

      ```
      chmod -R 777 /usr/hdp/current/oozie-server/libext-customer
      ```

3. Copy these files to the libext-customer directory

   ```
   cp /usr/hdp/current/hadoop-client/lib/hadoop*lzo*.jar /usr/hdp/
   current/oozie-server/libext-customer

   cp /usr/share/HDP-oozie/ext.zip /usr/hdp/current/oozie-server/
   libext-customer/ext-2.2.zip
   ```

   Also, copy Oozie db jar in libext-customer.

4. If Falcon was also installed and configured before upgrade in HDP 2.3.x, then after upgrade you might also need to do the following:

   ```
   cp /usr/hdp/current/falcon-server/oozie/ext/falcon-oozie-el-
   extension-"jar /usr/hdp/current/oozie-server/libext-customer
   ```

5. Extract share-lib.

   ```
   /usr/hdp/current/oozie/bin/oozie-setup.sh sharelib create -fs
   hdfs://<namenode>:8020
   ```

   To verify that the sharelibs extracted correctly, run the following command:

   ```
   oozie admin -oozie http://<oozie server host address>:11000/
   oozie -shareliblist
   ```

   There should be:

- Available ShareLib
- oozie
- hive
- distcp
- hcatalog
- sqoop
- mapreduce-streaming
- pig

Change the ownership and permissions of the oozie directory:

```
su -l hdfs -c "hdfs dfs -chown oozie:hadoop /user/oozie"
```

```
su -l hdfs -c "hdfs dfs -chmod -R 755 /user/oozie"
```

6. If a previous version of Oozie was created using auto schema creation, run the following SQL query:

```
insert into oozie_sys (name, data) values ('db.version', '2.5');
```

7. As the Oozie user (not root), run the upgrade.

```
su - oozie -c "/usr/hdp/current/oozie-server/bin/ooziedb.sh
upgrade -run"
```

8. As root, prepare the Oozie WAR file.

```
chown oozie:oozie /usr/hdp/current/oozie-server/oozie-server/
conf/server.xml
```

```
su - oozie -c "/usr/hdp/current/oozie-server/bin/oozie-setup.sh
prepare-war -d /usr/hdp/current/oozie-server/libext-customer"
```

Look for console output to indicate success. For example, if you are using MySQL you should see something similar to:

```
INFO: Adding extension: libext-customer/mysql-connector-java.jar
New Oozie WAR file with added 'JARs' at /var/lib/oozie/oozie-server/webapps/
oozie.war
```

9. Make sure that following property is added in oozie-log4j.properties:

```
log4j.appender.oozie.layout.ConversionPattern=%d{ISO8601} %5p
%c{1}:%L - SERVER[${oozie.instance.id}] %m%n
```

where ${oozie.instance.id} is determined by oozie, automatically.

10. If you have custom Oozie actions, you must define them in oozie-site.xml. Edit the `/etc/oozie/conf/oozie-site.xml` file and add the following property:

```
<property>
      <name>oozie.service.SchemaService.wf.ext.schemas</name>
      <value>[Comma separated list of custom actions]</value>
    </property>
```

For example, if you have added Spark Action, enter the following:

```
<property>
      <name>oozie.service.SchemaService.wf.ext.schemas</name>
      <value>spark-action-0.1.xsd</value>
    </property>
```

11. Configure HTTPS for the Oozie server.

    a. Create a self signed certificate or get certificate from a trusted CA for the Oozie Server

    b. Import the certificate to the client JDK trust store on all client nodes.

    c. In the Ambari Oozie configuration, set the following environment variables in `oozie-env.sh`, adding them if it does not exist:

    ```
    export OOZIE_HTTPS_PORT=11443
    export OOZIE_HTTPS_KEYSTORE_FILE=/home/oozie/.keystore
    export OOZIE_HTTPS_KEYSTORE_PASS=password
    ```

    d. Change `OOZIE_HTTP_PORT={{oozie_server_port}}` to `OOZIE_HTTP_PORT=11000`.

    e. Set the `oozie.base.url` to the HTTPS address.

    f. Save the configuration, and restart the Oozie components.

12. Start Oozie as the Oozie user:

```
su - oozie -c "/usr/hdp/current/oozie-server/bin/oozie-start.sh"
```

13. Check processes.

```
ps -ef | grep -i oozie
```

1.17. Configure and Start Apache WebHCat

Before you can upgrade Apache WebHCat, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "hdfs" to represent the HDFS Service user and webhcat to represent the WebHCat Service user. If you are using another

name for these Service users, you will need to substitute your Service user name for "hdfs" or "webhcat" in each of the `su` commands.

1. You must replace your configuration after upgrading. Copy `/etc/webhcat/conf` from the template to the conf directory in webhcat hosts.

2. Modify the WebHCat configuration files.

   a. Upload Pig, Hive and Sqoop tarballs to HDFS as the $HDFS_User (in this example, hdfs):

```
su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/pig/"

su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/hive/"

su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/sqoop/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/pig/pig.tar.gz /hdp/
apps/2.4.3.0-$BUILD/pig/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/hive/hive.tar.gz /
hdp/apps/2.4.3.0-$BUILD/hive/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/sqoop/sqoop.tar.gz /
hdp/apps/2.4.3.0-$BUILD/sqoop/"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/pig"

su - hdfs - "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/pig/pig.tar.
gz"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/hive"

su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/hive/hive.
tar.gz"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/sqoop/
sqoop.tar.gz"

su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp"
```

   b. Update the following properties in the webhcat-site.xml configuration file, as their values have changed:

```
<property>
 <name>templeton.pig.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/pig/pig.tar.gz</value>
</property>

<property>
 <name>templeton.hive.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/hive/hive.tar.gz</value>
</property>

<property>
 <name>templeton.streaming.jar</name>
 <value>hdfs:///hdp/apps/${hdp.version}/mapreduce/
```

```
   hadoop-streaming.jar</value>
 <description>The hdfs path to the Hadoop streaming jar file.</
description>
</property>

<property>
 <name>templeton.sqoop.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/sqoop/sqoop.tar.gz</value>
 <description>The path to the Sqoop archive.</description>
</property>

<property>
 <name>templeton.sqoop.path</name>
 <value>sqoop.tar.gz/sqoop/bin/sqoop</value>
 <description>The path to the Sqoop executable.</description>
</property>

<property>
 <name>templeton.sqoop.home</name>
 <value>sqoop.tar.gz/sqoop</value>
 <description>The path to the Sqoop home in the exploded archive.
   </description>
</property>
```

Note

You do not need to modify ${hdp.version}.

c. Add the following property if it is not present in webhcat-sitemxml:

```
<property>
 <name>templeton.libjars</name>
 <value>/usr/hdp/current/zookeeper-client/zookeeper.jar,/usr/hdp/current/
hive-client/lib/hive-common.jar</value>
 <description>Jars to add the classpath.</description>
</property>
```

d. Remove the following obsolete properties from webhcat-site.xml:

```
<property>
 <name>templeton.controller.map.mem</name>
 <value>1600</value>
 <description>Total virtual memory available to map tasks.</description>
</property>

<property>
 <name>hive.metastore.warehouse.dir</name>
 <value>/path/to/warehouse/dir</value>
</property>
```

e. Add new proxy users, if needed. In core-site.xml, make sure the following properties are also set to allow WebHCat to impersonate your additional HDP 2.4.3 groups and hosts:

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

```
<property>
 <name>hadoop.proxyuser.hcat.hosts</name>
 <value>*</value>
</property>
```

Where:

`hadoop.proxyuser.hcat.group`

Is a comma-separated list of the Unix groups whose users may be impersonated by 'hcat'.

`hadoop.proxyuser.hcat.hosts`

A comma-separated list of the hosts which are allowed to submit requests by 'hcat'.

3. Start WebHCat:

```
su - webhcat -c "/usr/hdp/current/hive-webhcat/sbin/
webhcat_server.sh start"
```

4. Smoke test WebHCat.

   a. If you have a non-secure cluster, on the WebHCat host machine, run the following command to check the status of WebHCat server:

   `curl http://$WEBHCAT_HOST_MACHINE:50111/templeton/v1/status`

   You should see the following return status:

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

   b. If you are using a Kerberos secure cluster, run the following command:

```
curl --negotiate -u: http://$WEBHCAT_HOST_MACHINE:50111/
templeton/v1/status
```

   You should see the following return status

   `{"status":"ok","version":"v1"}[machine@acme]$`

1.18. Configure Apache Pig

Before you can upgrade Apache Pig, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Replace your configuration after upgrading. Copy /etc/pig/conf from the template to the conf directory in pig hosts.

1.19. Configure and Start Apache Sqoop

Before you can upgrade Apache Sqoop, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The su commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the su commands.

1. Replace your configuration after upgrading. Copy /etc/sqoop/conf from the template to the conf directory in sqoop hosts.
2. As the HDFS Service user, upload the Sqoop tarball to HDFS.

```
su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp/apps/2.4.3.0-$BUILD/
sqoop"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/sqoop/sqoop.tar.gz /hdp/
apps/2.4.3.0-$BUILD/sqoop/sqoop.tar.gz"

su - hdfs -c "hdfs dfs -chmod 444 /hdp/apps/2.4.3.0-$BUILD/sqoop/sqoop.tar.
gz"
```

3. If you are using the MySQL database as a source or target, then the MySQL connector jar must be updated to 5.1.29 or later.

   Refer to the MySQL web site for information on updating the MySQL connector jar.

1.20. Configure, Start, and Validate Apache Flume

Before you can upgrade Apache Flume, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

1. Replace your configuration after upgrading. Copy /etc/flume/conf from the template to the conf directory in Flume hosts.
2. By default, Flume does not start running immediately upon installation. To validate your Flume upgrade, replace your default conf/flume.conf with the provided flume.conf file, restart Flume, and see if the data is flowing by examining the destination.

   Use this flume.conf file:

```
#1. Name the components on this agent
a1.sources = r1
a1.sinks = k1
a1.channels = c1

#2.Describe/configure the source
a1.sources.r1.type = seq

#3. Describe the sink
a1.sinks.k1.type = file_roll
a1.sinks.k1.channel = c1
a1.sinks.k1.sink.directory = /tmp/flume

#4. Use a channel which buffers events in memory
a1.channels.c1.type = memory

#5. Bind the source and sink to the channel
a1.sources.r1.channels = c1
a1.sinks.k1.channel = c1
```

3. After starting Flume, check /tmp/flume to see if there are any files there. The files should contain simple sequential numbers.

4. After validating, stop Flume and revert changes to flume.conf to prevent your disk from filling up.

1.21. Configure, Start, and Validate Apache Mahout

Before you can upgrade Apache Mahout, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Replace your configuration after upgrading. Copy /etc/mahout/conf from the template to the conf directory in mahout hosts.

To validate mahout:

1. Create a test user:

```
su - hdfs -c "dfs -put /tmp/sample-test.txt /user/testuser"
```

2. Set up mahout to convert the plain text file sample-test.txt into a sequence file that is in the output directory mahouttest.

```
mahout seqdirectory --input /user/testuser/sample-test.txt --output /user/
 testuser/mahouttest --charset utf-8
```

1.22. Configure and Start Hue

Before you can upgrade Hue, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

For HDP 2.4.3, use the Hue version shipped with HDP 2.4.3. If you have a previous version of Hue, use the following steps to upgrade Hue.

1. Migrate hue.ini setting from your old hue.ini configuration file to new hue.ini configuration file.

2. If you are using the embedded SQLite database, remember to restore your database after upgrade.

   To restore the database from a backup, make sure the destination database is empty before copying (if necessary, rename or remove the current destination database), then copy your backup to the destination database. For example:

   ```
   su - hue
   cd /var/lib/hue
   mv desktop.db desktop.db.old
   sqlite3 desktop.db < ~/hue_backup/desktop.bak
   exit
   ```

3. Synchronize Database

   ```
   cd /usr/lib/hue
   source ./build/env/bin/activate
   hue syncdb
   hue migrate
   deactivate
   ```

4. Start Hue. As a root user, run the following command on the Hue Server:

   ```
   /etc/init.d/hue start
   ```

1.23. Configure and Start Apache Knox

Before you can upgrade Apache Knox, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already

upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

When working with the Knox Gateway in your Hadoop cluster, it is important you have the latest version of Knox installed so you can take advantage of new features and enhancements, in addition to ensuring your instance of Knox is in sync with other Hadoop components (e.g. Ranger, Spark, Hive, Hue, etc.) for stability and performance. For example, if you need to upgrade your Hadoop cluster from 2.2 to 2.3.x, you should also make sure that your individual Hadoop components are also upgraded to the latest version.

HDP enables you to perform a rolling upgrade in 2.2.x and onward. A rolling upgrade means that you can upgrade a component, or the entire Hadoop stack, without losing service, and your users can continue to use the cluster and run jobs with no application or server downtime. The main distinction between a rolling upgrade and a traditional upgrade implies the use of a Knox cluster for high availability capabilities. This means that you can require multiple instances of the gateway and a load balancer serving each gateway instance from a single URL.

The main distinction between a rolling upgrade and a traditional upgrade implies the use of a Knox cluster for high availability capabilities. This means that you can require multiple instances of the gateway and a load balancer serving each gateway instance from a single URL. Once the upgrade process is completed, you will be up and running with the latest version of Knox on each server you have designated as a Knox server.

Note

In this document, whenever you see a {} with a value inside, this denotes a value you must define.

1.23.1. Upgrade the Knox Gateway

If you are not currently using Ambari to manage your Hadoop cluster, you will need to upgrade Knox manually to the latest version. Because “rolling upgrades” are now supported in HDP 2.4.3, it is not important which version of Knox you are currently running, only that you have an instance of the Knox Gateway running.

Note

If you have not already installed Knox, refer to the "Install the Knox RPMs on the Knox Server" section of the *Non-Ambari Cluster Installation Guide* for instructions on how to install and configure the Knox Gateway.

Before upgrading the Knox Gateway, there are a several steps you must follow to ensure your configuration files, settings, and topology files can be copied to the new Knox Gateway instance when the upgrade is complete, which are described below.

1. Back up your existing `conf` directory if you have not already done so.
2. Stop each Knox server if you have not already done so.

```
su -l knox /usr/hdp/{the current Knox version}/knox/bin/gateway.sh stop
```

3. Select the HDP server version you are upgrading to after you have stopped each Knox server if you have not already done so.

```
hdp-select set knox-server {the HDP server version}
```

4. 

Note

The `su` commands in this section use "knox" to represent the Knox Service user. If you are using another name for your Knox Service user, you will need to substitute your Knox Service user name for "knox" in each of the `su` commands.

For HDP 2.4.3, the default paths for Knox change. Upgrade Knox in order to update these paths.

a. Restore the backed up security directory. This will place the master secret and keystores back in place for the new deployment.

b. Start the Gateway:

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

c. Unzip your previously saved configuration directory (the `conf` directory you backed up in step 1) into the new */var/log/knox/gateway.conf* directory to import these files.

d. Restart the Knox server to complete the upgrade.

```
su -l knox /usr/hdp/{the new HDP server version}/knox/bin/gateway.sh
 start
```

1.23.2. Verify the Knox Upgrade

To verify the upgrade was successful, follow the steps listed below.

1. Navigate to the */var/log/knox/gateway* directory and check the `gateway.log`file for errors and an acknowledgement that the server restart was successful.

2. Verify you have cluster access using the `LISTSTATUS` WebHDFS API call.

```
curl -ivk -u {user}:{password} https://{knox host}:8443 /gateway/webhdfs/v1/
tmp?op=LISTSTATUS
```

3. Verify the Knox version using the Knox Admin service and Version API.

```
curl -ivk -u {adminuser}:{adminpassword} https://{knox host}:8443 /gateway/
admin/v1/version
```

Note

The Admin API requires you to be a member of an Admin group, as specified in the `admin.xml` authorization provider.

When you have verified the Knox upgrade was successful, you can begin using Knox. If, however, the upgrade was unsuccessful, you will need to downgrade the Knox Gateway to the previous version. The steps to downgrade the Knox Gateway are described in the next section.

1.23.3. Downgrade the Knox Gateway to the Previous Version

If the Knox Gateway upgrade was unsuccessful, you will need to downgrade Knox to the previous version to ensure you have a working Knox Gateway for your cluster. To downgrade Knox, follow the steps listed below.

1. For each server running Knox, stop the server.

```
su -l knox /usr/hdp/{current HDP server version}/knox/bin/gateway.sh stop
```

2. Select the HDP server version you want to use to downgrade your Knox Gateway.

```
hdp-select set knox-server {previous HDP server version}
```

3. Restart the server.

```
su -l knox /usr/hdp/{previous HDP server version}/knox/bin/gateway.sh start
```

1.23.4. Verify the Knox Downgrade Was Successful

When the restart is complete, verify you are running an older version of Knox by following the steps listed below.

1. Navigate to the */var/log/knox/gateway* directory and check the `gateway.log`file for errors and an acknowledgement that the server restart was successful
2. Verify you have cluster access using the `LISTSTATUS`WebHDFS API call.
3. Check the Knox version using the Knox Admin service and Version API using the following command:

```
curl -ivk -u {adminuser}"{adminpassword} https://{knox host}:8443 /gateway/
admin/v1/version
```

1.24. Configure and Validate Apache Falcon

Before you can upgrade Apache Falcon, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

In HDP 2.4.3, if authorization is enabled (for example, in the properties file with `*.falcon.security.authorization.enabled=true`") then Access Control List (ACL) is mandated for all entities.

Upgrade Falcon after you have upgraded HDFS, Hive, Oozie, and Pig. Stop Oozie jobs while running Falcon.

1. Replace your configuration after upgrading. Copy /etc/falcon/conf from the template to the conf directory in falcon hosts.

2. Check your Falcon entities. There should be no changes, but in some cases you may need to update your entities post-upgrade.

3. In HDP 2.4.3 for Falcon, TLS is enabled by default. When TLS is enabled, Falcon starts on https://<falcon_host>.15443/. You can disable TLS by adding the following line to the startup.properties file.

   ```
   "*.falcon.enableTLS=false
   ```

4. If Transport Layer Security (TLS) is disabled, check the client.properties file to make sure the property "falcon.uri" is set as follows:

   ```
   falcon.uri=http://<falcon_host>:15000/
   ```

1.25. Configure and Start Apache Storm

Before you can upgrade Apache Storm, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "zookeeper" to represent the ZooKeeper Service user. If you are using another name for your ZooKeeper Service user, you will need to substitute your ZooKeeper Service user name for "zookeeper" in each of the `su` commands.

Storm is fairly independent of changes to the HDP cluster, but you must upgrade Storm for rolling upgrade support in HDP 2.4.3 and be on the latest version of Storm.

1. Deactivate all running topologies.

2. Delete all states under zookeeper:

   ```
   /usr/hdp/current/zookeeper-client/bin/zkCli.sh (optionally in
   secure environment specify -server zk.server:port)

   rmr /storm
   ```

3. Delete all states under the storm-local directory:

   ```
   rm -rf <value of stormlocal.dir>
   ```

4. Stop Storm Services on the storm node.

5. Stop ZooKeeper Services on the storm node.

   ```
   su - zookeeper -c "export ZOOCFGDIR=/etc/zookeeper/conf ; export
   ZOOCFG=zoo.cfg ;source /etc/zookeeper/conf/zookeeper-env.sh ; /
   usr/lib/zookeeper/bin/zkServer.sh stop"
   ```

6. Remove Storm and zookeeper from the storm node and install the HDP 2.4.3 version:

- For **RHEL/CentOS/Oracle Linux**:

  ```
  yum erase storm

  yum erase zookeeper

  yum install storm

  yum install zookeeper
  ```

- For **SLES**:

  ```
  zypper rm storm

  zypper rm zookeeper

  zypper install storm

  zypper install zookeeper
  ```

- For **Ubuntu/Debian**:

  HDP support for Debian 6 is deprecated with HDP 2.4.3. Future versions of HDP will no longer be supported on Debian 6.

  ```
  apt-get remove storm --purge

  apt-get remove zookeeper --purge

  apt-get install storm

  apt-get install zookeeper
  ```

7. Replace your configuration after upgrading. Copy `/etc/storm/conf` from the template to the conf directory .

8. Replace your ZooKeeper configuration after upgrading. Replace the ZooKeeper template configuration in `/etc/zookeeper/conf`.

9. Ensure ZooKeeper is running. On the storm node, run the following command:

   ```
   su - zookeeper -c "source /etc/zookeeper/conf/zookeeper-env.sh; export
    ZOOCFGDIR=/etc/zookeeper/conf; /usr/hdp/current/zookeeper-server/bin/
   zkServer.sh start >> $ZOO_LOG_DIR/zoo.out\"
   ```

   where

   - $ZOO_LOG_DIR is the directory where ZooKeeper server logs are stored. For example, /var/log/zookeeper.

10. Start nimbus, then supervisor/ui/drpc/logviewer:

    `/usr/hdp/current/storm-nimbus/bin/storm nimbus`.

11. Start Storm, using a process controller, such as supervisor:

```
su - storm /usr/hdp/current/storm-supervisor/bin/storm
supervisor
```

You can use the same command syntax to start Storm using nimbus/ui and logviewer.

```
su - storm /usr/hdp/current/storm-supervisor/bin/storm nimbus

su - storm /usr/hdp/current/storm-supervisor/bin/storm ui

su - storm /usr/hdp/current/storm-supervisor/bin/storm logviewer

su - storm /usr/hdp/current/storm-supervisor/bin/storm drpc
```

1.26. Configure and Start Apache Ranger

Before you can upgrade the Apache Ranger service, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already performed the following tasks, however, if you have not already performed these steps, refer to the "Upgrade HDP 2.3 Components" section in this guide for instructions on how to upgrade your HDP components to 2.4.3.

1.26.1. Preparing Your Cluster to Upgrade Ranger

If you are not currently using Ambari to manage your Hadoop cluster, you will need to upgrade Ranger manually to the latest version. This section describes the steps you need to follow to prepare your cluster for the Ranger upgrade.

1. Back up the following Ranger configuration directories:
   - Ranger Policy Administration Service

     `/etc/ranger/admin/conf`
   - Ranger UserSync

     `/etc/ranger/usersync/conf`
   - Ranger Plugins:
     - Hadoop

       `/etc/hadoop/conf`
     - Hive

       `/etc/hive/conf`
     - HBase

       `/etc/hbase/conf`
     - Knox

       `/etc/knox/conf`

- Storm

```
/etc/storm/conf
```

2. Backup the Ranger Policy and Audit databases. Make sure to take note of the following details in the `install.properties` file:

   - db_host
   - db_name
   - db_user
   - db_password
   - policy manager configuration
   - LDAP directory configuration
   - LDAP settings
   - LDAP AD domain
   - LDAP URL

```
mysqldump -u root -p root xasecure > dest_dir/filename.sql
mysqldump -u root -p root xasecure_audit > dest_dir/filename.sql
```

1.26.2. Stop the Ranger Services

Now that you have prepared your cluster for the Ranger upgrade, you will need to stop the Ranger Admin and Ranger UserSync services. To stop the Ranger services, perform the steps described below.

1. Stop the Ranger Policy Admin service. When the service is stopped, you will receive an acknowledgement from the server that the service has been stopped.

```
service ranger-admin stop
```

2. Stop the Ranger UserSync service. When the service is stopped, you will receive an acknowledgement from the server that the service has been stopped.

```
service ranger-usersync stop
```

3. Stop the applicable services using the Ranger plugin (HDFS, HBase, Hive, Knox, Storm).

   See Stopping HDP Services for more information.

1.26.3. Preparing the Cluster for Upgrade

Before you begin the upgrade process, you will need to perform a series of steps to prepare the cluster for upgrade. These steps are described in the "*Getting Ready To Upgrade*" section of this guide, which you will need to follow before continuing to upgrade Ranger. Some of these steps include:

- Backing up HDP directories
- Stopping all long-running applications and services.
- Backing up the Hive and Oozie metastore databases.
- Backing up Hue
- Backing up specific directories and configurations

1.26.4. Registering the HDP 2.4.3 Repo

After you have prepared your cluster for the upgrade, you need to register the HDP 2.4.3 repo. This requires you to perform the following steps:

1. Update the `install.properties` file to migrate the database credentials properties and POLICYMGR_EXTERNAL-URL property from HDP 2.3. to HDP 2.4.3.
2. Change ownership of `/etc/ranger/admin/conf/` and `/etc/ranger/usersync/conf/` to `ranger:ranger`:

```
chown -R ranger:ranger /etc/ranger/admin/conf/
```

3. Install the Ranger Admin component. Be sure to set the JAVA_HOME environment variable if it is not already set.

```
cd /usr/hdp/current/ranger-admin/

cp  ews/webapp/WEB-INF/classes/conf.dist/ranger-admin-site.xml
         ews/webapp/WEB-INF/classes/conf/

cp  ews/webapp/WEB-INF/classes/conf.dist/ranger-admin-default-site.xml
        ews/webapp/WEB-INF/classes/conf/

cp  ews/webapp/WEB-INF/classes/conf.dist/security-applicationContext.xml
        ews/webapp/WEB-INF/classes/conf/

./setup.sh
```

4. This should successfully install the Ranger Admin component.
5. Start the Ranger Admin component.

```
service ranger-admin start
```

6. You must now configure and setup the Ranger UserSync component by migrating the properties from the HDP 2.3 `install.properties` file (POLICY_MGR_URL, SYNC_SOURCE and LDAP/AD properties.
7. Install the Ranger UserSync component. Be sure to set the JAVA_HOME component if it is not already set.

```
cd /usr/hdp/current/ranger-usersync/

./setup.sh
```

8. Start the Ranger UserSync component.

```
service ranger-usersync start
```

1.26.5. Install the Ranger Components

Next, you will need to re-install each Ranger component again to ensure you have the latest version. Because you have already upgraded your HDP stack, you only need to follow the instructions in the *Non-Ambari Cluster Installation Guide* to install each Ranger component. You must install the following Ranger components:

- Ranger Policy Admin
- Ranger UserSync
- Ranger Plugins:
  - HDFS
  - HBase
  - Hive
  - Knox
  - Storm

Note

When installing each Ranger component, you will also need to make sure you upgrade each individual component to version 2.4.3 before restarting each service.

With this release, Ranger has also added support for the following components:

- Solr
- Kafka
- YARN

1.26.6. Restart the Ranger Services

Once you have re-installed each Ranger component, you will then need to restart these components to ensure the new configurations are loaded in your cluster. This includes restarting the Policy Admin and UserSync components, NameNode, and each Ranger plugin.

Note

Before restarting the NameNode, make sure to remove the `set-hdfs-plugin-env.sh` from */etc/hadoop/conf*. You will need to re-enable the NameNode after finishing the upgrade.

The *Non-Ambari Cluster Installation Guide* describes how you can start the following Ranger services:

- Ranger Policy Admin service

```
service ranger-admin start
```

- Ranger UserSync service

```
service ranger-usersync start
```

1.26.7. Enable Ranger Plugins

The final step in the Ranger upgrade process requires you to re-enable the Ranger plugins. Although you are only required to enable HDFS in your cluster, you should re-enable all of the Ranger plugins because class names have changed for the 2.4.3 release, and to ensure smooth operation of Ranger services in your cluster.

Note

When you enable each Ranger plugin, be sure to remove all 2.3 class name values.

Note

Re-enabling a Ranger plugin does not affect policies you have already created. As long as you use the same database as the Policy store, all of your data will remain intact.

To re-enable the Ranger plugins, use the links listed below to access instructions in the *Non-Ambari Cluster Installation* guide that describe editing the `install.properties` file and enabling the Ranger plugins:

Important

Before enabling the HDFS plugin, remove `set-hdfs-plugin-env.sh` from `/etc/hadoop/conf`. You will need to re-enable this plugin after the upgrade is complete.

- HDFS Plugin
- YARN Plugin
- Kafka Plugin
- HBase Plugin
- Hive Plugin
- Knox Plugin
- Storm Plugin

1.27. Configuring and Upgrading Apache Spark

Before you can upgrade Apache Spark, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

To upgrade Spark, start the service and update configurations.

1. Stop the Spark `history-server`. If you are using the Spark `thrift-server`, stop the `thrift-server`.

```
su - spark -c "$SPARK_HOME/sbin/stop-history-server.sh"
su - spark -c "$SPARK_HOME/sbin/stop-thriftserver.sh"
```

2. Remove any reference to `hdp.version` from the Spark configuration files.

3. In HDP 2.4.3, the Spark History Server runs on top of HDFS, not YARN ATS, as in previous versions. Modify Spark configuration files as follows:

   a. As the hdfs service user, create an HDFS directory called spark-history with user:spark, user group:hadoop, and permissions = 777:

```
hdfs dfs -mkdir /spark-history
hdfs dfs -chown -R spark:hadoop /spark-history
hdfs dfs -chmod -R 777 /spark-history
```

   b. Edit the `spark-defaults.conf` file.

   - Add the following properties and values:

```
spark.eventLog.dir to hdfs:///spark-history
spark.eventLog.enabled to true
spark.history.fs.logDirectory to hdfs:///spark-history
```

   c. Edit the `spark-thrift-sparkconf.conf` file

   - Add the following properties and values:

```
spark.eventLog.dir to hdfs:///spark-history
spark.eventLog.enabled to true
spark.history.fs.logDirectory to hdfs:///spark-history
spark.hadoop.cacheConf to false
```

4. Restart the `history-server`:

```
su - spark -c "/usr/hdp/current/spark-historyserver/sbin/start-history-
server.sh"
```

5. If you are using the Spark thrift-server, restart the `thrift-server`. See (Optional) Starting the Spark Thrift Server.

6. Validate the Spark installation. As user `spark`, run the Spark Pi example in the *Spark Guide*.

For additional configuration information, see the Spark Guide.

1.28. Upgrade Apache Slider

Before you can upgrade Apache Slider, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

To upgrade Slider, simply upgrade the Slider client.

1. Upgrade Slider client:

```
hdp-select set slider-client 2.4.3.0-$BUILD
slider version
```

1.29. Upgrade Apache Kafka

Before you can upgrade Apache Kafka, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

For more information about Kafka on HDP, see the Kafka Guide

Upgrade each Kafka node one at a time. You can stop each Kafka broker and upgrade the component without downtime if you have enough replication for your topic.

1. Shut down the current Kafka daemon, switch to the new version, and start the daemon:

```
su - kafka -c "/usr/hdp/current/kafka-broker/bin/kafka stop"
hdp-select set kafka-broker 2.4.3.0-$BUILD
su - kafka -c "usr/hdp/current/kafka-broker/bin/kafka start"
```

2. To verify that the Kafka daemon joined the cluster, create a topic and submit it to Kafka. Send a test message for that topic, and then validate that it was received by a consumer.

3. If the upgrade process fails, follow the steps in "Downgrading Kafka" to return to your previous version of Kafka.

1.29.1. Downgrading Kafka

Downgrade each Kafka node one at a time. You can stop each Kafka broker and upgrade the component without downtime if you have enough replication for your topic.

1. Shut down the current Kafka daemon, switch to the previous version, and start the daemon:

```
su - kafka -c "/usr/hdp/current/kafka-broker/bin/kafka stop"
hdp-select set kafka-broker 2.4.3.0-$BUILD
su - kafka -c "/usr/hdp/current/kafka-broker/bin/kafka start"
```

2. To verify that the Kafka daemon joined the cluster, create a topic and submit it to Kafka. Send a test message for that topic, and then validate that it was received by a consumer.

1.30. Upgrading Apache Atlas

Before you can upgrade Apache Atlas, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.3 Components for instructions on how to upgrade your HDP components to 2.4.3.

To upgrade Atlas, perform the following steps:

1. From the command line, enter the following command:

```
yum upgrade atlas-metadata
```

2. Restart Atlas:

```
/usr/hdp/current/atlas-server/bin/atlas_start.py
```

1.31. Finalize Upgrade

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

You can start HDFS without finalizing the upgrade. When you are ready to discard your backup, you can finalize the upgrade.

1. Verify your file system health before finalizing the upgrade. (After you finalize an upgrade, your backup will be discarded!)

2. As the $HDFS_USER, enter:

```
su - hdfs -c "dfsadmin -finalizeUpgrade"
```

2. Upgrade from HDP 2.2 to HDP 2.4.3 Manually

Important

If you installed and manage HDP 2.2 with Ambari, **you must use the Ambari Upgrade Guide** to perform the HDP 2.2 to HDP 2.4.3 upgrade.

Note

These instructions cover the upgrade between two minor releases, such as HDP-2.2 to HDP-2.4. Component information has been updated for HDP-2.4.3 where appropriate. If you need to upgrade between two maintenance releases such as HDP-2.3.0 to 2.3.2, follow the upgrade instructions in the HDP Release Notes.

Starting with HDP 2.2, HDP 2.4.3 supports side-by-side installation of HDP 2.2 and subsequent releases, which lets you perform rolling upgrades on your cluster and improve execution times for in-place upgrade. To support side-by-side installation, the HDP package version naming convention for both RPMs and Debs has changed to include the HDP product version. For example, hadoop-hdfs is now hadoop-2.4.3-hdfs. HDP 2.2 marked the first release where HDP rpms, debs, and directories contained versions in the names to permit side-by-side installations of later HDP releases. To select from the releases you have installed side-by-side, Hortonworks provided hdp-select, a command that lets you select the active version of HDP from the versions you have installed.

Now with HDP 2.4.3, you can take advantage of this versioned installation to perform Rolling Upgrade from 2.2 to 2.4.3. However, Rolling Upgrade involves complex orchestration as well as side-by-side installation. It is too complex for a manual procedure, and is therefore supported only as an Ambari feature. If you wish to perform a Rolling Upgrade, refer to the Ambari Install instructions to install Ambari, then follow the Ambari Rolling Upgrade instructions, see Ambari Upgrade Guide.

This chapter provides instructions on how to manually upgrade to HDP 2.4.3 from the HDP 2.2 release. It assumes the existing HDP 2.2 was also installed manually.

The HDP packages for a complete installation of HDP 2.4.3 will occupy about 2.5 GB of disk space.

Warning

Until the upgrade is finalized, no HDFS data is deleted from the cluster. Be sure to review your capacity and ensure that you have extra space available during the upgrade window.

The following provides an overview of steps for upgrading to the latest release of HDP 2.4.3 from HDP 2.2:

1. Get ready to upgrade
2. Upgrade HDP 2.2 components

3. Use hdp-select to symlink the HDP 2.4.3 components into "current," in place of the former HDP 2.2 components
4. Configure and Start Apache ZooKeeper
5. Configure and Start Hadoop
6. Start HDFS
7. Configure and start YARN/MapReduce
8. Configure and Start Apache HBase
9. Configure and Start Apache Phoenix
10. Configure and Start Apache Accumulo
11. Configure and Start Apache Tez
12. Configure and Start Apache Hive and Apache HCatalog
13. Configure and Start Apache Oozie
14. Configure and Start Apache WebHCat (Templeton)
15. Configure and Start Apache Pig
16. Configure and Start Apache Sqoop
17. Configure and Start Apache Flume
18. Configure and Start Apache Mahout
19. Configure and Start Hue
20. Configure and Start Apache Knox
21. Configure and Start Apache Falcon
22. Configure and Start Apache Storm
23. Configure and Start Apache Ranger
24. Configure and Start Apache Spark
25. Upgrade Apache Slider
26. Upgrade Apache Kafka
27. Finalize Upgrade
28. Install new HDP 2.3 services if desired

2.1. Getting Ready to Upgrade

HDP Stack upgrade involves upgrading from HDP 2.2 to HDP 2.4.3 versions and adding the new HDP 2.3 services. These instructions change your configurations.

Hardware recommendations

Although there is no single hardware requirement for installing HDP, there are some basic guidelines. The HDP packages for a complete installation of HDP 2.4.3 will take up about 2.5 GB of disk space.

The first step is to ensure you keep a backup copy of your HDP 2.2 configurations.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Note

You must use **kinit** before running the commands as any particular user.

1. Back up the HDP directories for any hadoop components you have installed.

   The following is a list of all HDP directories:

   - `/etc/hadoop/conf`
   - `/etc/hbase/conf`
   - `/etc/phoenix/conf`
   - `/etc/hive-hcatalog/conf`
   - `/etc/hive-webhcat/conf`
   - `/etc/accumulo/conf`
   - `/etc/hive/conf`
   - `/etc/pig/conf`
   - `/etc/sqoop/conf`
   - `/etc/flume/conf`
   - `/etc/mahout/conf`
   - `/etc/oozie/conf`
   - `/etc/hue/conf`
   - `/etc/knox/conf`
   - `/etc/zookeeper/conf`
   - `/etc/tez/conf`

- `/etc/falcon/conf`
- `/etc/slider/conf/`
- `/etc/storm/conf/`
- `/etc/storm-slider-client/conf/`
- `/etc/spark/conf/`
- `/etc/ranger/admin/conf, /etc/ranger/usersync/conf` (If Ranger is installed, also take a backup of install.properties for all the plugins, ranger admin & ranger usersync.)
- Optional - Back up your userlogs directories, `${mapred.local.dir}/userlogs`.

2. Oozie runs a periodic purge on the shared library directory. The purge can delete libraries that are needed by jobs that started before the upgrade began and that finish after the upgrade. To minimize the chance of job failures, you should extend the `oozie.service.ShareLibService.purge.interval` and `oozie.service.ShareLibService.temp.sharelib.retention.days` settings.

   Add the following content to the the `oozie-site.xml` file prior to performing the upgrade:

```
<property>
<name>oozie.service.ShareLibService.purge.interval</name>
<value>1000</value><description>
How often, in days, Oozie should check for old ShareLibs and LauncherLibs to
 purge from HDFS.
</description>
</property>

<property>
<name>oozie.service.ShareLibService.temp.sharelib.retention.days</name>
<value>1000</value>
<description>
ShareLib retention time in days.</description>
</property>
```

3. Stop all long-running applications deployed using Slider:

   ```
   su - yarn "/usr/hdp/current/slider-client/bin/slider list"
   ```

   For all applications returned in previous command, run `su - yarn "/usr/hdp/current/slider-client/bin/slider stop <app_name>"`

4. Stop all services (including MapReduce) except HDFS, ZooKeeper, and Ranger, and client applications deployed on HDFS.

   See Stopping HDP Services for more information.

Component	Command
Accumulo	`/usr/hdp/current/accumulo-client/bin$ / usr/hdp/current/accumulo-client/bin/stop-all.sh`
Knox	`cd $GATEWAY_HOME su - knox -c "bin/ gateway.sh stop"`
Falcon	`su - falcon "/usr/hdp/current/falcon-server/bin/falcon-stop"`
Oozie	`su - oozie -c "/usr/hdp/current/oozie-server/bin/oozie-stop.sh"`
WebHCat	`su - webhcat -c "/usr/hdp/current/hive-webhcat/sbin/webhcat_server.sh stop"`

Hive	Run this command on the Hive Metastore and Hive Server2 host machine:
	
	`ps aux | awk '{print $1,$2}' | grep hive | awk '{print $2}' | xargs kill >/dev/ null 2>&1`
	
	Or you can use the following:
	
	`Killall -u hive -s 15 java`

HBase RegionServers	`su - hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-daemon.sh -- config /etc/hbase/conf stop regionserver"`
HBase Master host machine	`su - hbase -c "/usr/hdp/current/hbase-master/bin/hbase-daemon.sh --config /etc/ hbase/conf stop master"`

YARN & Mapred Histro	Run this command on all NodeManagers:
	
	`su - yarn -c "export HADOOP_LIBEXEC_DIR=/ usr/hdp/current/hadoop-client/libexec && /usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh -- config /etc/hadoop/conf stop nodemanager"`
	
	Run this command on the History Server host machine:
	
	`su - mapred -c "export HADOOP_LIBEXEC_DIR=/usr/hdp/current/ hadoop-client/libexec && /usr/hdp/ current/hadoop-mapreduce-historyserver/ sbin/mr-jobhistory-daemon.sh --config / etc/hadoop/conf stop historyserver"`
	
	Run this command on the ResourceManager host machine(s):
	
	`su - yarn -c "export HADOOP_LIBEXEC_DIR=*/usr/hdp/current/ hadoop-client/libexec && /usr/hdp/ current/hadoop-yarn-resourcemanager/sbin/ yarn-daemon.sh --config /etc/hadoop/conf stop resourcemanager"`
	
	Run this command on the ResourceManager host machine:
	
	`su - yarn -c "export HADOOP_LIBEXEC_DIR=/ usr/hdp/current/hadoop-client/libex && /usr/hdp/current/hadoop-yarn-timelineserver/sbin/yarn-daemon.sh -- config /etc/hadoop/conf"`

Component	Command
	Run this command on the YARN Timeline Server node: `su -l yarn -c "export HADOOP_LIBEXEC_DIR=*/usr/hdp/current/hadoop-client/libex *&& /usr/hdp/current/hadoop-yarn-timelineserver/sbin/yarn-daemon.sh --config /etc/hadoop/conf stop timelineserver"`

Storm	Deactivate all running topologies:
	
	`storm kill topology-name`
	
	Delete all states under zookeeper:
	
	`/usr/hdp/current/zookeeper-client/bin/zkCli.sh (optionally in secure environment specify -server zk.server:port)`
	
	`rmr /storm`
	
	Delete all states under the storm-local directory:
	
	`rm -rf <value of stormlocal.dir>`
	
	Stop Storm Serves on the storm node:
	
	`sudo service supervisord stop`
	
	Stop ZooKeeper Services on the storm node:
	
	`su - zookeeper -c "export ZOOCFGDIR=/etc/zookeeper/conf ; export ZOOCFG=zoo.cfg ;source /etc/zookeeper/conf/zookeeper-env.sh ; /usr/hdp/current//zookeeper-server/bin/zkServer.sh stop"`

Spark (History server)	`su - spark -c "/usr/hdp/current/spark-client/sbin/stop-history-server.sh"`

5. If you have the Hive component installed, back up the Hive Metastore database.

   The following instructions are provided for your convenience. For the latest backup instructions, see your database documentation.

Table 2.1. Hive Metastore Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sqlsbr For example: mysqldump hive > /tmp/mydir/backup_hive.sql	mysql $dbname < $inputfilename.sqlsbr For example: mysql hive < /tmp/mydir/backup_hive.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql sbr For example: sudo -u postgres pg_dump hive > /tmp/mydir/backup_hive.sql	sudo -u $username psql $databasename < $inputfilename.sqlsbr For example: sudo -u postgres psql hive < /tmp/mydir/backup_hive.sql

Database Type	Backup	Restore
Oracle	Export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database file=input_file.dmp

6. If you have the Oozie component installed, back up the Oozie metastore database.

   These instructions are provided for your convenience. Check your database documentation for the latest backup instructions.

Table 2.2. Oozie Metastore Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sql For example: mysqldump oozie > /tmp/mydir/backup_oozie.sql	mysql $dbname < $inputfilename.sql For example: mysql oozie < /tmp/mydir/backup_oozie.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql For example: `sudo -u postgres pg_dump oozie > /tmp/mydir/backup_oozie.sql`	sudo -u $username psql $databasename < $inputfilename.sql For example: `sudo -u postgres psql oozie < /tmp/mydir/backup_oozie.sql`
Oracle	Export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database file=input_file.dmp

7. **Optional**: Back up the Hue database.

   The following instructions are provided for your convenience. For the latest backup instructions, please see your database documentation. For database types that are not listed below, follow your vendor-specific instructions.

Table 2.3. Hue Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sqlsbr For example: mysqldump hue > /tmp/mydir/backup_hue.sql	mysql $dbname < $inputfilename.sqlsbr For example: mysql hue < /tmp/mydir/backup_hue.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql sbr For example: sudo -u postgres pg_dump hue > /tmp/mydir/backup_hue.sql	sudo -u $username psql $databasename < $inputfilename.sqlsbr For example: sudo -u postgres psql hue < /tmp/mydir/backup_hue.sql
Oracle	Connect to the Oracle database using sqlplus. Export the database.	Import the database:

Database Type	Backup	Restore
	For example: exp username/password@database full=yes file=output_file.dmp mysql $dbname < $inputfilename.sqlsbr	For example: imp username/password@database file=input_file.dmp
SQLite	/etc/init.d/hue stop su $HUE_USER mkdir ~/hue_backup sqlite3 desktop.db .dump > ~/hue_backup/desktop.bak /etc/init.d/hue start	/etc/init.d/hue stop cd /var/lib/hue mv desktop.db desktop.db.old sqlite3 desktop.db < ~/hue_backup/desktop.bak /etc/init.d/hue start

8. Back up the Knox data/security directory.

   ```
   cp -RL /etc/knox/data/security ~/knox_backup
   ```

9. Save the namespace by executing the following commands:

   ```
   su - hdfs

   hdfs dfsadmin -safemode enter

   hdfs dfsadmin -saveNamespace
   ```

   

   Note

   In secure mode, you must have Kerberos credentials for the hdfs user.

10. Run the `fsck` command as the HDFS Service user and fix any errors. (The resulting file contains a complete block map of the file system.)

    ```
    su - hdfs -c "hdfs fsck / -files -blocks -locations > dfs-old-fsck-1.log"
    ```

    

    Note

    In secure mode, you must have Kerberos credentials for the hdfs user.

11. Use the following instructions to compare status before and after the upgrade.

    The following commands must be executed by the user running the HDFS service (by default, the user is hdfs).

    a. Capture the complete namespace of the file system. (The following command does a recursive listing of the root file system.)

       

       Important

       Make sure the namenode is started.

       ```
       su - hdfs -c "hdfs dfs -ls -R / > dfs-old-lsr-1.log"
       ```

Note

In secure mode you must have Kerberos credentials for the hdfs user.

b. Run the report command to create a list of DataNodes in the cluster.

```
su - hdfs -c "hdfs dfsadmin -report > dfs-old-report-l.log"
```

c. **Optional:** You can copy all or unrecoverable only data storelibext-customer directory in HDFS to a local file system or to a backup instance of HDFS.

d. **Optional:** You can also repeat the steps 3 (a) through 3 (c) and compare the results with the previous run to ensure the state of the file system remained unchanged.

12. Finalize any prior HDFS upgrade, if you have not done so already.

```
su - hdfs -c "hdfs dfsadmin -finalizeUpgrade"
```

Note

In secure mode, you must have Kerberos credentials for the hdfs user.

13. Stop remaining services (HDFS, ZooKeeper, and Ranger).

See Stopping HDP Services for more information.

Component	Command

HDFS	On all DataNotes:
	
	If you are running secure cluster, run following command as root:
	
	`/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop datanode`
	
	Else:
	
	`su - hdfs -c "usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop datanode"`
	
	If you are not running a highly available HDFS cluster, stop the Secondary NameNode by executing this command on the Secondary NameNode host machine:
	
	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop secondarynamenode"`
	
	On the NameNode host machine(s)
	
	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop namenode"`
	
	If you are running NameNode HA, stop the ZooKeeper Failover Controllers (ZKFC) by executing this command on the NameNode host machine:

Component	Command

	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-deamon.sh --config /etc/hadoop/conf stop zkfc"`
	
	If you are running NameNode HA, stop the JournalNodes by executing these command on the JournalNode host machines:
	
	`su - hdfs -c "/usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop journalnode"`

ZooKeeper Host machines	`su - -c zookeeper "/usr/hdp/current/zookeeper-server/bin/zookeeper-server stop"`

Ranger (XA Secure)	`service ranger-admin stop`
	
	`service ranger-usersync stop`

14. Back up your NameNode metadata.

Note

It's recommended to take a backup of the full `/hadoop.hdfs/namenode` path.

a. Copy the following checkpoint files into a backup directory.

The NameNode metadata is stored in a directory specified in the hdfs-site.xml configuration file under the configuration value "dfs.namenode.dir".

For example, if the configuration value is:

```
<property>
  <name>dfs.namenode.name.dir</name>
  <value>/hadoop/hdfs/namenode</value>
</property>
```

Then, the NameNode metadata files are all housed inside the directory `/hadooop.hdfs/namenode`.

b. Store the layoutVersion of the namenode.

`${dfs.namenode.name.dir}/current/VERSION`

15. Verify that edit logs in `${dfs.namenode.name.dir}/current/edits*` are empty.

a. Run the following comand on the active NameNode machine:

```
hdfs oev -i ${dfs.namenode.name.dir}/current/edits_inprogress_* -o edits.out
```

b. Verify the edits.out file. It should only have OP_START_LOG_SEGMENT transaction. For example:

```
<?xml version="1.0" encoding="UTF-8"?><EDITS>
<EDITS_VERSION>-56</EDITS_VERSION>
<RECORD>
```

```
<OPCODE>OP_START_LOG_SEGMENT</OPCODE>
<DATA>
<TXID>5749</TXID>
</DATA>
</RECORD>
```

c. If edits.out has transactions other than OP_START_LOG_SEGMENT, run the following steps and then verify edit logs are empty.

- Start the existing version NameNode.
- Ensure there is a new FS image file.
- Shut the NameNode down:

```
hdfs dfsadmin – saveNamespace
```

16. Rename or delete any paths that are reserved in the new version of HDFS.

When upgrading to a new version of HDFS, it is necessary to rename or delete any paths that are reserved in the new version of HDFS. If the NameNode encounters a reserved path during upgrade, it will print an error such as the following:

```
/.reserved is a reserved path and .snapshot is a reserved path
component in this version of HDFS.

Please rollback and delete or rename this path, or upgrade with the
-renameReserved key-value pairs option to automatically rename these
paths during upgrade.
```

Specifying `-upgrade -renameReserved` optional key-value pairs causes the NameNode to automatically rename any reserved paths found during startup.

For example, to rename all paths named `.snapshot` to `.my-snapshot` and change paths named `.reserved` to `.my-reserved`, specify `-upgrade -renameReserved .snapshot=.my-snapshot,.reserved=.my-reserved`.

If no key-value pairs are specified with `-renameReserved`, the NameNode will then suffix reserved paths with:

`.<LAYOUT-VERSION>.UPGRADE_RENAMED`

For example: `.snapshot.-51.UPGRADE_RENAMED`.

Note

We recommend that you perform a `-saveNamespace` before renaming paths (running `-saveNamespace` appears in a previous step in this procedure). This is because a data inconsistency can result if an edit log operation refers to the destination of an automatically renamed file.

Also note that running `-renameReserved` will rename all applicable existing files in the cluster. This may impact cluster applications.

17. If you are on JDK 1.6, upgrade the JDK on all nodes to JDK 1.7 before upgrading HDP. If you are running JDK 1.7, no action is required.

Important

If you want to upgrade to from JDK 1.7 to JDK 1.8, you must update the HDP stack before upgrading to JDK 1.8. For example, the high-level process should follow:

a. Run HDP 2.2 with JDK 1.7.

b. Perform the stack upgrade to HDP 2.4.3. See Upgrade HDP 2.2 Components.

c. Upgrade JDK from 1.7 to 1.8.

2.2. Upgrade HDP 2.2 Components

Important

See the Release Notes for the HDP 2.4.3.0 repo information.

The upgrade process to HDP 2.4.3 involves the following steps.

Select your OS:

RHEL/CentOS/Oracle 6

1. On all hosts, clean the yum repository.

   ```
   yum clean all
   ```

2. Remove your old HDP 2.2 components. This command uninstalls the HDP 2.2 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for RHEL/CentOS/Oracle6 table for the names of the files that need to be removed for each component and use the following format:

   ```
   yum erase "accumulo_$VERSION_$BUILD*" "atlas_metadata_$VERSION_$BUILD*"
    "datafu_$VERSION_$BUILD*" "falcon_$VERSION_$BUILD*" "flume_$VERSION_
   $BUILD*" "hadoop_$VERSION_$BUILD*" "hadoop_$VERSION_$BUILD-client*" "hadoop_
   $VERSION_$BUILD-hdfs*" "hadooplzo_$VERSION_$BUILD*" ...
   ```

   Where:

   - $VERSION is the HDP version number in the following format: 2_4_3_0
   - $HUE-VERSION is the Hue version number in the following format : 2.6.1.2.4.3.0
   - $BUILD is the HDP build number in the following format : 258

   So, for example, the file name will look like: accumulo-2_4_3_0_258*

Table 2.4. HDP Component Files for RHEL/CentOS/Oracle6

Component	Associated Files
Accumulo	accumulo_$VERSION_$BUILD*

Component	Associated Files
Atlas	atlas_metadata_$VERSION_$BUILD*
Datafu	datafu_$VERSION_$BUILD*
Falcon	falcon_$VERSION_$BUILD*
Flume	flume_$VERSION_$BUILD*
Hadoop (includes hadoop_client and hadoop_hdfs)	hadoop_$VERSION_$BUILD* hadoop_$VERSION_$BUILD-client* hadoop_$VERSION_$BUILD-hdfs* hadooplzo_$VERSION_$BUILD*
HBase	hbase_$VERSION_$BUILD*
Hive	hive_$VERSION_$BUILD* hive_hbase_$VERSION_$BUILD* hive_hcatalog_$VERSION_$BUILD* hive_jdbc_$VERSION_$BUILD* hive_metastore_$VERSION_$BUILD* hive_server_$VERSION_$BUILD* hive_webhcathive_hbase_$VERSION_$BUILD* ranger_hive_$VERSION_$BUILD*
Hue	hue_$HUE_VERSION_$BUILD* hue_beeswax_$HUE_VERSION_$BUILD* hue_common_$HUE_VERSION_$BUILD* hue_hcatalog_$HUE_VERSION_$BUILD* hue_oozie_$HUE_VERSION_$BUILD* hue_pig_$HUE_VERSION_$BUILD* hue_server_$HUE_VERSION_$BUILD*
Kafka	kafka_$VERSION_$BUILD*
Knox	knox_$VERSION_$BUILD*
Mahout	mahout_$VERSION_$BUILD*
MapReduce	hadoop_$VERSION_$BUILD-mapreduce*
Oozie	oozie_$VERSION_$BUILD*
Phoenix	phoenix_$VERSION_$BUILD
Pig	ranger_$VERSION_$BUILD*
Ranger	ranger_$VERSION_$BUILD*
Slider	slider_$VERSION_$BUILD*
Spark	spark_$VERSION_$BUILD*
Sqoop	sqoop_$VERSION_$BUILD*
Storm	storm_$VERSION_$BUILD*
Tez	tez_$VERSION_$BUILD*
YARN	hadoop_$VERSION_$BUILD-yarn*
ZooKeeper	zookeeper_$VERSION_$BUILD*

3. Validate that all HDP 2.2 component binaries are uninstalled:

```
yum list installed | grep @HDP2.2
```

4. Remove your old hdp.repo file:

   ```
   rm /etc/yum.repos.d/hdp.repo
   ```

5. Install the HDP 2.4.3 repo:

   - Download the hdp.repo file:

     ```
     wget -nv http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.4.
     3.0/hdp.repo -O /etc/yum.repos.d/hdp.repo
     ```

   - Confirm the HDP repository is configured.

     ```
     yum repolist
     ```

     You should see something like this. Verify that you have the HDP-2.4.3 directory:

     ```
     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 namestatus
     HDP-2.4.3.0 Hortonworks Data Platform Version - HDP-2.4.3.0
     ```

6. Install the HDP 2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.2 components:

   ```
   yum install "hadoop" "hadoop-hdfs" "hadoop-libhdfs" "hadoop-yarn" "hadoop-
   mapreduce" "hadoop-client" "hadoop-lzo" "openssl" "hive-webhcat" "hive-
   hcatalog" "oozie" "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase"
    "hue" "hive" "tez" "storm" "falcon" "flume" "phoenix" "accumulo" "mahout"
    "knox" "kafka" "spark" "ranger" "slider" "hdp_mon_nagios_addons"
   ```

   

   Note

   If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded.

   ```
   yum list installed | grep HDP-<old.stack.version.number>
   ```

   No component file names should appear in the returned list.

SLES 11 SP 1

1. On all hosts, clean the yum repository.

   ```
   zypper clean -all
   ```

2. Remove your old HDP 2.2 components. This command uninstalls the HDP 2.2 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for SLES11 SP1 table for the names of the files that need to be removed for each component and use the following format:

```
zypper rm "accumulo_$VERSION_$BUILD*" "datafu_$VERSION_$BUILD*" "falcon_
$VERSION_$BUILD*" "flume_$VERSION_$BUILD*" "hadoop_$VERSION_$BUILD*"
 "hadooplzo_$VERSION_$BUILD*" ...
```

The following table lists the files that need to be deleted for each of the HDP components. Variables are used to indicate:

- $VERSION - HDP version number in the following format: 2_4_3_0
- $HUE-VERSION - Hue version number in the following format: 2.6.1.2.3.0.0
- $BUILD - the HDP build number in the following format: 258

So, for example, the file name will look like: accumulo-2_4_3_0_258*

Table 2.5. HDP Component Files for SLES11 SP1

Component	Associated Files
Accumulo	accumulo_$VERSION_$BUILD*
Atlas	atlas_metadata_$VERSION_$BUILD*
Datafu	datafu_$VERSION_$BUILD*
Falcon	falcon_$VERSION_$BUILD*
Flume	flume_$VERSION_$BUILD*
Hadoop (includes hadoop_client and hadoop_hdfs)	hadoop_$VERSION_$BUILD* hadoop_$VERSION_$BUILD-client* hadoop_$VERSION_$BUILD-hdfs* hadooplzo_$VERSION_$BUILD*
HBase	hbase_$VERSION_$BUILD*
Hive	hive_$VERSION_$BUILD* hive_hbase_$VERSION_$BUILD* hive_hcatalog_$VERSION_$BUILD* hive_jdbc_$VERSION_$BUILD* hive_metastore_$VERSION_$BUILD* hive_server_$VERSION_$BUILD* hive_webhcathive_hbase_$VERSION_$BUILD* ranger_hive_$VERSION_$BUILD*
Hue	hue_$HUE_VERSION_$BUILD* hue_beeswax_$HUE_VERSION_$BUILD* hue_common_$HUE_VERSION_$BUILD* hue_hcatalog_$HUE_VERSION_$BUILD* hue_oozie_$HUE_VERSION_$BUILD* hue_pig_$HUE_VERSION_$BUILD* hue_server_$HUE_VERSION_$BUILD*
Kafka	kafka_$VERSION_$BUILD*

Component	Associated Files
Knox	knox_$VERSION_$BUILD*
Mahout	mahout_$VERSION_$BUILD*
MapReduce	hadoop_$VERSION_$BUILD-mapreduce*
Oozie	oozie_$VERSION_$BUILD*
Phoenix	phoenix_$VERSION_$BUILD
Pig	ranger_$VERSION_$BUILD*
Ranger	ranger_$VERSION_$BUILD*
Slider	slider_$VERSION_$BUILD*
Spark	spark_$VERSION_$BUILD*
Sqoop	sqoop_$VERSION_$BUILD*
Storm	storm_$VERSION_$BUILD*
Tez	tez_$VERSION_$BUILD*
YARN	hadoop_$VERSION_$BUILD-yarn*
ZooKeeper	zookeeper_$VERSION_$BUILD*

3. Validate that all HDP 2.2 component binaries are uninstalled:

```
yum list installed | grep @HDP2.2
```

4. Remove your old hdp.repo file:

```
rm /etc/zypp/repos.d/hdp.repo
```

5. Download the HDP 2.4.3 hdp.repo file:

```
wget -nv http://public-repo-1.hortonworks.com/HDP/sles11sp1/2.x/updates/2.4.
3.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
```

6. Install the HDP 2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.2 components:

```
zypper install "hadoop" "hadoop-hdfs" "hadoop-libhdfs" "hadoop-yarn"
 "hadoop-mapreduce" "hadoop-client" "openssl" "hive-webhcat" "hive-hcatalog"
 "oozie" "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hive" "hue"
 "tez" "storm" "falcon" "flume" "phoenix" "accumulo" "mahout" "knox" "kafka"
 "spark" "slider" "hdp_mon_nagios_addons"
```

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

```
zypper up -r HDP-2.4
```

```
zypper install oozie-client
```

Note

If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded. For example, to verify hdfs, hive, and hcatalog:

```
rpm -qa | grep hdfs && rpm -qa | grep hive && rpm -qa | grep
hcatalog
```

No component files names should appear in the returned list.

SLES 11 SP3/SP4

1. On all hosts, clean the zypper repository.

   ```
   zypper clean -a
   ```

2. Remove your old HDP 2.2 components. This command uninstalls the HDP 2.2 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for SLES11 SP3/SP4 table for the names of the files that need to be removed for each component and use the following format:

   ```
   zypper rm "accumulo-$VERSION-$BUILD*" "datafu-$VERSION-$BUILD*" "falcon-
   $VERSION-$BUILD*" "flume-$VERSION-$BUILD*" "hadoop-$VERSION-$BUILD*"
    "hadooplzo-$VERSION-$BUILD*" ...
   ```

   The following table lists the files that need to be deleted for each of the HDP components. Variables are used to indicate:

   - $VERSION - HDP version number in the following format: 2_4_3_0
   - $HUE-VERSION - Hue version number in the following format : 2.6.1.2.3.0.0
   - $BUILD - the HDP build number in the following format : 258

   So, for example, the file name will look like: accumulo-2_4_3_0_258*

Table 2.6. HDP Component Files for SLES11 SP3/SP4

Component	Associated Files
Accumulo	accumulo_$VERSION_$BUILD*
Atlas	atlas_metadata_$VERSION_$BUILD*
Datafu	datafu_$VERSION_$BUILD*
Falcon	falcon_$VERSION_$BUILD*
Flume	flume_$VERSION_$BUILD*
Hadoop (includes hadoop_client and hadoop_hdfs)	hadoop_$VERSION_$BUILD* hadoop_$VERSION_$BUILD-client* hadoop_$VERSION_$BUILD-hdfs* hadooplzo_$VERSION_$BUILD*
HBase	hbase_$VERSION_$BUILD*
Hive	hive_$VERSION_$BUILD* hive_hbase_$VERSION_$BUILD* hive_hcatalog_$VERSION_$BUILD* hive_jdbc_$VERSION_$BUILD* hive_metastore_$VERSION_$BUILD*

Component	Associated Files
	hive_server_$VERSION_$BUILD* hive_webhcathive_hbase_$VERSION_$BUILD* ranger_hive_$VERSION_$BUILD*
Hue	hue_$HUE_VERSION_$BUILD* hue_beeswax_$HUE_VERSION_$BUILD* hue_common_$HUE_VERSION_$BUILD* hue_hcatalog_$HUE_VERSION_$BUILD* hue_oozie_$HUE_VERSION_$BUILD* hue_pig_$HUE_VERSION_$BUILD* hue_server_$HUE_VERSION_$BUILD*
Kafka	kafka_$VERSION_$BUILD*
Knox	knox_$VERSION_$BUILD*
Mahout	mahout_$VERSION_$BUILD*
MapReduce	hadoop_$VERSION_$BUILD-mapreduce*
Oozie	oozie_$VERSION_$BUILD*
Phoenix	phoenix_$VERSION_$BUILD
Pig	ranger_$VERSION_$BUILD*
Ranger	ranger_$VERSION_$BUILD*
Slider	slider_$VERSION_$BUILD*
Spark	spark_$VERSION_$BUILD*
Sqoop	sqoop_$VERSION_$BUILD*
Storm	storm_$VERSION_$BUILD*
Tez	tez_$VERSION_$BUILD*
YARN	hadoop_$VERSION_$BUILD-yarn*
ZooKeeper	zookeeper_$VERSION_$BUILD*

3. Validate that all HDP 2.2 component binaries are uninstalled:

   ```
   zypper search --installed-only --repo HDP-2.2.6.0
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/zypp/repos.d/hdp.repo
   ```

5. Download the HDP 2.4.3.0 hdp.repo file:

   ```
   http://public-repo-1.hortonworks.com/HDP/suse11sp3/2.x/
   updates/2.4.3.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
   ```

6. Install the HDP 2.4.3.0 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.2 components:

   ```
   zypper install "hadoop" "hadoop-hdfs" "hadoop-libhdfs" "hadoop-yarn"
    "hadoop-mapreduce" "hadoop-client" "openssl" "oozie" "collectd" "gccxml"
    "pig" "sqoop" "zookeeper" "hbase" "hue" "hive" "tez" "storm" "falcon"
    "flume" "phoenix" "accumulo" "mahout" "knox" "ranger" "kafka" "spark"
   ```

```
 "spark-python" "hdp_mon_nagios_addons" "slider" "hive-webcat" "hive-
hcatalog""
```

```
zypper up -r HDP-2.4
```

```
zypper install oozie-client
```

Note

If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded. For example, to verify hdfs, hive, and hcatalog:

   ```
   rpm -qa | grep hdfs, && rpm -qa | grep hive && rpm -qa | grep
   hcatalog
   ```

   No component files names should appear in the returned list.

Ubuntu 12

1. On all hosts, clean the apt-get repository.

   ```
   apt-get clean -&-all
   ```

2. Remove your old HDP 2.2 components. This command uninstalls the HDP 2.2 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for Ubuntu 12 table for the names of the files that need to be removed for each component and use the following format:

   ```
   apt-get remove "accumulo-$VERSION-$BUILD*" "datafu-$VERSION-$BUILD*"
    "falcon-$VERSION-$BUILD*" "flume-$VERSION-$BUILD*" "hadoop-$VERSION-
   $BUILD*" "hadooplzo-$VERSION-$BUILD*" ...
   ```

   The following table lists the files that need to be deleted for each of the HDP components. Variables are used to indicate:

   - $VERSION - HDP version in the following format: 2-4-2-0
   - $HUE-VERSION - Hue version in the following format : 2.6.1.2.3.0.0
   - $BUILD - version of the HDP build in the following format : 258

   So, for example, the file name will look like: accumulo-2-4-2-0-258*

Table 2.7. HDP Component Files for Ubuntu 12

Component	HDP Files to Re-install for Upgrade
Accumulo	accumulo-$VERSION-$BUILD*
Atlas	atlas-metadata-$VERSION-$BUILD*
Datafu	datafu-$VERSION-$BUILD*
Falcon	falcon-$VERSION-$BUILD*
Flume	flume-$VERSION-$BUILD*
Hadoop (includes hadoop-client and hadoop-hdfs)	hadoop-$VERSION-$BUILD*

Component	HDP Files to Re-install for Upgrade
	hadoop-$VERSION-$BUILD-client* hadoop-$VERSION-$BUILD-hdfs* hadooplzo-$VERSION-$BUILD*
HBase	hbase-$VERSION-$BUILD*
Hive	hive-$VERSION-$BUILD* hive-hbase-$VERSION-$BUILD* hive-hcatalog-$VERSION-$BUILD* hive-jdbc-$VERSION-$BUILD* hive-metastore-$VERSION-$BUILD* hive-server-$VERSION-$BUILD* hive-webhcathive-hbase-$VERSION-$BUILD* ranger-hive-$VERSION-$BUILD*
Hue	hue-$HUE-VERSION-$BUILD* hue-beeswax-$HUE-VERSION-$BUILD* hue-common-$HUE-VERSION-$BUILD* hue-hcatalog-$HUE-VERSION-$BUILD* hue-oozie-$HUE-VERSION-$BUILD* hue-pig-$HUE-VERSION-$BUILD* hue-server-$HUE-VERSION-$BUILD*
Kafka	kafka-$VERSION-$BUILD*
Knox	knox-$VERSION-$BUILD*
Mahout	mahout-$VERSION-$BUILD*
MapReduce	hadoop-$VERSION-$BUILD-mapreduce*
Oozie	oozie-$VERSION-$BUILD*
Phoenix	phoenix-$VERSION-$BUILD
Pig	ranger-$VERSION-$BUILD*
Ranger	ranger-$VERSION-$BUILD*
Slider	slider-$VERSION-$BUILD*
Spark	spark-$VERSION-$BUILD*
Sqoop	sqoop-$VERSION-$BUILD*
Storm	storm-$VERSION-$BUILD*
Tez	tez-$VERSION-$BUILD*
YARN	hadoop-$VERSION-$BUILD-yarn*
ZooKeeper	zookeeper-$VERSION-$BUILD*

3. Validate that all HDP 2.2 component binaries are uninstalled:

```
yum list installed | grep @HDP2.2
```

4. Remove your old hdp.repo file:

```
rm /etc/apt/sources.list.d/hdp.list
```

5. Download the HDP 2.4.3.0 hdp.repo file:

   ```
   wget http://public-repo-1.hortonworks.com/HDP/ubuntu12/2.x/
   updates/2.4.3.0/hdp.list - O /etc/apt/sources.list.d/hdp.list
   ```

6. Run an update:

   ```
   apt-get update
   ```

7. Install the HDP 2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.2 components:

   ```
   apt-get install "hadoop" "hadoop-lzo" "hadoop-hdfs" "libhdfs0" "hadoop-yarn"
    "hadoop-mapreduce" "hadoop-client" "openssl" "webhcat" "hcatalog" "oozie"
    "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hive" "tez" "storm"
    "falcon" "flume" "phoenix" "accumulo" "mahout" "knox" "spark" "slider*"
    "hdp_mon_nagios_addons"
   ```

   

   Note

   If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

Debian 6 (Deprecated)

HDP support for Debian 6 is deprecated with HDP 2.4.3. Future versions of HDP will no longer be supported on Debian 6.

1. On all hosts, clean the apt-get repository.

   ```
   apt-get clean
   ```

2. Remove your old HDP 2.2 components. This command uninstalls the HDP 2.2 components. It leaves the user data, and metadata, but removes your configurations. Refer to the HDP Component Files for Debian 6 (Deprecated)/Debian 7 table for the names of the files that need to be removed for each component and use the following format:

   ```
   apt-get remove "accumulo-$VERSION-$BUILD*" "datafu-$VERSION-$BUILD*"
    "falcon-$VERSION-$BUILD*" "flume-$VERSION-$BUILD*" "hadoop-$VERSION-
   $BUILD*" "hadooplzo-$VERSION-$BUILD*" ...
   ```

   The following table lists the files that need to be deleted for each of the HDP components. Variables are used to indicate:

   - $VERSION - HDP version in the following format: 2-4-2-0
   - $HUE-VERSION - Hue version in the following format : 2.6.1.2.3.0.0
   - $BUILD - version of the HDP build in the following format : 258

   So, for example, the file name will look like: accumulo-2-4-2-0-258*

Table 2.8. HDP Component Files for Debian 6 (Deprecated)/Debian 7

Component	HDP Files to Re-install for Upgrade
Accumulo	accumulo-$VERSION-$BUILD*
Atlas	atlas-metadata-$VERSION-$BUILD*
Datafu	datafu-$VERSION-$BUILD*
Falcon	falcon-$VERSION-$BUILD*
Flume	flume-$VERSION-$BUILD*
Hadoop (includes hadoop-client and hadoop-hdfs)	hadoop-$VERSION-$BUILD* hadoop-$VERSION-$BUILD-client* hadoop-$VERSION-$BUILD-hdfs* hadooplzo-$VERSION-$BUILD*
HBase	hbase-$VERSION-$BUILD*
Hive	hive-$VERSION-$BUILD* hive-hbase-$VERSION-$BUILD* hive-hcatalog-$VERSION-$BUILD* hive-jdbc-$VERSION-$BUILD* hive-metastore-$VERSION-$BUILD* hive-server-$VERSION-$BUILD* hive-webhcathive-hbase-$VERSION-$BUILD* ranger-hive-$VERSION-$BUILD*
Hue	hue-$HUE-VERSION-$BUILD* hue-beeswax-$HUE-VERSION-$BUILD* hue-common-$HUE-VERSION-$BUILD* hue-hcatalog-$HUE-VERSION-$BUILD* hue-oozie-$HUE-VERSION-$BUILD* hue-pig-$HUE-VERSION-$BUILD* hue-server-$HUE-VERSION-$BUILD*
Kafka	kafka-$VERSION-$BUILD*
Knox	knox-$VERSION-$BUILD*
Mahout	mahout-$VERSION-$BUILD*
MapReduce	hadoop-$VERSION-$BUILD-mapreduce*
Oozie	oozie-$VERSION-$BUILD*
Phoenix	phoenix-$VERSION-$BUILD
Pig	ranger-$VERSION-$BUILD*
Ranger	ranger-$VERSION-$BUILD*
Slider	slider-$VERSION-$BUILD*
Spark	spark-$VERSION-$BUILD*
Sqoop	sqoop-$VERSION-$BUILD*
Storm	storm-$VERSION-$BUILD*
Tez	tez-$VERSION-$BUILD*

Component	HDP Files to Re-install for Upgrade
YARN	hadoop-$VERSION-$BUILD-yarn*
ZooKeeper	zookeeper-$VERSION-$BUILD*

3. Validate that all HDP 2.2 component binaries are uninstalled:

```
dpkg -l | grep "^ii" | grep hadoop
```

4. Remove your old hdp.repo file:

```
rm /etc/apt/sources.list.d/hdp.list
```

5. Download the HDP 2.4.3 hdp.repo file:

```
wget -nv http://public-repo-1.hortonworks.com/HDP/debian6/2.x/
updates/2.4.3.0/hdp.list - O /etc/apt/sources.list.d/hdp.list
```

6. Run an update:

```
apt-get update
```

7. Install the HDP 2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.2 components:

```
apt-get install "hadoop" "hive-webhcat" "hadoop-hdfs" "libhdfs0" "hadoop-
yarn" "hadoop-mapreduce" "hadoop-client" "openssl" "hcatalog" "oozie"
 "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hive" "tez" "storm"
 "falcon" "flume" "phoenix" "accumulo" "mahout" "knox" "spark" "slider*"
 "hdp_mon_nagios_addons"
```

Note

If you installed Apache Ranger, see Upgrade Ranger for more information on the upgrade path.

2.3. Symlink Directories with hdp-select

Warning

HDP 2.4.3 installs hdp-select automatically with the installation or upgrade of the first HDP component. If you have not already upgraded ZooKeeper, hdp-select has not been installed.

To prevent version-specific directory issues for your scripts and updates, Hortonworks provides hdp-select, a script that symlinks directories to hdp-current and modifies paths for configuration directories.

1. Before you run hdp-select, remove one link:

```
rm /usr/bin/oozie
```

2. Run hdp-select set all on your NameNode and all your DataNodes:

```
hdp-select set all 2.4.3.0-$BUILD
```

For example:

```
/usr/bin/hdp-select set all 2.4.3.0-$BUILD
```

2.4. Configure and Start Apache ZooKeeper

Before you can upgrade Apache ZooKeeper, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Tip

If you are running a highly available HDFS cluster, configure and restart ZooKeeper **before** you upgrade HDFS. This best practice lets the upgraded ZKFC work with your primary NameNode and your Standby NameNode.

1. Replace your configuration after upgrading on all the ZooKeeper nodes. Replace the ZooKeeper template configuration in `/etc/zookeeper/conf`.

2. Start ZooKeeper.

   On all ZooKeeper server host machines, run the following command to start ZooKeeper and the ZKFC:

   ```
   su - zookeeper -c "/usr/hdp/current/zookeeper-server/bin/zookeeper-server
    start"
   ```

2.5. Configure Hadoop

RHEL/CentOS/Oracle Linux

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml.

2. Paths have changed in HDP 2.2. Make sure you remove old path specifications from hadoop-env.sh, such as:

   ```
   export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-
   amd64-64
   ```

   If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

SLES

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml.

2. Paths have changed since HDP 2.2. Make sure you remove old path specifications from hadoop-env.sh, such as:

```
export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-
amd64-64
```

If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

Ubuntu/Debian

HDP support for Debian 6 is deprecated with HDP 2.4.3. Future versions of HDP will no longer be supported on Debian 6.

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml
2. Paths have changed in HDP 2.4.3. Make sure you remove old path specifications from hadoop-env.sh, such as:

   ```
   export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-
   amd64-64
   ```

   If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

2.6. Start Hadoop Core

Warning

Before you start HDFS on a highly available HDFS cluster, you must start the ZooKeeper service. If you do not start the ZKFC, there can be failures.

To start HDFS, run commands as the $HDFS_USER.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

1. Replace your configuration after upgrading on all the HDFS nodes. Replace the HDFS template configuration in `/etc/hdfs/conf`.
2. If you are upgrading from a highly available HDFS cluster configuration, start all JournalNodes. On each JournalNode host, run the following commands:

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

Important

All JournalNodes must be running when performing the upgrade, rollback, or finalization operations. If any JournalNodes are down when running any such operation, the operation fails.

3. If you are running HDFS on a highly available namenode, you must first start the ZooKeeper service

   

   Note

   Perform this step only if you are on a highly available HDFS cluster.

   Run this command on all NameNode hosts.

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

4. Start the NameNode.

   Because the file system version has now changed you must start the NameNode manually.

   On the active NameNode host, run the following commands:

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

   On a large system, this can take a long time to complete.

   

   Note

   Run this command with the -upgrade option only once. After you have completed this step, you can bring up the NameNode using this command without including the -upgrade option.

   

   Note

   If you receive the error:

   ```
   Failed to find Premain-Class manifest attribute in
        /usr/hdp/<HDP-version>/hadoop/lib/ranger-hdfs-plugin-
   shim-0.5.0.***.jar
   Error occurred during initialization of VM
   agent library failed to init: instrument
   ```

   after an upgrade to HDP-2.4.3, remove set-hdfs-plugin-env.sh from the /usr/hdp/<hdp-version>/hadoop/conf/ directory.

   To check if the Upgrade is in progress, check that the "\previous" directory has been created in the \NameNode and \JournalNode directories. The "\previous" directory contains a snapshot of the data before upgrade.

   In a highly available HDFS cluster configuration, this NameNode will not enter the standby state as usual. Rather, this NameNode will immediately enter the active state, perform an upgrade of its local storage directories, and also perform an upgrade of the shared edit log. At this point, the standby NameNode in the HA pair is still down. It will be out of sync with the upgraded active NameNode.

To synchronize the active and standby NameNode, re-establishing HA, re-bootstrap the standby NameNode by running the NameNode with the '-bootstrapStandby' flag. Do NOT start this standby NameNode with the '-upgrade' flag.

```
su - hdfs -c "hdfs namenode -bootstrapStandby -force"
```

The bootstrapStandby command will download the most recent fsimage from the active NameNode into the $dfs.name.dir directory of the standby NameNode. You can enter that directory to make sure the fsimage has been successfully downloaded. After verifying, start the ZKFailoverController, then start the standby NameNode. You can check the status of both NameNodes using the Web UI.

5. Verify that the NameNode is up and running:

   ```
   ps -ef|grep -i NameNode
   ```

6. If you do not have a highly available HDFS cluster configuration (non_HA namenode), start the Secondary NameNode.

   

   Note

   Do not perform this step if you have a highly available HDFS cluster configuration.

   On the Secondary NameNode host machine, run the following commands:

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

7. Verify that the Secondary NameNode is up and running.

   

   Note

   Do not perform this step if you have a highly available HDFS cluster environment.

   ```
   ps -ef|grep SecondaryNameNode
   ```

8. Start DataNodes.

   On each of the DataNodes, enter the following command. Note: If you are working on a `non-secure` DataNode, use $HDFS_USER. For a secure DataNode, use root.

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

9. Verify that the DataNode process is up and running:

   ```
   ps -ef|grep DataNode
   ```

10. Verify that NameNode can go out of safe mode.

    ```
    >su - hdfs -c "hdfs dfsadmin -safemode wait"
    ```

You should see the following result: `Safe mode is OFF`

In general, it takes 5-10 minutes to get out of safemode. For thousands of nodes with millions of data blocks, getting out of safemode can take up to 45 minutes.

2.7. Verify HDFS Filesystem Health

Analyze if the filesystem is healthy.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Important

If you have a secure server, you will need Kerberos credentials for hdfs user access.

1. Run the fsck command on namenode as $HDFS_USER:

   ```
   su - hdfs -c "hdfs fsck / -files -blocks -locations > dfs-new-fsck-1.log"
   ```

   Open dfs-new-fsck-1.log to see that the filesystem under path / is HEALTHY.

2. Run `hdfs` namespace and report.

   a. List directories.

      ```
      su - hdfs -c "hdfs dfs -ls -R / > dfs-new-lsr-1.log"
      ```

   b. Open the `dfs-new-lsr-l.log` and confirm that you can see the file and directory listing in the namespace.

   c. Run report command to create a list of DataNodes in the cluster.

      ```
      su - hdfs -c "hdfs dfsadmin -report > dfs-new-report-1.log"
      ```

   d. Open the `dfs-new-report` file and validate the admin report.

3. Compare the namespace report before the upgrade and after the upgrade. Verify that user files exist after upgrade.

   The file names are listed below:

   ```
   dfs-old-fsck-1.log < -- > dfs-new-fsck-1.log
   ```

   ```
   dfs-old-lsr-1.log < -- > dfs-new-lsr-1.log
   ```

Note

You must do this comparison manually to catch all errors.

4. From the NameNode WebUI, determine if all DataNodes are up and running.

   ```
   http://<namenode>:<namenodeport>
   ```

5. If you are on a highly available HDFS cluster, go to the StandbyNameNode web UI to see if all DataNodes are up and running:

   ```
   http://<standbynamenode>:<namenodeport>
   ```

6. If you are **not** on a highly available HDFS cluster, go to the SecondaryNameNode web UI to see if it the secondary node is up and running:

   ```
   http://<secondarynamenode>:<secondarynamenodeport>
   ```

7. Verify that read and write to hdfs works successfully.

   ```
   hdfs dfs -put [input file] [output file]

   hdfs dfs -cat [output file]
   ```

2.8. Configure YARN and MapReduce

After you upgrade Hadoop, complete the following steps to update your configs.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Important

In secure mode, you must have Kerberos credentials for the hdfs user.

1. Upload the MapReduce tarball to HDFS. As the HDFS user, for example 'hdfs':

   ```
   su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/
   mapreduce/"

   su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/hadoop/
   mapreduce.tar.gz /hdp/apps/2.4.3.0-$BUILD/mapreduce/"

   su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp"

   su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/
   mapreduce"
   ```

```
su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/
mapreduce/mapreduce.tar.gz"
```

2. Make sure that the following properties are in `/etc/hadoop/conf/mapred-site.xml`:

   - Make sure mapreduce.application.framework.path exists in mapred-site.xml:

```
<property>
 <name>mapreduce.application.framework.path</name>
 <value>/hdp/apps/${hdp.version}/mapreduce/mapreduce.tar.gz#mr-framework</
value>
</property>

<property>
 <name>yarn.app.mapreduce.am.admin-command-opts</name>
 <value>-Dhdp.version=${hdp.version}</value>
</property>
```

Note

You do not need to modify ${hdp.version}.

   - Modify the following existing properties to include ${hdp.version}:

```
<property>
 <name>mapreduce.admin.user.env</name>
 <value>LD_LIBRARY_PATH=/usr/hdp/${hdp.version}/hadoop/lib/native:/usr/
hdp/${hdp.version}/hadoop/
     lib/native/Linux-amd64-64</value>
</property>

<property>
 <name>mapreduce.admin.map.child.java.opts</name>
 <value>-server -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.
version}</value>
 <final>true</final>
</property>

<property>
 <name>mapreduce.admin.reduce.child.java.opts</name>
 <value>-server -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.
version}</value>
 <final>true</final>
</property>

<property>
 <name>mapreduce.application.classpath</name>
 <value>$PWD/mr-framework/hadoop/share/hadoop/mapreduce/*,
   $PWD/mr-framework/hadoop/share/hadoop/mapreduce/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/common/*,
   $PWD/mr-framework/hadoop/share/hadoop/common/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/yarn/*,
   $PWD/mr-framework/hadoop/share/hadoop/yarn/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/hdfs/*,
   $PWD/mr-framework/hadoop/share/hadoop/hdfs/lib/*,
  /usr/hdp/${hdp.version}/hadoop/lib/hadoop-lzo-0.6.0.${hdp.version}.jar,
  /etc/hadoop/conf/secure</value>
```

```
</property>
```

Note

You do not need to modify ${hdp.version}.

Note

If you are planning to use Spark in yarn-client mode, make Spark work in yarn-client mode 2.4.3.0-$BUILD.

3. Make sure the following property is in `/etc/hadoop/conf/yarn-site.xml`:

```
<property>
 <name>yarn.application.classpath</name>
 <value>$HADOOP_CONF_DIR,/usr/hdp/${hdp.version}/hadoop-client/*,
  /usr/hdp/${hdp.version}/hadoop-client/lib/*,
  /usr/hdp/${hdp.version}/hadoop-hdfs-client/*,
  /usr/hdp/${hdp.version}/hadoop-hdfs-client/lib/*,
  /usr/hdp/${hdp.version}/hadoop-yarn-client/*,
  /usr/hdp/${hdp.version}/hadoop-yarn-client/lib/*</value>
</property>
```

4. On secure clusters only, add the following properties to `/etc/hadoop/conf/yarn-site.xml`:

```
 <property>
  <name>yarn.timeline-service.recovery.enabled</name>
  <value>TRUE</value>
  </property>

  <property>
<name>yarn.timeline-service.state-store.class</name>
<value>org.apache.hadoop.yarn.server.timeline.recovery.
LeveldbTimelineStateStore</value>
</property>

  <property>
  <name>yarn.timeline-service.leveldb-state-store.path</name>
  <value><the same as the default of "yarn.timeline-service-leveldb-
timeline-store.path</value>
  </property>
```

5. For secure clusters, you must create and configure the container-executor.cfg configuration file:

   - Create the container-executor.cfg file in /etc/hadoop/conf/container-executor.cfg

   - Insert the following properties:

```
yarn.nodemanager.linux-container-executor.group=hadoop
banned.users=hdfs,yarn,mapred
min.user.id=1000
```

     - yarn.nodemanager.linux-container-executor.group - Configured value of yarn.nodemanager.linux-container-executor.group. This must match the value of yarn.nodemanager.linux-container-executor.group in yarn-site.xml.

- banned.users - Comma-separated list of users who can not run container-executor.
- min.user.id - Minimum value of user id. This prevents system users from running container-executor.
- allowed.system.users - Comma-separated list of allowed system users.

- Set the file /etc/hadoop/conf/container-executor.cfg file permissions to only be readable by root:

```
chown root:hadoop /etc/hadoop/conf/container-executor.cfg
chmod 400 /etc/hadoop/conf/container-executor.cfg
```

- Set the container-executor program so that only root or hadoop group users can run it:

```
chown root:hadoop /usr/hdp/${hdp.version}/hadoop-yarn/bin/container-
executor

chmod 6050 /usr/hdp/${hdp.version}/hadoop-yarn/bin/container-executor
```

2.9. Start YARN/MapReduce Services

To start YARN, run commands as a YARN user. To start MapReduce, run commands as a MapReduce user.

Note

The su commands in this section use "yarn" to represent the YARN Service user and mapreduce to represent the MAPREDUCE Service user. If you are using another name for these Service users, you will need to substitute your Service user name for "yarn" or "mapreduce" in each of the su commands.

1. Manually clear the ResourceManager state store.

```
su - yarn -c "yarn resourcemanager -format-state-store"
```

2. Start the ResourceManager on all your ResourceManager hosts.

```
su - yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.
sh start resourcemanager"

ps -ef | grep -i resourcemanager
```

3. Start the TimelineServer on your TimelineServer host.

```
su - yarn -c "/usr/hdp/current/hadoop-yarn-timelineserver/sbin/yarn-daemon.
sh start timelineserver"

ps -ef | grep -i timelineserver
```

4. Start the NodeManager on all your NodeManager hosts.

```
su - yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh
 start nodemanager"
```

```
ps -ef | grep -i nodemanager
```

5. To start MapReduce, run the following commands:

```
su - mapred -c "/usr/hdp/current/hadoop-mapreduce-historyserver/sbin/mr-
jobhistory-daemon.sh start historyserver"

ps -ef | grep -i jobhistoryserver
```

2.10. Run Hadoop Smoke Tests

To smoke test your Hadoop upgrade, you can run the following MapReduce job as a regular user.

The job uses MapReduce to write 100MB of data into HDFS with RandomWriter

```
hadoop jar /usr/hdp/current/hadoop-mapreduce-client/hadoop-mapreduce-examples.
jar
            randomwriter -Dtest.randomwrite.total_bytes=10000000 test-after-
upgrade.
```

You should see messages similar to:

```
map 0% reduce 0%
…map 100% reduce 100%
Job …. completed successfully
```

MapReduce upgraded successfully. You can now upgrade your other components.

Basic Troubleshooting

To find the number of active nodes and NodeManagers, access the ResourceManager web UI:

`http://<resource manager host>:8088/cluster/nodes`

The number of active nodes should be equal to the number of nodemanagers.

Accessing error messages:

1. Access the ApplicationMaster WebUI to view the container logs.
2. At your console logs for MapReduce job, look for a line with this format:

   13/10/02 17:57:21 INFO mapreduce.Job: The url to track the job: http://<resource manager host>:8088/proxy/application_1380673658357_0007/

3. Select the logs link under ApplicationMaster table. It will redirect you to the container logs. Error messages display here.

2.11. Configure and Start Apache HBase

Before you can upgrade Apache HBase, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "hbase" to represent the HBASE Service user. If you are using another name for your HBASE Service user, you will need to substitute your HBASE Service user name for "hbase" in each of the `su` commands.

The hbase.bucketcache.percentage.in.combinedcache is removed in HDP 2.4.3. To enable Kerberos for clusters with dual home network setting, each HBase RegionServer must have its own key. See Installing HDP Manually. This simplifies the configuration of block cache. BucketCache configurations from HDP 2.2 will need to be recalculated to attain identical memory allotments in HDP 2.4.3. The L1 LruBlockCache will be whatever hfile.block.cache.size is set to and the L2 BucketCache will be whatever hbase.bucketcache.size is set to.

1. Replace your configuration after upgrading. Replace the HBase template configuration in `/etc/hbase/conf`.
2. Start services. From root, assuming that $HBASE_USER=hbase:

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

   ```
   su - hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-
   daemon.sh start regionserver"
   ```

3. Check processes.

   ```
   ps -ef | grep -i hmaster
   ```

   ```
   ps -ef | grep -i hregion
   ```

2.12. Configure Apache Phoenix

Before you can upgrade Apache Phoenix, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

To configure Phoenix, complete the following steps:

1. Add the following property to the `/etc/hbase/hbase-site.xml` file on all HBase nodes, the MasterServer, and all RegionServers to prevent deadlocks from occurring during maintenance on global indexes:

   ```
   <property>
     <name>hbase.regionserver.wal.codec</name>
     <value>org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec</
   value>
   </property>
   ```

2. To enable user-defined functions, configure the following property in `/etc/hbase/conf` on all Hbase nodes.

```
<property>
 <name>phoenix.functions.allowUserDefinedFunctions</name>
 <value>true</value>
 <description>enable UDF functions</description>
</property>
```

3. Ensure the client side `hbase-site.xml` matches the server side configuration.
4. If the folder specified in `hbase.tmp.dir` property on `hbase-site.xml` does not exist, create that directory with adequate permissions.
5. Set the following porperty in the hbase-site.xml file for all RegionServers, but not on the client side:

```
<property>
  <name>hbase.rpc.controllerfactory.class</name>
  <value>org.apache.hadoop.hbase.ipc.controller.ServerRpcControllerFactory</
value>
</property>
```

6. Restart the HBase Master and RegionServers.

Configuring Phoenix to Run in a Secure Cluster

Perform the following additional steps to configure Phoenix to run in a secure Hadoop cluster:

1. To link the HBase configuration file with the Phoenix libraries:

   ```
   ln -sf HBASE_CONFIG_DIR/hbase-site.xml PHOENIX_HOME/bin/hbase-
   site.xml
   ```

2. To link the Hadoop configuration file with the Phoenix libraries:

   ```
   ln -sf HADOOP_CONFIG_DIR/core-site.xml PHOENIX_HOME/bin/core-
   site.xml
   ```

Note

When running the pssql.py and sqlline.py Phoenix scripts in secure mode, you can safely ignore the following warnings.

```
14/04/19 00:56:24 WARN util.NativeCodeLoader:
Unable to load native-hadoop library for your platform...
  using builtin-java classes where applicable

14/04/19 00:56:24 WARN util.DynamicClassLoader: Failed to identify the fs of
dir hdfs://<HOSTNAME>:8020/apps/hbase/data/lib, ignored java.io.IOException:
No FileSystem for scheme: hdfs
```

2.13. Configure and Start Apache Accumulo

Before you can upgrade Apache Accumulo, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these

steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The su commands in this section use "accumulo" to represent the Accumulo Service user. If you are using another name for your Accumulo Service user, you will need to substitute your Accumulo Service user name for "accumulo" in each of the su commands.

1. You must replace your configuration after upgrading. Copy /etc/accumulo/conf from the template to the conf directory in Accumulo hosts.
2. Start the services:

```
su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` master"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` tserver"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` gc"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` tracer"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` monitor"
```

3. Check that the processes are running

   ```
   ps -ef | grep accumulo
   ```

   or visit http://<hostname>:50095 in your browser

2.14. Configure and Start Apache Tez

Before you can upgrade Apache Tez, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The su commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the su commands.

To upgrade Tez:

1. Copy your previously backed-up copy of tez-site.xml into the /etc/tez/conf directory.
2. Upload the Tez tarball to HDFS.

```
su - hdfs
hdfs dfs -mkdir -p /hdp/apps/<hdp_version>/tez/
hdfs dfs -put /usr/hdp/<hdp_version>/tez/lib/tez.tar.gz /hdp/apps/
<hdp_version>/tez/
hdfs dfs -chown -R hdfs:hadoop /hdp
hdfs dfs -chmod -R 555 /hdp/apps/<hdp_version>/tez
hdfs dfs -chmod -R 444 /hdp/apps/<hdp_version>/tez/tez.tar.gz
```

Where <hdp_version> is the current HDP version, for example 2.4.3.0-258.

3. Edit the tez.lib.uris property in the tez-site.xml file to point to /hdp/apps/<hdp_version>/tez/tez.tar.gz

```
...
<property>
 <name>tez.lib.uris</name>
 <value>/hdp/apps/<hdp_version>/tez/tez.tar.gz</value>
</property>
...
```

Where <hdp_version> is the current HDP version, for example 2.4.3.0-258.

4. **Optional** Earlier releases of Tez did not have access control. In the current version of Tez, the default behavior restricts the ability to view the Tez history to only the owner of the job. To retain unrestricted access for non-secure clusters, set tez.am.view-acls set to "*".

5. Change the value of the tez.tez-ui.history-url.base property to the url for the upgraded Tez View. For information on setting up the Tez view, see Deploying the Tez View in the HDP Ambari Views Guide.

2.15. Configure and Start Apache Hive and Apache HCatalog

Before you can upgrade Apache Hive and Apache HCatalog, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The su commands in this section use "hive" to represent the Hive Service user. If you are using another name for your Hive Service user, you will need to substitute your Hive Service user name for "hive" in each of the su commands.

Important

When using HiveServer2 in HTTP mode, you must configure the mapping from Kerberos Principals to short names in the “hadoop.security.auth_*to*_local" property setting in the core-site.xml file.

1. Prior to starting the upgrade process, set the following in your hive configuration file:

```
datanucleus.autoCreateSchema=false
```

2. Copy the jdbc connector jar from OLD_HIVE_HOME/lib to CURRENT_HIVE_HOME/lib.

3. Upgrade the Hive Metastore database schema. Restart the Hive Metastore database and run:

```
su - hive -c "/usr/hdp/current/hive-metastore/bin/schematool -upgradeSchema
 -dbType <$databaseType>"
```

The value for $databaseType can be derby, mysql, oracle, or postgres.

Note

If you are using Postgres 8 and Postgres 9, you should reset the Hive Metastore database owner to <HIVE_USER>:

```
psql -U <POSTGRES_USER> -c "
ALTER DATABASE <HIVE-METASTORE-DB-NAME> OWNER TO <HIVE_USER>"
```

Note

If you are using Oracle 11, you may see the following error message:

```
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
optimize.mapjoin.mapreduce does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
heapsize does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
server2.enable.impersonation does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
semantic.analyzer.factory.impl does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.auto.
convert.sortmerge.join.noconditionaltask does not exist
Metastore connection URL: jdbc:oracle:thin:@//ip-172-31-42-1.ec2.
internal:1521/XE
Metastore Connection Driver : oracle.jdbc.driver.OracleDriver
Metastore connection User: hiveuser
Starting upgrade metastore schema from version 0.13.0 to 0.14.0
Upgrade script upgrade-0.13.0-to-0.14.0.oracle.sql
Error: ORA-00955: name is already used by an existing object
 (state=42000,code=955)
Warning in pre-upgrade script pre-0-upgrade-0.13.0-to-0.14.0.
oracle.sql: Schema script failed, errorcode 2
Completed upgrade-0.13.0-to-0.14.0.oracle.sql
schemaTool completed
```

You can safely ignore this message. The error is in the pre-upgrade script and can be ignored; the schematool succeeded.

4. Edit the hive-site.xml file and modify the properties based on your environment. Search for TODO in the file for the properties to replace.

   a. Edit the following properties in the hive-site.xml file:

```
<property>
 <name>fs.file.impl.disable.cache</name>
```

```
 <value>false</value>
 <description>Set to false or remove fs.file.impl.disable.cache</
description>
</property>

<property>
 <name>fs.hdfs.impl.disable.cache</name>
 <value>false</value>
 <description>Set to false or remove fs.hdfs.impl.disable.cache
 </description>
</property>
```

b. **Optional:** To enable the Hive buildin authorization mode, make the following changes. If you want to use the advanced authorization provided by Ranger, refer to the Ranger instructions.

Set the following Hive authorization parameters in the hive-site.xml file:

```
<property>
 <name>hive.server2.enable.doAs</name>
 <value>false</value>
</property>

<property>
 <name>hive.security.metastore.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.
   StorageBasedAuthorizationProvider,org.apache.hadoop.hive.ql.security.
authorization.MetaStoreAuthzAPIAuthorizeEmbedOnly</value>
</property>

<property>
 <name>hive.security.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.
SQLStdConfOnlyAuthorizeFactory</value>
</property>
```

Also set hive.users.in.admin.role to the list of comma-separated users who need to be added to admin role. A user who belongs to the admin role needs to run the "set role" command before getting the privileges of the admin role, as this role is not in the current roles by default.

Set the following in the hiveserver2-site.xml file.

```
<property>
 <name>hive.security.authenticator.manager</name>
 <value>org.apache.hadoop.hive.ql.security.
SessionStateUserAuthenticator</value>
</property>

<property>
 <name>hive.security..authorization.enabled</name>
 <value>true</value>
</property>

<property>
 <name>hive.security.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.
SQLStdHiveAuthorizeFactory/value>
</property>
```

c. For a remote Hive metastore database, set the IP address (or fully-qualified domain name) and port of the metastore host using the following hive-site.xml property value.

```
<property>
 <name>hive.metastore.uris</name>
 <value>thrift://$metastore.server.full.hostname:9083</value>
 <description>URI for client to contact metastore server.
   To enable HiveServer2, leave the property value empty.
   </description>
</property>
```

You can further fine-tune your configuration settings based on node hardware specifications, using the HDP utility script.

5. Start Hive Metastore.

   On the Hive Metastore host machine, run the following command:

   ```
   su - hive -c "nohup /usr/hdp/current/hive-metastore/bin/hive
   --service metastore -hiveconf hive.log.file=hivemetastore.log
   >/var/log/hive/hivemetastore.out 2>/var/log/hive/
   hivemetastoreerr.log &"
   ```

6. Start Hive Server2.

   On the Hive Server2 host machine, run the following command:

   ```
   su - hive

   nohup /usr/hdp/current/hive-server2/bin/hiveserver2 -hiveconf
   hive.metastore.uris=" " -hiveconf hive.log.file=hiveserver2.log
   >/var/log/hive/hiveserver2.out 2> /var/log/hive/
   hiveserver2err.log &
   ```

2.16. Configure and Start Apache Oozie

Before you can upgrade Apache Oozie, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The duration of the Oozie upgrade is dependent on the amount of job history stored in ooziedb. This history must be backed up and restored during the upgrade process. Best practice when planning for upgrade is to backup ooziedb from your production oozie server and restore it to a test or development oozie server. This can help you estimate the time that will be required to upgrade Oozie during your production upgrade.

Note

The `su` commands in this section use "hdfs" to represent the HDFS Service user and "oozie" to represent the Oozie Service user. If you are using another name for your HDFS Service user or your Oozie Service user, you will need to substitute your Service user names for "hdfs" or "oozie" in each of the `su` commands.

Upgrading Oozie is a complex process. Although the instructions are straightforward, set aside a dedicated block of time to upgrade oozie clients and servers.

Perform the following preparation steps on each oozie server host:

1. You must restore `oozie-site.xml` from your backup to the conf directory on each oozie server and client.
2. Copy the JDBC jar from `/usr/share/java` to libext-customer:
   a. Create the `/usr/hdp/2.4.3.0-<version>/oozie-server/libext-customer` directory.

      ```
      cd /usr/hdp/2.4.3.0-<version>/oozie-server

      mkdir libext-customer
      ```

   b. Grant read/write/execute access to all users for the libext-customer directory.

      ```
      chmod -R 777 /usr/hdp/2.4.3.0-<version>/oozie-server/libext-customer
      ```

3. Copy these files to the libext-customer directory

   ```
   cp /usr/hdp/2.4.3.0-<version>/hadoop-client/lib/
   hadoop*lzo*.jar /usr/hdp/current/oozie-server/libext-customer

   cp /usr/share/HDP-oozie/ext.zip /usr/hdp/2.4.3.0-<version>/
   oozie-server/libext-customer/
   ```

   Also, copy Oozie db jar in libext-customer.

4. If Falcon was also installed and configured before upgrade in HDP 2.2.x, then after upgrade you might also need to do the following:

   ```
   cp /usr/hdp/current/falcon-server/oozie/ext/falcon-oozie-el-
   extension-"jar /usr/hdp/current/oozie-server/libext-customer
   ```

5. Extract share-lib.

   ```
   /usr/hdp/current/oozie/bin/oozie-setup.sh sharelib create -fs
   hdfs://<namenode>:8020
   ```

   To verify that the sharelibs extracted correctly, run the following command:

   ```
   oozie admin -oozie http://<oozie server host address>:11000/
   oozie -shareliblist
   ```

There should be:

- Available ShareLib
- oozie
- hive
- distcp
- hcatalog
- sqoop
- mapreduce-streaming
- pig

Change the ownership and permissions of the oozie directory:

```
su -l hdfs -c "hdfs dfs -chown oozie:hadoop /user/oozie"

su -l hdfs -c "hdfs dfs -chmod -R 755 /user/oozie"
```

6. If a previous version of Oozie was created using auto schema creation, run the following SQL query:

```
insert into oozie_sys (name, data) values ('db.version', '2.5');
```

7. As the Oozie user (not root), run the upgrade.

```
su - oozie -c "/usr/hdp/current/oozie-server/bin/ooziedb.sh
upgrade -run"
```

8. As root, prepare the Oozie WAR file.

```
chown oozie:oozie /usr/hdp/current/oozie-server/oozie-server/
conf/server.xml

su - oozie -c "/usr/hdp/current/oozie-server/bin/oozie-setup.sh
prepare-war -d /usr/hdp/current/oozie-server/libext-customer"
```

Look for console output to indicate success. For example, if you are using MySQL you should see something similar to:

```
INFO: Adding extension: libext-customer/mysql-connector-java.jar
New Oozie WAR file with added 'JARs' at /var/lib/oozie/oozie-server/webapps/
oozie.war
```

9. Make sure that following property is added in oozie-log4j.properties:

```
log4j.appender.oozie.layout.ConversionPattern=%d{ISO8601} %5p
%c{1}:%L - SERVER[${oozie.instance.id}] %m%n
```

where ${oozie.instance.id} is determined by oozie, automatically.

10. If you have custom Oozie actions, you must define them in oozie-site.xml. Edit the `/etc/oozie/conf/oozie-site.xml` file and add the following property:

```
<property>
      <name>oozie.service.SchemaService.wf.ext.schemas</name>
      <value>[Comma separated list of custom actions]</value>
    </property>
```

For example, if you have added Spark Action, enter the following:

```
<property>
      <name>oozie.service.SchemaService.wf.ext.schemas</name>
      <value>spark-action-0.1.xsd</value>
    </property>
```

11. Configure HTTPS for the Oozie server.

a. Create a self signed certificate or get certificate from a trusted CA for the Oozie Server

b. Import the certificate to the client JDK trust store on all client nodes.

c. In the Ambari Oozie configuration, set the following environment variables in `oozie-env.sh`, adding them if it does not exist:

```
export OOZIE_HTTPS_PORT=11443
export OOZIE_HTTPS_KEYSTORE_FILE=/home/oozie/.keystore
export OOZIE_HTTPS_KEYSTORE_PASS=password
```

d. Change `OOZIE_HTTP_PORT={{oozie_server_port}}` to `OOZIE_HTTP_PORT=11000`.

e. Set the `oozie.base.url` to the HTTPS address.

f. Save the configuration, and restart the Oozie components.

12. Start Oozie as the Oozie user:

```
su - oozie -c "/usr/hdp/current/oozie-server/bin/oozie-start.sh"
```

13. Check processes.

```
ps -ef | grep -i oozie
```

2.17. Configure and Start Apache WebHCat

Before you can upgrade Apache WebHCat, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "hdfs" to represent the HDFS Service user and webhcat to represent the WebHCat Service user. If you are using another name for these Service users, you will need to substitute your Service user name for "hdfs" or "webhcat" in each of the `su` commands.

1. You must replace your configuration after upgrading. Copy `/etc/webhcat/conf` from the template to the conf directory in webhcat hosts.

2. Modify the WebHCat configuration files.

   a. Upload Pig, Hive and Sqoop tarballs to HDFS as the $HDFS_User (in this example, hdfs):

```
su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/pig/"

su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/hive/"

su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/sqoop/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/pig/pig.tar.gz /hdp/
apps/2.4.3.0-$BUILD/pig/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/hive/hive.tar.gz /
hdp/apps/2.4.3.0-$BUILD/hive/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/sqoop/sqoop.tar.gz /
hdp/apps/2.4.3.0-$BUILD/sqoop/"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/pig"

su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/pig/pig.
tar.gz"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/hive"

su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/hive/hive.
tar.gz"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/sqoop/
sqoop.tar.gz"

su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp"
```

   b. Update the following properties in the webhcat-site.xml configuration file, as their values have changed:

```
<property>
 <name>templeton.pig.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/pig/pig.tar.gz</value>
</property>

<property>
 <name>templeton.hive.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/hive/hive.tar.gz</value>
```

```
</property>

<property>
 <name>templeton.streaming.jar</name>
 <value>hdfs:///hdp/apps/${hdp.version}/mapreduce/
   hadoop-streaming.jar</value>
 <description>The hdfs path to the Hadoop streaming jar file.</
description>
</property>

<property>
 <name>templeton.sqoop.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/sqoop/sqoop.tar.gz</value>
 <description>The path to the Sqoop archive.</description>
</property>

<property>
 <name>templeton.sqoop.path</name>
 <value>sqoop.tar.gz/sqoop/bin/sqoop</value>
 <description>The path to the Sqoop executable.</description>
</property>

<property>
 <name>templeton.sqoop.home</name>
 <value>sqoop.tar.gz/sqoop</value>
 <description>The path to the Sqoop home in the exploded archive.
   </description>
</property>
```

Note

You do not need to modify ${hdp.version}.

c. Add the following property if it is not present in webhcat-sitemxml:

```
<property>
 <name>templeton.libjars</name>
 <value>/usr/hdp/current/zookeeper-client/zookeeper.jar,/usr/hdp/current/
hive-client/lib/hive-common.jar</value>
 <description>Jars to add the classpath.</description>
</property>
```

d. Remove the following obsolete properties from webhcat-site.xml:

```
<property>
 <name>templeton.controller.map.mem</name>
 <value>1600</value>
 <description>Total virtual memory available to map tasks.</description>
</property>

<property>
 <name>hive.metastore.warehouse.dir</name>
 <value>/path/to/warehouse/dir</value>
</property>
```

e. Add new proxy users, if needed. In core-site.xml, make sure the following properties are also set to allow WebHCat to impersonate your additional HDP 2.4.3 groups and hosts:

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

<property>
 <name>hadoop.proxyuser.hcat.hosts</name>
 <value>*</value>
</property>
```

Where:

`hadoop.proxyuser.hcat.group`

Is a comma-separated list of the Unix groups whose users may be impersonated by 'hcat'.

`hadoop.proxyuser.hcat.hosts`

A comma-separated list of the hosts which are allowed to submit requests by 'hcat'.

3. Start WebHCat:

```
sudo su -c "usr/hdp/current/hive-webhcat/sbin/webhcat_server.sh
start" hcat
```

4. Smoke test WebHCat.

   a. If you have a non-secure cluster, on the WebHCat host machine, run the following command to check the status of WebHCat server:

   ```
   curl http://$WEBHCAT_HOST_MACHINE:50111/templeton/v1/status
   ```

   You should see the following return status:

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

   b. If you are using a Kerberos secure cluster, run the following command:

   ```
   curl --negotiate -u: http://$WEBHCAT_HOST_MACHINE:50111/
   templeton/v1/status
   ```

   You should see the following return status

   ```
   {"status":"ok","version":"v1"}[machine@acme]$
   ```

2.18. Configure Apache Pig

Before you can upgrade Apache Pig, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

1. Replace your configuration after upgrading. Copy `/etc/pig/conf` from the template to the conf directory in pig hosts.

2. To validate the Pig upgrade, complete the following steps:

   a. On the host machine where Pig is installed, run the following commands:

   ```
   sudo su -c "hadoop fs -copyFromLocal /etc/passwd passwd" $HDFS_USER
   ```

   b. Create a Pig script file named `/tmp/id.pig` that contains the following Pig Latin commands:

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

   c. Run the Pig script:

   ```
   su - $HDFS_USER
   pig -l /tmp/pig.log /tmp/id.pig
   ```

2.19. Configure and Start Apache Sqoop

Before you can upgrade Apache Sqoop, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

1. Replace your configuration after upgrading. Copy /etc/sqoop/conf from the template to the conf directory in sqoop hosts.

2. As the HDFS Service user, upload the Sqoop tarball to HDFS.

```
su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp/apps/2.4.3.0-$BUILD/
sqoop"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/sqoop/sqoop.tar.gz /hdp/
apps/2.4.3.0-$BUILD/sqoop/sqoop.tar.gz"

su - hdfs -c "hdfs dfs -chmod 444 /hdp/apps/2.4.3.0-$BUILD/sqoop/sqoop.tar.
gz"
```

3. If you are using the MySQL database as a source or target, then the MySQL connector jar must be updated to 5.1.29 or later.

Refer to the MySQL web site for information on updating the MySQL connector jar.

4. Because Sqoop is a client tool with no server component, you will need to run your own jobs to validate the upgrade.

2.20. Configure, Start, and Validate Apache Flume

Before you can upgrade Apache Flume, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

1. To confirm that Flume is working correctly, create an example configuration file. The following snippet is a sample configuration that can be set using the properties file. For more detailed information, see the "Flume User Guide."

```
agent.sources = pstream
agent.channels = memoryChannel
agent.channels.memoryChannel.type = memory

agent.sources.pstream.channels = memoryChannel
agent.sources.pstream.type = exec
agent.sources.pstream.command = tail -f /etc/passwd

agent.sinks = hdfsSink
agent.sinks.hdfsSink.type = hdfs
agent.sinks.hdfsSink.channel = memoryChannel
agent.sinks.hdfsSink.hdfs.path = hdfs://tmp/flumetest
agent.sinks.hdfsSink.hdfs.fileType = SequenceFile
agent.sinks.hdfsSink.hdfs.writeFormat = Text
```

   The source here is defined as an exec source. The agent runs a given command on startup, which streams data to stdout, where the source gets it. The channel is defined as an in-memory channel and the sink is an HDFS sink.

2. Given this configuration, you can start Flume as follows:

```
$ bin/flume-ng agent --conf ./conf --conf-file example.conf --name a1 -
Dflume.root.logger=INFO,console
```

Note

The directory specified for `--conf agrument` would include a shell script flume-env.sh and potentially a log4j properties file. In this example, we pass a Java option to force Flume to log to the console and we go without a custom environment script.

3. After validating data in `hdfs://tmp/flumetest`, stop Flume and restore any backup files. Copy `/etc/flume/conf` to the conf directory in Flume hosts.

2.21. Configure, Start, and Validate Apache Mahout

Before you can upgrade Apache Mahout, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Replace your configuration after upgrading. Copy `/etc/mahout/conf` from the template to the conf directory in mahout hosts.

To validate mahout:

1. Create a test user named "testuser" in the Linux cluster and in HDFS, and log in as that user.
2. Export the required environment variables for Mahout:

```
export JAVA_HOME="your_jdk_home_install_location_here
export HADOOP_HOME=/usr/hdp/current/hadoop-client
export MAHOUT_HOME=/usr/hdp.current/mahout-client
export PATH="$PATH":$HADOOP_HOME/bin:$MAHOUT_HOME/bin
export CLASSPATH="$CLASSPATH":$MAHOUT_HOME
```

3. Upload a few megabytes of natural-language plain text to the Linux server as `/tmp/sample-test.txt`.
4. Transfer the sample-test.txt file to a subdirectory of the testusers's HDFS home directory.

```
hdfs dfs -mkdir /user/testuser/testdata
hdfs dfs -put /tmp/sample-test.txt /user/testuser/testdata
```

5. Enter the mahout command to convert the plain text file sample-test.txt into a sequence file stored in the output directory mahouttest:

```
mahout seqdirectory --input /user/testuser/testdata --output /user/testuser/
mahouttest -ow --charset utf-8
```

2.22. Configure and Start Hue

Before you can upgrade Hue, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

For HDP 2.4.3, use the Hue version shipped with HDP 2.4.3. If you have a previous version of Hue, use the following steps to upgrade Hue.

Complete one of the following:

- **SQLite**

  1. Copy the hue.ini settings from your old hue.ini configuration file to new hue.ini configuration file.

  2. Restore your database after upgrade.

     To restore the database from a backup, make sure the destination database is empty before copying (if necessary, rename or remove the current destination database), then copy your backup to the destination database. For example:

     ```
     su - hue
     cd /var/lib/hue
     mv desktop.db desktop.db.old
     sqlite3 desktop.db < ~/hue_backup/desktop.bak
     exit
     ```

  3. Synchronize database.

     ```
     cd /usr/lib/hue
     source ./build/env/bin/activate
     hue syncdb
     hue migrate
     deactivate
     ```

  4. Start Hue. As a root user, run the following command on the Hue Server:

     ```
     /etc/init.d/hue start
     ```

- **MySQL**

  1. Copy the hue.ini settings from your old hue.ini configuration file to new hue.ini configuration file.

  2. Start Hue. As a root user, run the following command on the Hue Server:

     ```
     /etc/init.d/hue start
     ```

2.23. Configure and Start Apache Knox

Before you can upgrade Apache Knox, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already

upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

When working with the Knox Gateway in your Hadoop cluster, it is important you have the latest version of Knox installed so you can take advantage of new features and enhancements, in addition to ensuring your instance of Knox is in sync with other Hadoop components (e.g., Ranger, Spark, Hive, Hue, etc.) for stability and performance. For example, if you need to upgrade your Hadoop cluster from 2.2 to 2.4.3, you should also make sure that your individual Hadoop components are also upgraded to the latest version.

Note

In this document, whenever you see a {} with a value inside, this denotes a value you must define.

2.23.1. Upgrade the Knox Gateway

If you are not currently using Ambari to manage your Hadoop cluster, you will need to upgrade Knox manually to the latest version. Because "rolling upgrades" are now supported in HDP 2.4.3, it is not important which version of Knox you are currently running, only that you have an instance of the Knox Gateway running.

Note

If you have not already installed Knox, refer to the "Install the Knox RPMs on the Knox Server" section of the *Non-Ambari Cluster Installation Guide* for instructions on how to install and configure the Knox Gateway.

Before upgrading the Knox Gateway, there are a several steps you must follow to ensure your configuration files, settings, and topology files can be copied to the new Knox Gateway instance when the upgrade is complete, which are described below.

Note

The `su` commands in this section use "knox" to represent the Knox Service user. If you are using another name for your Knox Service user, you will need to substitute your Knox Service user name for "knox" in each of the `su` commands.

1. Back up your existing `conf` directory if you have not already done so.

2. Stop each Knox server if you have not already done so.

```
su -l knox /usr/hdp/{the current Knox version}/knox/bin/gateway.sh stop
```

3. Select the HDP server version you are upgrading to after you have stopped each Knox server if you have not already done so.

```
hdp-select set knox-server {the HDP server version}
```

4. Start the ldap service.

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

5. For HDP 2.4.3, the default paths for Knox change. Upgrade Knox in order to update these paths.

   a. Restore the backed up security directory. This will place the master secret and keystores back in place for the new deployment.

   b. Start the Gateway:

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

   c. Unzip your previously saved configuration directory (the `conf` directory you backed up in step 1) into the new */var/log/knox/gateway.conf* directory to import these files.

   d. Restart the Knox server to complete the upgrade.

   ```
   su -l knox -c "/usr/hdp/{the new HDP server version}/knox-server/bin/
   gateway.sh start"
   ```

2.23.2. Verify the Knox Upgrade

To verify the upgrade was successful, follow the steps listed below.

1. Navigate to the */var/log/knox/gateway* directory and check the `gateway.log` file for errors and an acknowledgement that the server restart was successful.

2. Verify you have cluster access using the `LISTSTATUS` WebHDFS API call.

   ```
   curl -ivk -u {user}:{password} https://{knox host}:8443 /gateway/webhdfs/v1/
   ?op=LISTSTATUS
   ```

3. Verify the Knox version using the Knox Admin service and Version API.

   ```
   curl -i -v -k -u admin:admin-password -X GET "https://localhost:8443/
   gateway/admin/api/v1/version"
   curl -i -v -k -u admin:admin-password -X GET "https://localhost:8443/
   gateway/sandbox/webhdfs/v1?op=LISTSTATUS"
   ```

Note

The Admin API requires you to be a member of an Admin group, as specified in the `admin.xml` authorization provider.

When you have verified the Knox upgrade was successful, you can begin using Knox.

2.24. Configure and Validate Apache Falcon

Before you can upgrade Apache Falcon, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

In HDP 2.4.3, if authorization is enabled (for example, in the properties file with `*.falcon.security.authorization.enabled=true`") then Access Control List (ACL) is mandated for all entities.

Upgrade Falcon after you have upgraded HDFS, Hive, Oozie, and Pig. Stop Oozie jobs while upgrading Falcon.

1. Replace your configuration after upgrading. Copy /etc/falcon/conf from the template to the conf directory in falcon hosts.
2. Check your Falcon entities. There should be no changes, but in some cases you may need to update your entities post-upgrade.
3. In HDP 2.4.3 for Falcon, TLS is enabled by default. When TLS is enabled, Falcon starts on https://<falcon_host>.15443/. You can disable TLS by adding the following line to the startup.properties file.

   `"*.falcon.enableTLS=false`
4. If Transport Layer Security (TLS) is disabled, check the client.properties file to make sure the property "falcon.uri" is set as follows:

   `falcon.uri=http://<falcon_host>:15000/`

2.25. Configure and Start Apache Storm

Before you can upgrade Apache Storm, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "zookeeper" to represent the ZooKeeper Service user. If you are using another name for your ZooKeeper Service user, you will need to substitute your ZooKeeper Service user name for "zookeeper" in each of the `su` commands.

Storm is fairly independent of changes to the HDP cluster, but you must upgrade Storm for rolling upgrade support in HDP 2.4.3 and be on the latest version of Storm.

1. After upgrading Storm, replace your configuration. Copy `/etc/storm/conf` from the template to the `conf` directory .
2. Replace your ZooKeeper configuration after upgrading. Replace the ZooKeeper template configuration in `/etc/zookeeper/conf`.
3. Ensure ZooKeeper is running. On the storm node, run the following command:

```
su - zookeeper -c "source /etc/zookeeper/conf/zookeeper-env.sh; export
 ZOOCFGDIR=/etc/zookeeper/conf; /usr/hdp/current/zookeeper-server/bin/
zkServer.sh start >> $ZOO_LOG_DIR/zoo.out\"
```

where

- $ZOO_LOG_DIR is the directory where ZooKeeper server logs are stored. For example, /var/log/zookeeper.

4. Start nimbus, then supervisor/ui/drpc/logviewer:

   `/usr/hdp/current/storm-nimbus/bin/storm nimbus`.

5. Start Storm, using a process controller, such as supervisor:

   ```
   su - storm /usr/hdp/current/storm-supervisor/bin/storm
   supervisor
   ```

   You can use the same command syntax to start Storm using nimbus/ui and logviewer.

   ```
   su - storm /usr/hdp/current/storm-supervisor/bin/storm nimbus

   su - storm /usr/hdp/current/storm-supervisor/bin/storm ui

   su - storm /usr/hdp/current/storm-supervisor/bin/storm logviewer

   su - storm /usr/hdp/current/storm-supervisor/bin/storm drpc
   ```

2.26. Configure and Start Apache Ranger

Before you can upgrade the Apache Ranger service, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already performed the following tasks, however, if you have not already performed these steps, refer to the "Upgrade HDP 2.2 Components" section in this guide for instructions on how to upgrade your HDP components to 2.4.3.

2.26.1. Preparing Your Cluster to Upgrade Ranger

If you are not currently using Ambari to manage your Hadoop cluster, you will need to upgrade Ranger manually to the latest version. This section describes the steps you need to follow to prepare your cluster for the Ranger upgrade.

1. Back up the following Ranger configuration directories:

   - Ranger Policy Administration Service

     ```
     /etc/ranger/admin/conf
     ```

   - Ranger UserSync

     ```
     /etc/ranger/usersync/conf
     ```

   - Ranger Plugins:

     - Hadoop

       ```
       /etc/hadoop/conf
       ```

- Hive

  ```
  /etc/hive/conf
  ```

- HBase

  ```
  /etc/hbase/conf
  ```

- Knox

  ```
  /etc/knox/conf
  ```

- Storm

  ```
  /etc/storm/conf
  ```

2. Backup the Ranger Policy and Audit databases. Make sure to take note of the following details in the `install.properties` file:

   - db_host
   - db_name
   - db_user
   - db_password
   - policy manager configuration
   - LDAP directory configuration
   - LDAP settings
   - LDAP AD domain
   - LDAP URL

   ```
   mysqldump -u root -p root xasecure > dest_dir/filename.sql
   mysqldump -u root -p root xasecure_audit > dest_dir/filename.sql
   ```

2.26.2. Stop the Ranger Services

Now that you have prepared your cluster for the Ranger upgrade, you will need to stop the Ranger Admin and Ranger UserSync services. To stop the Ranger services, perform the steps described below.

1. Stop the Ranger Policy Admin service. When the service is stopped, you will receive an acknowledgement from the server that the service has been stopped.

   ```
   service ranger-admin stop
   ```

2. Stop the Ranger UserSync service. When the service is stopped, you will receive an acknowledgement from the server that the service has been stopped.

   ```
   service ranger-usersync stop
   ```

3. Stop the applicable services using the Ranger plugin (HDFS, HBase, Hive, Knox, Storm).

See Stopping HDP Services for more information.

2.26.3. Preparing the Cluster for Upgrade

Before you begin the upgrade process, you will need to perform a series of steps to prepare the cluster for upgrade. These steps are described in the "*Getting Ready To Upgrade*" section of this guide, which you will need to follow before continuing to upgrade Ranger. Some of these steps include:

- Backing up HDP directories
- Stopping all long-running applications and services.
- Backing up the Hive and Oozie metastore databases.
- Backing up Hue
- Backing up specific directories and configurations

2.26.4. Registering the HDP 2.4.3 Repo

After you have prepared your cluster for the upgrade, you need to register the HDP 2.4.3 repo. This requires you to perform the following steps:

1. (Optional) The Ranger components should already have installed in the at the beginning of the HDP upgrade process, but you can use the following commands to confirm that the Ranger packages have been installed:

   ```
   hdp-select status ranger-admin
   hdp-select status ranger-usersync
   ```

   If the packages have not been installed, you can use the install commands specific to your OS. For example, for RHEL/CentOS you would use the following commands to install the packages.

   ```
   yum install ranger_2_3_*-admin
   yum install ranger_2_3_*-usersync
   ```

2. Select the Ranger Admin and Ranger UserSync versions you want to use.

   ```
   hdp-select set ranger-admin <HDP_server_version>
   hdp-select set ranger-usersync <HDP_server_version>
   ```

3. Change ownership of `/etc/ranger/admin/conf/` and `/etc/ranger/usersync/conf/` to `ranger:ranger`:

   ```
   chown -R ranger:ranger /etc/ranger/admin/conf/
   ```

4. Update the `install.properties` file to migrate the database credentials properties and POLICYMGR_EXTERNAL-URL property from HDP 2.2. to HDP 2.4.3.

5. Install the Ranger Admin component. Be sure to set the JAVA_HOME environment variable if it is not already set.

```
cd /usr/hdp/current/ranger-admin/

cp ews/webapp/WEB-INF/classes/conf.dist/ranger-admin-site.xml ews/webapp/
WEB-INF/classes/conf/

cp ews/webapp/WEB-INF/classes/conf.dist/ranger-admin-default-site.xml ews/
webapp/WEB-INF/classes/conf/

cp ews/webapp/WEB-INF/classes/conf.dist/security-applicationContext.xml ews/
webapp/WEB-INF/classes/conf/

./setup.sh
```

6. This should successfully install the Ranger Admin component.
7. Start the Ranger Admin component.

```
service ranger-admin start
```

8. You must now configure and setup the Ranger UserSync component by migrating the properties from the HDP 2.2 `install.properties` file (POLICY_MGR_URL, SYNC_SOURCE and LDAP/AD properties.
9. Install the Ranger UserSync component. Be sure to set the JAVA_HOME component if it is not already set.

```
cd /usr/hdp/current/ranger-usersync/

./setup.sh
```

10. Start the Ranger UserSync component.

```
service ranger-usersync start
```

2.26.5. Install the Ranger Components

Next, you will need to re-install each Ranger component again to ensure you have the latest version. Because you have already upgraded your HDP stack, you only need to follow the instructions in the *Non-Ambari Cluster Installation Guide* to install each Ranger component. You must install the following Ranger components:

- Ranger Policy Admin
- Ranger UserSync
- Ranger Plugins:
  - HDFS
  - HBase
  - Hive
  - Knox
  - Storm

Note

When installing each Ranger component, you will also need to make sure you upgrade each individual component to version 2.4.3 before restarting each service.

With this release, Ranger has also added support for the following components:

- Solr
- Kafka
- YARN

2.26.6. Restart the Ranger Services

Once you have re-installed each Ranger component, you will then need to restart these components to ensure the new configurations are loaded in your cluster. This includes restarting the Policy Admin and UserSync components, NameNode, and each Ranger plugin.

Note

Before restarting the NameNode, make sure to remove the `set-hdfs-plugin-env.sh` from */etc/hadoop/conf*. You will need to re-enable the NameNode after finishing the upgrade.

The *Non-Ambari Cluster Installation Guide* describes how you can start the following Ranger services:

- Ranger Policy Admin service

```
service ranger-admin start
```

- Ranger UserSync service

```
service ranger-usersync start
```

2.26.7. Enable Ranger Plugins

The final step in the Ranger upgrade process requires you to re-enable the Ranger plugins. Although you are only required to enable HDFS in your cluster, you should re-enable all of the Ranger plugins because class names have changed for the 2.4.3 release, and to ensure smooth operation of Ranger services in your cluster.

Note

When you enable each Ranger plugin, be sure to remove all 2.2 class name values.

Note

Re-enabling a Ranger plugin does not affect policies you have already created. As long as you use the same database as the Policy store, all of your data will remain intact.

To re-enable the Ranger plugins, use the links listed below to access instructions in the *Non-Ambari Cluster Installation* guide that describe editing the `install.properties` file and enabling the Ranger plugins:

Important

Before enabling the HDFS plugin, remove `set-hdfs-plugin-env.sh` from `/etc/hadoop/conf`. You will need to re-enable this plugin after the upgrade is complete.

- HDFS Plugin
- YARN Plugin
- Kafka Plugin
- HBase Plugin
- Hive Plugin
- Knox Plugin
- Storm Plugin

2.27. Configuring and Upgrading Apache Spark

Before you can upgrade Apache Spark, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

Instructions in this section are specific to HDP-2.4.3 and later. For earlier versions for HDP, refer to the version-specific documentation.

1. Add the node where you want Spark 1.6 History Server to run. Install the version corresponding to the HDP version you currently have installed.

   a. `su - root`

   b. `yum install spark_2_4_3_0_$BUILD-master -y`

   c. To use Python: `yum install spark_2_4_3_0_$BUILD-python`

   d. `conf-select create-conf-dir --package spark --stack-version spark_2_4_3_0_$BUILD --conf-version 0`

   e. `cp /etc/spark/spark_2_4_3_0_$BUILD/0/* /etc/spark/spark_2_4_3_0_$BUILD/0/`

   f. `conf-select set-conf-dir --package spark --stack-version spark_2_4_3_0_$BUILD --conf-version 0`

   g. `hdp-select set spark-client spark_2_4_3_0_$BUILD`

h. `hdp-select set spark-historyserver spark_2_4_3_0_$BUILD`

2. In HDP 2.4.3, the Spark History Server runs on top of HDFS, not YARN ATS, as in previous versions. Modify Spark configuration files as follows:

   a. As the hdfs service user, create an HDFS directory called spark-history with user:spark, user group:hadoop, and permissions = 777:

   ```
   hdfs dfs -mkdir /spark-history
   hdfs dfs -chown -R spark:hadoop /spark-history
   hdfs dfs -chmod -R 777 /spark-history
   ```

   b. Edit the `spark-defaults.conf` file.

   - Add the following properties and values:

   ```
   spark.eventLog.dir to hdfs:///spark-history
   spark.eventLog.enabled to true
   spark.history.fs.logDirectory to hdfs:///spark-history
   ```

   - Delete the `spark.yarn.services` property.

   c. Edit the `spark-thrift-sparkconf.conf` file

   - Add the following properties and values:

   ```
   spark.eventLog.dir to hdfs:///spark-history
   spark.eventLog.enabled to true
   spark.history.fs.logDirectory to hdfs:///spark-history
   ```

3. Restart Spark on YARN in either yarn-cluster mode or yarn-client mode:

   - yarn-cluster mode: `./usr/hdp/current/spark-client/bin/spark-submit --class path.to.your.Class --master yarn-cluster [options] [app options]`
   - yarn-client mode: `./usr/hdp/current/spark-client/bin/spark-shell --master yarn-client`

4. Validate the Spark installation. As user spark, run SparkPI example:

   a. `sudo su spark`

   b. `cd /usr/hdp/current/spark-client`

   c. `./bin/run-example SparkPi 10`

For additional configuration information, see the Spark Guide.

2.28. Upgrade Apache Slider

Before you can upgrade Apache Slider, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have

already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

To upgrade Slider, simply upgrade the Slider client.

1. Upgrade Slider client:

```
hdp-select set slider-client 2.4.3.0-<version>
slider version
```

2.29. Upgrade Apache Kafka

Before you can upgrade Apache Kafka, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.2 Components for instructions on how to upgrade your HDP components to 2.4.3.

For more information about Kafka on HDP, see the Kafka Guide

To upgrade Kafka 2.4.3, you must stop all of the Kafka brokers before you start the upgrade.

1. Shut down the current Kafka daemon, switch to the new version, and start the daemon:

```
su - kafka -c "/usr/hdp/current/kafka-broker/bin/kafka stop"
   hdp-select set kafka-broker 2.4.3.0-165
   su - kafka -c "usr/hdp/current/kafka-broker/bin/kafka start"
```

2. To verify that the Kafka daemon joined the cluster, create a topic and submit it to Kafka. Send a test message for that topic, and then validate that it was received by a consumer.

3. If the upgrade process fails, follow the steps in "Downgrading Kafka" to return to your previous version of Kafka.

2.29.1. Downgrading Kafka

Downgrade each Kafka node one at a time. You can stop each Kafka broker and upgrade the component without downtime if you have enough replication for your topic.

1. Shut down the current Kafka daemon, switch to your previous version, and start the daemon. For example:

```
su - kafka -c "/usr/hdp/current/kafka-broker/bin/kafka stop"
hdp-select set kafka-broker 2.2.2.0-2041
su - kafka -c "/usr/hdp/current/kafka-broker/bin/kafka start"
```

2. To verify that the Kafka daemon joined the cluster, create a topic and submit it to Kafka. Send a test message for that topic, and then validate that it was received by a consumer.

2.30. Finalize the Upgrade

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

You can start HDFS without finalizing the upgrade. When you are ready to discard your backup, you can finalize the upgrade.

1. Verify your file system health before finalizing the upgrade. (After you finalize an upgrade, your backup will be discarded!)
2. As the $HDFS_USER, enter:

```
su - hdfs -c "hdfs dfsadmin -finalizeUpgrade"
```

2.31. Install New HDP Services

Install new HDP 2.3.0 Services if desired. See the Non-Ambari Cluster Installation Guide for details.

- Atlas – a low-level service, similar to YARN, that provides metadata services to the HDP platform.
- SmartSense – a next generation subscription model that features upgrade and configuration recommendations.

3. Upgrade from HDP 2.1 to HDP 2.4.3 Manually

Important

If you installed and manage HDP-2.1 with Ambari, **you must use the Ambari Upgrade Guide** to perform the the HDP-2.1 to HDP-2.4.3 upgrade.

Note

These instructions cover the upgrade between two minor releases, such as HDP-2.1 to HDP-2.4. Component information has been updated for HDP-2.4.3 where appropriate. If you need to upgrade between two maintenance releases such as HDP-2.1.x to 2.1.7, follow the upgrade instructions in the HDP Release Notes.

Starting with HDP-2.2, HDP supports side-by-side installation of HDP 2.2 and subsequent releases, which lets you perform rolling upgrades on your cluster and improve execution times for in-place upgrade. To support side-by-side installation, the HDP package version naming convention for both RPMs and Debs has changed to include the HDP product version. For example, hadoop-hdfs in HDP-2.2.4 is now hadoop_2.2.4.2.hdfs. HDP-2.2 marked the first release where HDP rpms, debs, and directories contained versions in the names to permit side-by-side installations of later HDP releases. To select from the releases you have installed side-by-side, Hortonworks provides hdp-select, a command that lets you select the active version of HDP from the versions you have installed.

However, because HDP-2.1 did not support side-by-side installation, you will upgrade to HDP-2.4.3 in a way very similar to previous minor-version upgrades. Subsequent upgrades after 2.4.3 will be easier, and if you choose to add Ambari to your cluster, you will be able to use the Rolling Upgrade feature.

Important

You cannot perform a side-by-side rolling upgrade from HDP 2.1 to HDP-2.4.3; only upgrade in place is supported. Side-by-side and rolling upgrade support starts with releases HDP 2.2 and above.

This document provides instructions on how to manually upgrade to HDP-2.4.3 from the HDP 2.1 release. It assumes the existing HDP 2.1 was also installed manually.

The HDP packages for a complete installation of HDP-2.4.3 will take about 2.5 GB of disk space.

Warning

Until the upgrade is finalized, no HDFS data is deleted from the cluster. Be sure to review your capacity and ensure that you have extra space available during the upgrade window.

Important

You are strongly encouraged to read completely through this entire document before starting the Manual Upgrade process, in order to understand the interdependencies and the order of the steps.

The following steps are required to upgrade from HDP-2.1 to the latest release of HDP-2.4.3 from HDP-2.1:

1. Get ready to upgrade
2. Upgrade HDP 2.1 Components
3. Symlink Directories with hdp-select
4. Configure and Start Apache ZooKeeper
5. Configure and Start Hadoop
6. Start HDFS
7. Configure and Start Apache HBase
8. Configure and Start Apache Phoenix
9. Configure and Start Apache Accumulo
10. Configure and Start Apache Tez
11. Configure and Start Apache Hive and Apache HCatalog
12. Configure and Start Apache Oozie
13. Configure and Start Apache WebHCat (Templeton)
14. Configure and Start Apache Pig
15. Configure and Start Apache Sqoop
16. Configure and Start Apache Flume
17. Configure and Validate Apache Mahout
18. Configure and Start Hue
19. Configure and Start Apache Knox
20. Configure and Start Apache Falcon
21. Finalize Upgrade
22. Install new HDP services if desired

3.1. Getting Ready to Upgrade

HDP Stack upgrade involves upgrading from HDP 2.1 to HDP-2.4.3 versions and adding the new HDP-2.4.3 services. These instructions change your configurations.

Note

You must use **kinit** before running the commands as any particular user.

Hardware recommendations

Although there is no single hardware requirement for installing HDP, there are some basic guidelines. The HDP packages for a complete installation of HDP-2.4.3 will take up about 2.5 GB of disk space.

The first step is to ensure you keep a backup copy of your HDP 2.1 configurations.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

1. Back up the HDP directories for any hadoop components you have installed.

   The following is a list of all HDP directories:

   - `/etc/hadoop/conf`
   - `/etc/hbase/conf`
   - `/etc/hive-hcatalog/conf`
   - `/etc/hive-webhcat/conf`
   - `/etc/accumulo/conf`
   - `/etc/hive/conf`
   - `/etc/pig/conf`
   - `/etc/sqoop/conf`
   - `/etc/flume/conf`
   - `/etc/mahout/conf`
   - `/etc/oozie/conf`
   - `/etc/hue/conf`
   - `/etc/zookeeper/conf`
   - `/etc/tez/conf`
   - `/etc/storm/conf`
   - Optional - Back up your userlogs directories, `${mapred.local.dir}/userlogs`.

2. Navigate to the $HIVE_HOME/lib directory. Backup the JDBC jar file for the type of Hive metastore you are using (Postgre, MySQL etc).

3. Run the `fsck` command as the HDFS Service user and fix any errors. (The resulting file contains a complete block map of the file system.)

   ```
   su - hdfs -c "hdfs fsck / -files -blocks -locations > dfs-old-fsck-1.log"
   ```

4. Use the following instructions to compare status before and after the upgrade.

   The following commands must be executed by the user running the HDFS service (by default, the user is hdfs).

   a. Capture the complete namespace of the file system. (The following command does a recursive listing of the root file system.)

   

   Important

   Make sure the NameNode is started.

   ```
   su - hdfs -c "hdfs dfs -ls -R / > dfs-old-lsr-1.log"
   ```

   

   Note

   In secure mode you must have Kerberos credentials for the hdfs user.

   b. Run the report command to create a list of DataNodes in the cluster.

   ```
   su - hdfs -c “hdfs dfsadmin –report > dfs-old-report-1.log"
   ```

   c. **Optional:** You can copy all or unrecoverable only data storelibext-customer directory in HDFS to a local file system or to a backup instance of HDFS.

   d. **Optional:** You can also repeat the steps 3 (a) through 3 (c) and compare the results with the previous run to ensure the state of the file system remained unchanged.

5. Save the namespace by executing the following commands:

   ```
   su - hdfs

   hdfs dfsadmin -safemode enter

   hdfs dfsadmin -saveNamespace
   ```

6. Backup your NameNode metadata.

   a. Copy the following checkpoint files into a backup directory:

   The NameNode metadata is stored in a directory specified in the hdfs-site.xml configuration file under the configuration value "dfs.namenode.dir".

   For example, if the configuration value is:

   ```
   <property>
   ```

```
  <name>dfs.namenode.name.dir</name>
  <value>/hadoop/hdfs/namenode</value>
</property>
```

Then, the NameNode metadata files are all housed inside the directory `/hadooop.hdfs/namenode`.

b. Store the layoutVersion of the namenode.

`${dfs.namenode.name.dir}/current/VERSION`

7. Finalize any prior HDFS upgrade, if you have not done so already.

```
su - hdfs -c "hdfs dfsadmin -finalizeUpgrade"
```

8. If you have the Hive component installed, back up the Hive Metastore database.

The following instructions are provided for your convenience. For the latest backup instructions, see your database documentation.

Table 3.1. Hive Metastore Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sqlsbr For example: mysqldump hive > /tmp/mydir/backup_hive.sql	mysql $dbname < $inputfilename.sqlsbr For example: mysql hive < /tmp/mydir/backup_hive.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql sbr For example: sudo -u postgres pg_dump hive > /tmp/mydir/backup_hive.sql	sudo -u $username psql $databasename < $inputfilename.sqlsbr For example: sudo -u postgres psql hive < /tmp/mydir/backup_hive.sql
Oracle	Export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database file=input_file.dmp

9. If you have the Oozie component installed, back up the Oozie metastore database.

These instructions are provided for your convenience. Please check your database documentation for the latest backup instructions.

Table 3.2. Oozie Metastore Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sql For example: mysqldump oozie > /tmp/mydir/backup_oozie.sql	mysql $dbname < $inputfilename.sql For example: mysql oozie < /tmp/mydir/backup_oozie.sql

Database Type	Backup	Restore

Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql
	
	For example:
	
	`sudo -u`
	
	`postgres pg_dump oozie > /tmp/mydir/backup_oozie.sql`
		sudo -u $username psql $databasename < $inputfilename.sql
		
		For example:
		
		`sudo -u`
		
		`postgres psql oozie < /tmp/mydir/backup_oozie.sql`

Oracle	Export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database file=input_file.dmp

10. **Optional**: Back up the Hue database.

    The following instructions are provided for your convenience. For the latest backup instructions, please see your database documentation. For database types that are not listed below, follow your vendor-specific instructions.

Table 3.3. Hue Database Backup and Restore

Database Type	Backup	Restore
MySQL	mysqldump $dbname > $outputfilename.sqlsbr For example: mysqldump hue > /tmp/mydir/backup_hue.sql	mysql $dbname < $inputfilename.sqlsbr For example: mysql hue < /tmp/mydir/backup_hue.sql
Postgres	sudo -u $username pg_dump $databasename > $outputfilename.sql sbr For example: sudo -u postgres pg_dump hue > /tmp/mydir/backup_hue.sql	sudo -u $username psql $databasename < $inputfilename.sqlsbr For example: sudo -u postgres psql hue < /tmp/mydir/backup_hue.sql
Oracle	Connect to the Oracle database using sqlplus. Export the database. For example: exp username/password@database full=yes file=output_file.dmp mysql $dbname < $inputfilename.sqlsbr	Import the database: For example: imp username/password@database file=input_file.dmp
SQLite	/etc/init.d/hue stop su $HUE_USER mkdir ~/hue_backup sqlite3 desktop.db .dump > ~/hue_backup/desktop.bak /etc/init.d/hue start	/etc/init.d/hue stop cd /var/lib/hue mv desktop.db desktop.db.old sqlite3 desktop.db < ~/hue_backup/desktop.bak /etc/init.d/hue start

11. Stop all services (including MapReduce) and client applications deployed on HDFS:

Component	Command
Knox	`cd /usr/lib/knox/ su knox -c "bin/gateway.sh stop”`

Oozie	`su $OOZIE_USER`
	
	`/usr/lib/oozie/bin/oozied.sh stop`

WebHCat	`su - hcat -c "/usr/lib/hive-hcatalog/sbin/webhcat_server.sh stop"`
Hive	Run this command on the Hive Metastore and Hive Server2 host machine: `ps aux | awk '{print $1,$2}' | grep hive | awk '{print $2}' | xargs kill >/dev/null 2>&1`
HBase RegionServers	`su - hbase -c "/usr/lib/hbase/bin/hbase-daemon.sh --config /etc/hbase/conf stop regionserver"`
HBase Master host machine	`su - hbase -c "/usr/lib/hbase/bin/hbase-daemon.sh --config /etc/hbase/conf stop master"`

YARN	Run this command on all NodeManagers:
	
	`su - yarn -c "export HADOOP_LIBEXEC_DIR=/usr/lib/hadoop/libexec && /usr/lib/hadoop-yarn/sbin/yarn-daemon.sh --config /etc/hadoop/conf stop nodemanager"`
	
	Run this command on the History Server host machine:
	
	`su - mapred -c "export HADOOP_LIBEXEC_DIR=/usr/lib/hadoop/libexec && /usr/lib/hadoop-mapreduce/sbin/mr-jobhistory-daemon.sh --config /etc/hadoop/conf stop historyserver"`
	
	Run this command on the ResourceManager host machine(s):
	
	`su - yarn -c "export HADOOP_LIBEXEC_DIR=/usr/lib/hadoop/libexec && /usr/lib/hadoop-yarn/sbin/yarn-daemon.sh --config /etc/hadoop/conf stop resourcemanager"`
	
	Run this command on the YARN Timeline Server node:
	
	`su -l yarn -c "export HADOOP_LIBEXEC_DIR=/usr/lib/hadoop/libexec && /usr/lib/hadoop-yarn/sbin/yarn-daemon.sh --config /etc/hadoop/conf stop timelineserver"`

HDFS	On all DataNodes:
	
	`su - hdfs -c "/usr/lib/hadoop/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop datanode"`
	
	If you are not running a highly available HDFS cluster, stop the Secondary NameNode by executing this command on the Secondary NameNode host machine:
	
	`su - hdfs -c "/usr/lib/hadoop/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop secondarynamenode”`

Component	Command

	On the NameNode host machine(s):
	
	`su - hdfs -c "/usr/lib/hadoop/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop namenode"`
	
	If you are running NameNode HA, stop the ZooKeeper Failover Controllers (ZKFC) by executing this command on the NameNode host machine:
	
	`su - hdfs -c "/usr/lib/hadoop/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop zkfc"`
	
	If you are running NameNode HA, stop the JournalNodes by executing these commands on the JournalNode host machines:
	
	`su - hdfs -c "/usr/lib/hadoop/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop journalnode"`

ZooKeeper Host machines	`su - zookeeper -c "export ZOOCFGDIR=/etc/zookeeper/conf ; export ZOOCFG=zoo.cfg ;source /etc/zookeeper/conf/zookeeper-env.sh ; /usr/lib/zookeeper/bin/zkServer.sh stop"`

Ranger (XA Secure)	`service xapolicymgr stop`
	
	`service uxugsync stop`

12. Verify that edit logs in `${dfs.namenode.name.dir}/current/edits*` are empty.

    a. Run: `hdfs oev -i ${dfs.namenode.name.dir}/current/edits_inprogress_* -o edits.out`

    b. Verify edits.out file. It should only have OP_START_LOG_SEGMENT transaction. For example:

```
<?xml version="1.0" encoding="UTF-8"?><EDITS>
<EDITS_VERSION>-56</EDITS_VERSION>
<RECORD>
<OPCODE>OP_START_LOG_SEGMENT</OPCODE>
<DATA>
<TXID>5749</TXID>
</DATA>
</RECORD>
```

    c. If edits.out has transactions other than OP_START_LOG_SEGMENT, run the following steps and then verify edit logs are empty.

    - Start the existing version NameNode.
    - Ensure there is a new FS image file.
    - Shut the NameNode down:

      `hdfs dfsadmin – saveNamespace`

13. Rename or delete any paths that are reserved in the new version of HDFS.

When upgrading to a new version of HDFS, it is necessary to rename or delete any paths that are reserved in the new version of HDFS. If the NameNode encounters a reserved path during upgrade, it will print an error such as the following:

```
/.reserved is a reserved path and .snapshot is a reserved path
component in this version of HDFS.

Please rollback and delete or rename this path, or upgrade with the
-renameReserved key-value pairs option to automatically rename these
paths during upgrade.
```

Specifying `-upgrade -renameReserved` optional key-value pairs causes the NameNode to automatically rename any reserved paths found during startup.

For example, to rename all paths named `.snapshot` to `.my-snapshot` and change paths named `.reserved` to `.my-reserved`, specify `-upgrade -renameReserved .snapshot=.my-snapshot,.reserved=.my-reserved`.

If no key-value pairs are specified with `-renameReserved`, the NameNode will then suffix reserved paths with:

`.<LAYOUT-VERSION>.UPGRADE_RENAMED`

For example: `.snapshot.-51.UPGRADE_RENAMED`.

Note

We recommend that you perform a `-saveNamespace` before renaming paths (running `-saveNamespace` appears in a previous step in this procedure). This is because a data inconsistency can result if an edit log operation refers to the destination of an automatically renamed file.

Also note that running `-renameReserved` will rename all applicable existing files in the cluster. This may impact cluster applications.

14. If you are on JDK 1.6, upgrade the JDK on all nodes to JDK 1.7 or JDK 1.8 before upgrading HDP.

3.2. Upgrade HDP 2.1 Components

Important

See the Release Notes for the HDP 2.4.3.0 repo information.

The upgrade process to HDP-2.4.3 involves the following steps

RHEL/CentOS/Oracle 6

1. On all hosts, clean the yum repository.

   ```
   yum clean all
   ```

2. Remove your old HDP 2.1 components. This command uninstalls the HDP 2.1 components. It leaves the user data, and metadata, but removes your configurations:

```
yum erase "hadoop*" "webhcat*" "oozie*" "collectd*" "gccxml*" "pig*" "hdfs*"
 "sqoop*" "zookeeper*" "hbase*" "hive*" "tez*" "storm*" "falcon*" "flume*"
 "phoenix*" "accumulo*" "mahout*" "hue" "hue-common" "hue-shell" "knox*"
 "hdp_mon_nagios_addons"
```

3. Validate that all HDP 2.1 component binaries are uninstalled:

   ```
   yum list installed | grep @HDP2.1
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/yum.repos.d/hdp.repo
   ```

5. Install the HDP-2.4.3 repo:

   - Download the hdp.repo file:

     ```
     wget -nv http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.4.
     3.0/hdp.repo -O /etc/yum.repos.d/hdp.repo
     ```

   - Confirm the HDP repository is configured.

     ```
     yum repolist
     ```

     You should see something like this. Verify that you have the HDP-2.4.3 directory:

     ```
     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 namestatus
     HDP-2.4.3 Hortonworks Data Platform Version - HDP-2.4.3
     ```

6. Install the HDP-2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.1 components:

   

   Important

   For Falcon, we recommend upgrading from HDP 2.1 to HDP 2.2 and then upgrading HDP 2.2 to HDP 2.4. See Configure and Validate Falcon for more information.

   ```
   yum install "hadoop" "hadoop-hdfs" "hadoop-libhdfs" "hadoop-yarn"
    "hadoop-mapreduce" "hadoop-client" "openssl" "hive-webhcat" "hcatalog"
    "oozie" "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hue"
    "hive" "tez" "storm" "flume" "phoenix" "accumulo" "mahout" "knox"
    "hdp_mon_nagios_addons"
   ```

   

   Note

   If you installed Apache Argus, it is now Apache Ranger. See Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded.

   ```
   yum list installed | grep HDP-<old.stack.version.number>
   ```

No component file names should appear in the returned list.

RHEL/CentOS/Oracle 5 (Deprecated)

1. On all hosts, clean the yum repository.

   ```
   yum clean all
   ```

2. Remove your old HDP 2.1 components. This command uninstalls the HDP 2.1 components. It leaves the user data, and metadata, but removes your configurations:

   ```
   yum erase "hadoop*" "webhcat*" "oozie*" "collectd*" "gccxml*" "pig*" "hdfs*"
    "sqoop*" "zookeeper*" "hbase*" "hive*" "tez*" "storm*" "falcon*" "flume*"
    "phoenix*" "accumulo*" "mahout*" "hue" "hue-common" "hue-shell" "knox*"
    "hdp_mon_nagios_addons"
   ```

3. Validate that all HDP 2.1 component binaries are uninstalled:

   ```
   yum list installed | grep @HDP2.1
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/yum.repos.d/hdp.repo
   ```

5. Install the HDP-2.4.3 repo:

   - Download the hdp.repo file:

     ```
     wget -nv http://public-repo-1.hortonworks.com/HDP/centos5/2.x/updates/2.4.
     3.0/hdp.repo -O /etc/yum.repos.d/hdp.repo
     ```

   - Confirm the HDP repository is configured.

     ```
     yum repolist
     ```

     You should see something like this. Verify that you have the HDP-2.4.3 directory:

     ```
     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 namestatus
     HDP-2.4.3 Hortonworks Data Platform Version - HDP-2.4.3
     ```

6. Install the HDP-2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.1 components:

   

   Important

   For Falcon, we recommend upgrading from HDP 2.1 to HDP 2.2 and then upgrading HDP 2.2 to HDP 2.4. See Configure and Validate Falcon for more information.

   ```
   yum install "hadoop" "hadoop-hdfs" "hadoop-libhdfs" "hadoop-yarn"
    "hadoop-mapreduce" "hadoop-client" "openssl" "hive-webhcat" "hcatalog"
   ```

```
"oozie" "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hive"
"hue" "tez" "storm" "flume" "phoenix" "accumulo" "mahout" "knox"
"hdp_mon_nagios_addons"
```

Note

If you installed Apache Argus, it is now Apache Ranger. See Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded.

   ```
   yum list installed | grep HDP-<old.stack.version.number>
   ```

   No component file names should appear in the returned list.

SLES 11 SP 1

1. On all hosts, clean the yum repository.

   ```
   zypper clean -all
   ```

2. Remove your old HDP 2.1 components. This command uninstalls the HDP 2.1 components. It leaves the user data, and metadata, but removes your configurations:

   ```
   zypper rm "hadoop*" "webhcat*" "oozie*" "collectd*" "gccxml*" "pig*" "hdfs*"
    "sqoop*" "zookeeper*" "hbase*" "hive*" "tez*" "storm*" "falcon*" "flume*"
    "phoenix*" "accumulo*" "mahout*" "hue" "hue-common" "hue-shell" "knox*"
    "hdp_mon_nagios_addons"
   ```

3. Validate that all HDP 2.1 component binaries are uninstalled:

   ```
   yum list installed | grep @HDP2.1
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/zypp/repos.d/hdp.repo
   ```

5. Download the HDP-2.4.3 hdp.repo file:

   ```
   wget -nv http://public-repo-1.hortonworks.com/HDP/sles11sp1/2.x/updates/2.4.
   3.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
   ```

6. Install the HDP-2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.1 components:

   

   Important

   For Falcon, we recommend upgrading from HDP 2.1 to HDP 2.2 and then upgrading HDP 2.2 to HDP 2.4. See Configure and Validate Falcon for more information.

   ```
   zypper install "hadoop" "hadoop-hdfs" "hadoop-libhdfs" "hadoop-yarn"
    "hadoop-mapreduce" "hadoop-client" "openssl" "hive-webhcat" "hcatalog"
    "oozie" "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hive"
    "hue" "tez" "storm" "flume" "phoenix" "accumulo" "mahout" "knox"
    "hdp_mon_nagios_addons"
   ```

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

zypper up -r HDP-2.3

zypper install oozie-client
```

Note

If you installed Apache Argus, it is now Apache Ranger. See Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded. For example, to verify hdfs, hive, and hcatlog:

   ```
   rpm -qa | grep hdfs, && rpm -qa | grep hive && rpm -qa | grep
   hcatalog
   ```

   No component files names should appear in the returned list.

SLES 11 SP3/SP4

1. On all hosts, clean the yum repository.

   ```
   zypper clean -all
   ```

2. Remove your old HDP 2.1 components.

   ```
   zypper rm "hadoop*" "webhcat*" "hcatalog*" "oozie*" "collectd*" "gccxml*"
    "pig*" "hdfs*" "sqoop*" "zookeeper*" "hbase*" "hive*" "tez*" "storm*"
    "falcon*" "flume*" "phoenix*" "accumulo*" "mahout*" "hue" "hue-common"
    "hue-shell" "knox*" "hdp_mon_nagios_addons"
   ```

3. Validate that all HDP 2.1 component binaries are uninstalled:

   ```
   zypper search -i | grep @HDP2.1
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/zypp/repos.d/hdp.list
   ```

5. Download the HDP-2.4.3 hdp.repo file:

   ```
   wget http://public-repo-1.hortonworks.com/HDP/suse11sp3/2.x/
   updates/2.4.3.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
   ```

6. Install the HDP-2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.1 components:

Important

For Falcon, we recommend upgrading from HDP 2.1 to HDP 2.2 and then upgrading HDP 2.2 to HDP 2.4. See Configure and Validate Falcon for more information.

```
zypper install "hadoop" "hadoop-hdfs" "hadoop-libhdfs" "hadoop-yarn"
 "hadoop-mapreduce" "hadoop-client" "openssl" "hive-webhcat" "hcatalog"
 "oozie" "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hue"
 "hive" "tez" "storm" "flume" "phoenix" "accumulo" "mahout" "knox"
 "hdp_mon_nagios_addons"
```

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

```
zypper up -r HDP-2.3
```

```
zypper install oozie-client
```

Note

If you installed Apache Argus, it is now Apache Ranger. See Upgrade Ranger for more information on the upgrade path.

7. Verify that the components were upgraded. For example, to verify hdfs, hive, and hcatalog:

   ```
   rpm -qa | grep hdfs, && rpm -qa | grep hive && rpm -qa | grep
   hcatalog
   ```

   No component files names should appear in the returned list.

Ubuntu 12

1. On all hosts, clean the apt-get repository.

   ```
   apt-get clean -&-all
   ```

2. Remove your old HDP 2.1 components. This command uninstalls the HDP 2.1 components. It leaves the user data, and metadata, but removes your configurations:

   ```
   apt-get remove "hadoop*" "webhcat*" "hcatalog*" "oozie*" "collectd*"
    "gccxml*" "pig*" "hdfs*" "sqoop*" "zookeeper*" "hbase*" "hive*" "tez.
   *" "storm*" "falcon*" "flume*" "phoenix*" "accumulo*" "mahout*" "hue.*"
    "knox*" "hdp_mon_nagios_addons" --purge
   ```

3. Validate that all HDP 2.1 component binaries are uninstalled:

   ```
   yum list installed | grep @HDP2.1
   ```

4. Remove your old hdp.repo file:

   ```
   rm /etc/apt/sources.list.d/hdp.list
   ```

5. Download the HDP-2.4.3 hdp.repo file:

   ```
   wget http://public-repo-1.hortonworks.com/HDP/ubuntu12/2.x/
   updates/2.4.3.0/hdp.list - O /etc/apt/sources.list.d/hdp.list
   ```

6. Run an update:

   ```
   apt-get update
   ```

7. Install the HDP-2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.1 components:

Important

For Falcon, we recommend upgrading from HDP 2.1 to HDP 2.2 and then upgrading HDP 2.2 to HDP 2.4. See Configure and Validate Falcon for more information.

```
apt-get install "hadoop" "hadoop-hdfs" "libhdfs0" "hadoop-yarn" "hadoop-
mapreduce" "hadoop-client" "openssl" "hive-webhcat" "hcatalog" "oozie"
 "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hive" "tez" "storm"
 "flume" "phoenix" "accumulo" "mahout" "knox" "hdp_mon_nagios_addons"
```

Note

If you installed Apache Argus, it is now Apache Ranger. See Upgrade Ranger for more information on the upgrade path.

Debian 6 (Deprecated)

HDP support for Debian 6 is deprecated with HDP 2.4.2. Future versions of HDP will no longer be supported on Debian 6.

1. On all hosts, clean the apt-get repository.

```
apt-get clean -&-all
```

2. Remove your old HDP 2.1 components. This command uninstalls the HDP 2.1 components. It leaves the user data, and metadata, but removes your configurations:

```
apt-get remove "hadoop*" "webhcat*" "hcatalog*" "oozie*" "collectd*"
 "gccxml*" "pig*" "hdfs*" "sqoop*" "zookeeper*" "hbase*" "hive*" "tez*"
 "storm*" "falcon*" "flume*" "phoenix*" "accumulo*" "mahout*" "hue*" "knox*"
 "hdp_mon_nagios_addons"
```

3. Validate that all HDP 2.1 component binaries are uninstalled:

```
yum list installed | grep @HDP2.1
```

4. Remove your old hdp.repo file:

```
rm /etc/apt/sources.list.d/hdp.list
```

5. Download the HDP-2.4.3 hdp.repo file:

```
wget -nv http://public-repo-1.hortonworks.com/HDP/debian6/2.x/
updates/2.4.3.0/hdp.list - O /etc/apt/sources.list.d/hdp.list
```

6. Run an update:

```
apt-get update
```

7. Install the HDP-2.4.3 versions of the components that you want to upgrade. For example, if you installed and want to upgrade all HDP 2.1 components:

Important

For Falcon, we recommend upgrading from HDP 2.1 to HDP 2.2 and then upgrading HDP 2.2 to HDP 2.4. See Configure and Validate Falcon for more information.

```
apt-get install "hadoop" "hadoop-hdfs" "libhdfs0" "hadoop-yarn" "hadoop-
mapreduce" "hadoop-client" "openssl" "hive-webhcat" "hcatalog" "oozie"
 "collectd" "gccxml" "pig" "sqoop" "zookeeper" "hbase" "hive" "tez" "storm"
 "flume" "phoenix" "accumulo" "mahout" "knox" "hdp_mon_nagios_addons"
```

Note

If you installed Apache Argus, it is now Apache Ranger. See Upgrade Ranger for more information on the upgrade path.

Install the Compression Libraries. See Install Compression Libraries.

3.3. Symlink Directories with hdp-select

Warning

HDP-2.4.3 installs hdp-select automatically with the installation or upgrade of the first HDP component. If you have not already upgraded ZooKeeper, hdp-select has not been installed.

To prevent version-specific directory issues for your scripts and updates, Hortonworks provides hdp-select, a script that symlinks directories to hdp-current and modifies paths for configuration directories.

1. Before you run hdp-select, remove one link:

   ```
   rm /usr/bin/oozie
   ```

2. Run hdp-select set all on your NameNode and all your DataNodes:

   ```
   hdp-select set all 2.4-$BUILD
   ```

   For example:

   ```
   /usr/bin/hdp-select set all 2.4-$BUILD
   ```

3.4. Configure and Start Apache ZooKeeper

Before you can upgrade Apache ZooKeeper, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Tip

If you are running a highly available HDFS cluster, configure and restart ZooKeeper **before** you upgrade HDFS. This best practice lets the upgraded ZKFC work with your primary NameNode and your Standby NameNode.

1. Replace your configuration after upgrading on all the ZooKeeper nodes. Replace the ZooKeeper template configuration in `/etc/zookeeper/conf`.

2. Start ZooKeeper.

   On all ZooKeeper server host machines, run the following command to start ZooKeeper and the ZKFC:

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

3.5. Configure Hadoop

RHEL/CentOS/Oracle Linux

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml.

2. Paths have changed in HDP-2.4.3. Make sure you remove old path specifications from hadoop-env.sh, such as:

   `export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-amd64-64`

   If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

SLES

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml.

2. Paths have changed in HDP-2.4.3. Make sure you remove old path specifications from hadoop-env.sh, such as:

   `export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-amd64-64`

   If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

Ubuntu/Debian

HDP support for Debian 6 is deprecated with HDP 2.4.2. Future versions of HDP will no longer be supported on Debian 6.

1. Use the HDP Utility script to calculate memory configuration settings. You must update the memory/cpu settings in yarn-site.xml and mapred-site.xml

2. Paths have changed in HDP-2.4.3. make sure you remove old path specifications from hadoop-env.sh, such as:

   ```
   export JAVA_LIBRARY_PATH=/usr/lib/hadoop/lib/native/Linux-amd64-64
   ```

   If you leave these paths in your hadoop-env.sh file, the lzo compression code will not load, as this is not where lzo is installed.

3.6. Start Hadoop Core

Warning

Before you start HDFS on a highly available HDFS cluster, you must start the ZooKeeper service. If you do not start the ZKFC, there can be failures.

To start HDFS, run commands as the $HDFS_USER.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

1. Replace your configuration after upgrading on all the HDFS nodes. Replace the HDFS template configuration in `/etc/hdfs/conf`.

2. If you are upgrading from a highly available HDFS cluster configuration, start all JournalNodes. On each JournalNode host, run the following commands:

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

   

   ### Important

   All JournalNodes must be running when performing the upgrade, rollback, or finalization operations. If any JournalNodes are down when running any such operation, the operation fails.

3. If you are running HDFS on a highly available namenode, you must first start the ZooKeeper service.

   

   ### Note

   Perform this step only if you are on a highly available HDFS cluster.

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

4. Start the NameNode.

   Because the file system version has now changed you must start the NameNode manually.

   On the active NameNode host, run the following commands:

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

   On a large system, this can take a long time to complete.

   

   Note

   Run this command with the -upgrade option only once. After you have completed this step, you can bring up the NameNode using this command without including the -upgrade option.

   To check if the Upgrade is in progress, check that the "\previous" directory has been created in the \NameNode and \JournalNode directories. The "\previous" directory contains a snapshot of the data before upgrade.

   In a highly available HDFS cluster configuration, this NameNode will not enter the standby state as usual. Rather, this NameNode will immediately enter the active state, perform an upgrade of its local storage directories, and also perform an upgrade of the shared edit log. At this point, the standby NameNode in the HA pair is still down. It will be out of sync with the upgraded active NameNode.

   To synchronize the active and standby NameNode, re-establishing HA, re-bootstrap the standby NameNode by running the NameNode with the '-bootstrapStandby' flag. Do NOT start this standby NameNode with the '-upgrade' flag.

   ```
   su - hdfs -c "hdfs namenode -bootstrapStandby -force"
   ```

   The bootstrapStandby command will download the most recent fsimage from the active NameNode into the $dfs.name.dir directory of the standby NameNode. You can enter that directory to make sure the fsimage has been successfully downloaded. After verifying, start the ZKFailoverController, then start the standby NameNode. You can check the status of both NameNodes using the Web UI.

5. Verify that the NameNode is up and running:

   ```
   ps -ef|grep -i NameNode
   ```

6. If you do not have a highly available HDFS cluster configuration (non_HA namenode), start the Secondary NameNode.

   

   Note

   Do not perform this step if you have a highly available HDFS cluster configuration.

   On the Secondary NameNode host machine, run the following commands:

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

7. Verify that the Secondary NameNode is up and running.

Note

Do not perform this step if you have a highly available HDFS cluster environment.

```
ps -ef|grep SecondaryNameNode
```

8. Start DataNodes.

   On each of the DataNodes, enter the following command. Note: If you are working on a `non-secure` DataNode, use $HDFS_USER. For a secure DataNode, use root.

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

9. Verify that the DataNode process is up and running:

```
ps -ef|grep DataNode
```

10. Verify that NameNode can go out of safe mode.

```
>su - hdfs -c "hdfs dfsadmin -safemode wait"
```

   You should see the following result: `Safe mode is OFF`

   In general, it takes 5-10 minutes to get out of safemode. For thousands of nodes with millions of data blocks, getting out of safemode can take up to 45 minutes.

3.7. Verify HDFS Filesystem Health

Analyze if the filesystem is healthy.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Important

If you have a secure server, you will need Kerberos credentials for hdfs user access.

1. Run the fsck command on namenode as $HDFS_USER:

```
su - hdfs -c "hdfs fsck / -files -blocks -locations > dfs-new-
fsck-1.log"
```

You should see feedback that the filesystem under path / is HEALTHY.

2. Run `hdfs` namespace and report.

   a. List directories.

      ```
      su - hdfs -c "hdfs dfs -ls -R / > dfs-new-lsr-1.log"
      ```

   b. Open the `dfs-new-lsr-l.log` and confirm that you can see the file and directory listing in the namespace.

   c. Run report command to create a list of DataNodes in the cluster.

      ```
      su - hdfs -c "hdfs dfsadmin -report > dfs-new-report-1.log"
      ```

   d. Open the `dfs-new-report` file and validate the admin report.

3. Compare the namespace report before the upgrade and after the upgrade. Verify that user files exist after upgrade.

   The file names are listed below:

   ```
   dfs-old-fsck-1.log < -- > dfs-new-fsck-1.log
   ```

   ```
   dfs-old-report-1.log < -- > dfs-new-report-1.log
   ```

   

   Note

   You must do this comparison manually to catch all errors.

4. From the NameNode WebUI, determine if all DataNodes are up and running.

   ```
   http://<namenode>:<namenodeport>
   ```

5. If you are on a highly available HDFS cluster, go to the StandbyNameNode web UI to see if all DataNodes are up and running:

   ```
   http://<standbynamenode>:<namenodeport>
   ```

6. If you are **not** on a highly available HDFS cluster, go to the SecondaryNameNode web UI to see if it the secondary node is up and running:

   ```
   http://<secondarynamenode>:<secondarynamenodeport>
   ```

7. Verify that read and write to hdfs works successfully.

   ```
   hdfs dfs -put [input file] [output file]
   ```

   ```
   hdfs dfs -cat [output file]
   ```

3.8. Configure YARN and MapReduce

After you upgrade Hadoop, complete the following steps to update your configs.

Note

The su commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the su commands.

Important

If you have a secure server, you will need Kerberos credentials for hdfs user access.

1. Ensure that all HDFS directories configured in yarn-site.xml configuration files (for example, yarn.timeline-service.entity-group-fs-store.active-dir, yarn.timeline-service.entity-group-fs-store.done-dir) in HDFS.
2. Upload the MapReduce tarball to HDFS. As the HDFS user, for example 'hdfs':

   ```
   su - hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/
   mapreduce/"

   su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/hadoop/
   mapreduce.tar.gz /hdp/apps/2.4.3.0-$BUILD/mapreduce/"

   su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp"

   su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/
   mapreduce"

   su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/
   mapreduce/mapreduce.tar.gz"
   ```

3. Make the following changes to /etc/hadoop/conf/mapred-site.xml:

   - Add:

     ```
     <property>
      <name>mapreduce.application.framework.path</name>
      <value>/hdp/apps/${hdp.version}
        /mapreduce/mapreduce.tar.gz#mr-framework
      </value>
     </property>

     <property>
      <name>yarn.app.mapreduce.am.admin-comand-opts</name>
      <value>Dhdp.version=${hdp.version}</value>
     </property>
     ```

     

     #### Note

     You do not need to modify ${hdp.version}.

   - Modify the following existing properties to include ${hdp.version}:

     ```
     <property>
      <name>mapreduce.admin.user.env</name>
     ```

```
 <value>LD_LIBRARY_PATH=/usr/hdp/${hdp.version}
   /hadoop/lib/native:/usr/hdp/${hdp.version}/hadoop/
     lib/native/Linux-amd64-64
   </value>
</property>

<property>
 <name>mapreduce.admin.map.child.java.opts</name>
 <value>-server -Djava.net.preferIPv4Stack=true
   -Dhdp.version=${hdp.version}
   </value>
 <final>true</final>
</property>

<property>
 <name>mapreduce.admin.reduce.child.java.opts</name>
 <value>-server -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.
version}</value>
 <final>true</final>
</property>

<property>
 <name>mapreduce.application.classpath</name>
 <value>$PWD/mr-framework/hadoop/share/hadoop/mapreduce/*,
   $PWD/mr-framework/hadoop/share/hadoop/mapreduce/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/common/*,
   $PWD/mr-framework/hadoop/share/hadoop/common/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/yarn/*,
   $PWD/mr-framework/hadoop/share/hadoop/yarn/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/hdfs/*,
   $PWD/mr-framework/hadoop/share/hadoop/hdfs/lib/*,
  /usr/hdp/${hdp.version}/hadoop/lib/hadoop-lzo-0.6.0.${hdp.version}.jar,
  /etc/hadoop/conf/secure</value>
</property>
```

Note

You do not need to modify ${hdp.version}.

- Remove the following properties from `/etc/hadoop/conf/mapred-site.xml`: `mapreduce.task.tmp.dir`, `mapreduce.job.speculative.slownodethreshold` (deprecated), and `mapreduce.job.speculative.speculativecap` (deprecated).

4. Add the following properties to `/etc/hadoop/conf/yarn-site.xml`:

```
<property>
 <name>yarn.application.classpath</name>
 <value>$HADOOP_CONF_DIR,/usr/hdp/${hdp.version}/hadoop-client/*,
  /usr/hdp/${hdp.version}/hadoop-client/lib/*,
  /usr/hdp/${hdp.version}/hadoop-hdfs-client/*,
  /usr/hdp/${hdp.version}/hadoop-hdfs-client/lib/*,
  /usr/hdp/${hdp.version}/hadoop-yarn-client/*,
  /usr/hdp/${hdp.version}/hadoop-yarn-client/lib/*</value>
</property>
```

5. On secure clusters only, add the following properties to `/etc/hadoop/conf/yarn-site.xml`:

```
 <property>
  <name>yarn.timeline-service.recovery.enabled:</name>
  <value>TRUE</value>
  </property>

  <property>
  <name>yarn.timeline-service.state-store.class: org.apache.hadoop.yarn.
server.timeline.recovery:</name>
  <value>LeveldbTimelineStateStore</value>
  </property>

  <property>
  <name>yarn.timeline-service.leveldb-state-store.path:</name>
  <value><the same as the default of "yarn.timeline-service-leveldb-
timeline-store.path</value>
  </property>
```

6. Modify the following property to `/etc/hadoop/conf/yarn-site.xml`:

```
<property>
 <name>mapreduce.application.classpath</name>
 <value>$PWD/mr-framework/hadoop/share/hadoop/mapreduce/*,
   $PWD/mr-framework/hadoop/share/hadoop/mapreduce/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/common/*,
   $PWD/mr-framework/hadoop/share/hadoop/common/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/yarn/*,
   $PWD/mr-framework/hadoop/share/hadoop/yarn/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/hdfs/*,
   $PWD/mr-framework/hadoop/share/hadoop/hdfs/lib/*,
   $PWD/mr-framework/hadoop/share/hadoop/share/hadoop/tools/lib/*,
  /usr/hdp/${hdp.version}/hadoop/lib/hadoop-lzo-0.6.0.${hdp.version}.jar,
  /etc/hadoop/conf/secure</value>
</property>
```

7. Make the following change to the `/etc/hadoop/conf/yarn-env.sh`:

   Change `export HADOOP_YARN_HOME=/usr/lib/hadoop-yarn`

   to

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

8. Make the following change to the `/etc/hadoop/conf/yarn-env.sh`:

   Change

   `export HADOOP_LIBEXEC_DIR=/usr/lib/hadoop/libexec`

   to

   `HADOOP_LIBEXEC_DIR=/usr/hdp/current/hadoop-client/libexec/`

9. For secure clusters, you must create and configure the container-executor.cfg configuration file:

   - Create the container-executor.cfg file in /etc/hadoop/conf/.

- Insert the following properties:

```
yarn.nodemanager.linux-container-executor.group=hadoop
banned.users=hdfs,yarn,mapred
min.user.id=1000
```

  - yarn.nodemanager.linux-container-executor.group - Configured value of yarn.nodemanager.linux-container-executor.group. This must match the value of yarn.nodemanager.linux-container-executor.group in yarn-site.xml.
  - banned.users - Comma-separated list of users who can not run container-executor.
  - min.user.id - Minimum value of user id. This prevents system users from running container-executor.
  - allowed.system.users - Comma-separated list of allowed system users.

- Set the file /etc/hadoop/conf/container-executor.cfg file permissions to only be readable by root:

```
chown root:hadoop /etc/hadoop/conf/container-executor.cfg
chmod 400 /etc/hadoop/conf/container-executor.cfg
```

- Set the container-executor program so that only root or hadoop group users can run it:

```
chown root:hadoop /usr/hdp/${hdp.version}/hadoop-yarn-server-nodemanager/
bin/container-executor

chmod 6050 /usr/hdp/${hdp.version}/hadoop-yarn-server-nodemanager/bin/
container-executor
```

3.9. Start YARN/MapReduce Services

To start YARN, run commands as a YARN user. To start MapReduce, run commands as a MapReduce user.

Note

The `su` commands in this section use "yarn" to represent the YARN Service user and mapreduce to represent the MAPREDUCE Service user. If you are using another name for these Service users, you will need to substitute your Service user name for "yarn" or "mapreduce" in each of the `su` commands.

1. If you have a highly available HDFS cluster configuration, manually clear the ResourceManager state store.

```
su - yarn -c "yarn resourcemanager -format-state-store"
```

2. Start the ResourceManager on all your ResourceManager hosts.

```
su - yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.
sh start resourcemanager"

ps -ef | grep -i resourcemanager
```

3. Start the TimelineServer on your TimelineServer host.

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

4. Start the NodeManager on all your NodeManager hosts.

```
su - yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh
 start nodemanager"

ps -ef | grep -i nodemanager
```

5. To start MapReduce, run the following commands:

```
su - mapreduce -c "/usr/hdp/current/hadoop-mapreduce-historyserver/sbin/mr-
jobhistory-daemon.sh start historyserver"

ps -ef | grep -i jobhistoryserver
```

3.10. Run Hadoop Smoke Tests

To smoke test your Hadoop upgrade, you can run the following MapReduce job as a regular user.

The job uses MapReduce to write 100MB of data into HDFS with RandomWriter

```
hadoop jar /usr/hdp/current/hadoop-mapreduce-client/hadoop-mapreduce-examples.
jar
            randomwriter -Dtest.randomwrite.total_bytes=10000000 test-after-
upgrade.
```

You should see messages similar to:

```
map 0% reduce 0%
…map 100% reduce 100%
Job …. completed successfully
```

MapReduce upgraded successfully. You can now upgrade your other components.

Basic Troubleshooting

To find the number of active nodes and NodeManagers, access the ResourceManager web UI:

`http://<resource manager host>:8088/cluster/nodes`

The number of active nodes should be equal to the number of nodemanagers.

Accessing error messages:

1. Access the ApplicationMaster WebUI to view the container logs.
2. At your console logs for MapReduce job, look for a line with this format:

   13/10/02 17:57:21 INFO mapreduce.Job: The url to track the job: http://<resource manager host>:8088/proxy/application_1380673658357_0007/
3. Select the logs link under ApplicationMaster table. It will redirect you to the container logs. Error messages display here.

3.11. Configure and Start Apache HBase

Before you can upgrade Apache HBase, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "hbase" to represent the HBASE Service user. If you are using another name for your HBASE Service user, you will need to substitute your HBASE Service user name for "hbase" in each of the `su` commands.

The hbase.bucketcache.percentage.in.combinedcache is removed in HDP-2.4.3. This simplifies the configuration of block cache. BucketCache configurations from HDP 2.1 will need to be recalculated to attain identical memory allotments in HDP-2.4.3. The L1 LruBlockCache will be whatever hfile.block.cache.size is set to and the L2 BucketCache will be whatever hbase.bucketcache.size is set to.

1. Replace your configuration after upgrading. Replace the HBase template configuration in `/etc/hbase/conf`.
2. Start services. From root, assuming that $HBASE_USER=hbase:

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

   ```
   su - hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-
   daemon.sh start regionserver"
   ```

3. Check processes.

   ```
   ps -ef | grep -i hmaster
   ```

   ```
   ps -ef | grep -i hregion
   ```

3.12. Configure Apache Phoenix

Before you can upgrade Apache Phoenix, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

To configure Phoenix, complete the following steps:

1. Add the following property to the `/etc/hbase/hbase-site.xml` file on all HBase nodes, the MasterServer, and all RegionServers to prevent deadlocks from occurring during maintenance on global indexes:

```
<property>
  <name>hbase.regionserver.wal.codec</name>
  <value>org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec</
value>
</property>
```

2. To enable user-defined functions, configure the following property in `/etc/hbase/conf` on all Hbase nodes.

```
<property>
 <name>phoenix.functions.allowUserDefinedFunctions</name>
 <value>true</value>
 <description>enable UDF functions</description>
</property>
```

3. Ensure the client side hbase-site.xml matches the server side configuration.

4. If the folder specified in `hbase.tmp.dir` property on `hbase-site.xml` does not exist, create that directory with adequate permissions.

5. Set the following porperty in the hbase-site.xml file for all RegionServers, but not on the client side:

```
<property>
  <name>hbase.rpc.controllerfactory.class</name>
  <value>org.apache.hadoop.hbase.ipc.controller.ServerRpcControllerFactory</
value>
</property>
```

6. Restart the HBase Master and RegionServers.

Configuring Phoenix to Run in a Secure Cluster

Perform the following additional steps to configure Phoenix to run in a secure Hadoop cluster:

1. To link the HBase configuration file with the Phoenix libraries:

   ```
   ln -sf HBASE_CONFIG_DIR/hbase-site.xml PHOENIX_HOME/bin/hbase-
   site.xml
   ```

2. To link the Hadoop configuration file with the Phoenix libraries:

   ```
   ln -sf HADOOP_CONFIG_DIR/core-site.xml PHOENIX_HOME/bin/core-
   site.xml
   ```

Note

When running the psql.py and sqlline.py Phoenix scripts in secure mode, you can safely ignore the following warnings.

```
14/04/19 00:56:24 WARN util.NativeCodeLoader:
Unable to load native-hadoop library for your platform...
  using builtin-java classes where applicable

14/04/19 00:56:24 WARN util.DynamicClassLoader: Failed to identify the fs of
dir hdfs://<HOSTNAME>:8020/apps/hbase/data/lib, ignored java.io.IOException:
No FileSystem for scheme: hdfs
```

3.13. Configure and Start Apache Accumulo

Before you can upgrade Apache Accumulo, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "accumulo" to represent the Accumulo Service user. If you are using another name for your Accumulo Service user, you will need to substitute your Accumulo Service user name for "accumulo" in each of the `su` commands.

Upon upgrade from HDP 2.1 to HDP-2.4.3, Accumulo automatically changes the HDFS and ZooKeeper data stored on Accumulo. This change is not backward compatible, and HDP 2.1 will not run on this updated data.

After upgrading from HDP 2.1 to HDP-2.4.3, Accumulo automatically upgrades the internal metadata, notably the data in ZooKeeper, when the Accumulo Master for HDP-2.4.3 first starts. This change is not backward compatible and HDP 2.1 will no longer run against the data.

Prior to performing the following steps, you need to initialize Accumulo. See Initialization.

1. You must replace your configuration after upgrading. Copy `/etc/accumulo/conf` from the template to the conf directory in Accumulo hosts.

2. In HDP-2.4.3, the instance.dfs.dir and instance.dfs.uri properties are deprecated with the instance.volumes property. If it does not already exist, add the instance.volumes property to the accumulo-site.xml file. Do **not** remove the instance.dfs.dir and instance.dfs.uri properties. You can extrapolate the value for the instance.volumes property from the instance.dfs.dir and instance.dfs.uri properties.

   For example:

   ```
   <property>
      <name>instance.dfs.dir</name>
      <value>/accumulo</value>
   </property>
   <property>
      <name>instance.dfs.uri</name>
      <value>hdfs://my_namenode:8020</value>
   </property>
   <property>
      <name>instance.volumes</name>
      <value>hdfs://my_namenode:8020/accumulo</value>
   </property>
   ```

3. Start the services:

   ```
   su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
    `hostname` master"
   ```

```
su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` tserver"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` gc"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` tracer"

su - accumulo -c "/usr/hdp/current/accumulo-master/bin/start-server.sh
 `hostname` monitor"
```

4. Check that the processes are running

   ```
   ps -ef | grep accumulo
   ```

   or visit http://<hostname>:50095 in your browser

3.14. Configure and Start Apache Tez

Before you can upgrade Apache Tez, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Important

If you have a secure server, you will need Kerberos credentials for hdfs user access.

To upgrade Tez:

1. Copy your previously backed-up copy of tez-site.xml into the `/etc/tez/conf` directory.

2. Check to see if the HDP-2.4.3 Tez tarball libraries are already in the `/hdp/apps/<hdp 2.3>/tez/` version> directory - if so, then skip this step. If not, put the HDP-2.4.3 Tez tarball libraries in the `/hdp/apps` directory in HDFS, so that submitted Tez applications to this cluster will have access to these shared Tez tarball libraries. Execute this step on any host that has the Tez client installed.

   ```
   su - hdfs
   hdfs dfs -mkdir -p /hdp/apps/<hdp_version>/tez/
   hdfs dfs -put /usr/hdp/<hdp_version>/tez/lib/tez.tar.gz /hdp/apps/
   <hdp_version>/tez/
   hdfs dfs -chown -R hdfs:hadoop /hdp
   hdfs dfs -chmod -R 555 /hdp/apps/<hdp_version>/tez
   ```

```
hdfs dfs -chmod -R 444 /hdp/apps/<hdp_version>/tez/tez.tar.gz
```

Where <hdp_version> is the current HDP version, for example 2.4.3-258.

3. Edit the tez.lib.uris property in the tez-site.xml file to point to /hdp/apps/<hdp_version>/tez/tez.tar.gz

```
...
<property>
 <name>tez.lib.uris</name>
 <value>/hdp/apps/<hdp_version>/tez/tez.tar.gz</value>
</property>
...
```

Where <hdp_version> is the current HDP version, for example 2.4.3-258.

4. **Optional** Earlier releases of Tez did not have access control. In the current version of Tez, the default behavior restricts the ability to view the Tez history to only the owner of the job. To retain unrestricted access for non-secure clusters, set tez.am.view-acls set to "*".

5. Update the following tez-site.xml properties to their new names.

Old Property Name	New Property Name
tez.am.java.opts	tez.am.launch.cmd-opts
tez.am.env	tez.am.launch.env
tez.am.shuffle-vertex-manager.min-src-fraction	tez.shuffle-vertex-manager.min-src-fraction
tez.am.shuffle-vertex-manager.max-src-fraction	tez.shuffle-vertex-manager.max-src-fraction
tez.am.shuffle-vertex-manager.enable.auto-parallel	tez.shuffle-vertex-manager.enable.auto-parallel
tez.am.shuffle-vertex-manager.desired-task-input-size	tez.shuffle-vertex-manager.desired-task-input-size
tez.am.shuffle-vertex-manager.min-task-parallelism	tez.shuffle-vertex-manager.min-task-parallelism
tez.am.grouping.split-count	tez.grouping.split-count
tez.am.grouping.by-length	tez.grouping.by-length
tez.am.grouping.by-count	tez.grouping.by-count
tez.am.grouping.max-size	tez.grouping.max-size
tez.am.grouping.min-size	tez.grouping.min-size
tez.am.grouping.rack-split-reduction	tez.grouping.rack-split-reduction
tez.am.am.complete.cancel.delegation.tokens	tez.cancel.delegation.tokens.on.completion
tez.am.max.task.attempts	tez.am.task.max.failed.attempts
tez.generate.dag.viz	tez.generate.debug.artifacts
tez.runtime.intermediate-output.key.comparator.class	tez.runtime.key.comparator.class
tez.runtime.intermediate-output.key.class	tez.runtime.key.class
tez.runtime.intermediate-output.value.class	tez.runtime.value.class
tez.runtime.intermediate-output.should-compress	tez.runtime.compress
tez.runtime.intermediate-output.compress.codec	tez.runtime.compress.codec
tez.runtime.intermediate-input.key.secondary.comparator.class	tez.runtime.key.secondary.comparator.class
tez.runtime.broadcast.data-via-events.enabled	tez.runtime.transfer.data-via-events.enabled
tez.runtime.broadcast.data-via-events.max-size	tez.runtime.transfer.data-via-events.max-size
tez.runtime.shuffle.input.buffer.percent	tez.runtime.shuffle.fetch.buffer.percent
tez.runtime.task.input.buffer.percent	tez.runtime.task.input.post-merge.buffer.percent

Old Property Name	New Property Name
tez.runtime.job.counters.max	tez.am.counters.max.keys
tez.runtime.job.counters.group.name.max	tez.am.counters.group-name.max.keys
tez.runtime.job.counters.counter.name.max	tez.am.counters.name.max.keys
tez.runtime.job.counters.groups.max	tez.am.counters.groups.max.keys
tez.task.merge.progress.records	tez.runtime.merge.progress.records
tez.runtime.metrics.session.id	tez.runtime.framework.metrics.session.id
tez.task.scale.memory.additional.reservation.fraction.per-io	tez.task.scale.memory.additional-reservation.fraction.per-io
tez.task.scale.memory.additional.reservation.fraction.max	tez.task.scale.memory.additional-reservation.fraction.max
tez.task.initial.memory.scale.ratios	tez.task.scale.memory.ratios
tez.resource.calculator.process-tree.class	tez.task.resource.calculator.process-tree.class

For more information on setting Tez configuration parameters in HDP-2.4.3, see Installing and Configuring Apache Tez in the Non-Ambari Cluster Installation Guide.

6. Change the value of the tez.tez-ui.history-url.base property to the url for the upgraded Tez View. For more details, see Using the Tez View in the HDP Ambari Views Guide.

3.15. Configure and Start Apache Hive and Apache HCatalog

Before you can upgrade Apache Hive and Apache HCatalog, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "hive" to represent the Hive Service user. If you are using another name for your Hive Service user, you will need to substitute your Hive Service user name for "hive" in each of the `su` commands.

1. Prior to starting the upgrade process, set the following in your hive configuration file:

```
datanucleus.autoCreateSchema=false
```

2. Copy the jdbc connector jar from OLD_HIVE_HOME/lib to CURRENT_HIVE_HOME/lib.

3. Restore the JDBC jar files backed up into the $HIVE_HOME/lib directory. Make sure to restore all Metastore-related properties, (such as, ConnectionURL, user etc), from your older hive installation.

4. Upgrade the Hive Metastore database schema. Restart the Hive Metastore database and run:

```
su - hive -c "/usr/hdp/current/hive-metastore/bin/schematool -upgradeSchema
 -dbType <$databaseType>"
```

The value for $databaseType can be derby, mysql, oracle, or postgres.

Important

When you use MySQL as your Hive metastore, you must use `mysql-connector-java-5.1.35.zip` or later JDBC driver.

Note

If you are using Postgres 8 and Postgres 9, you should reset the Hive Metastore database owner to <HIVE_USER>:

```
sudo <POSTGRES_USER>
```

Start the Postgres CLU using the `psql` command.

Execute: `ALTER DATABASE <HIVE-METASTORE-DB-NAME> OWNER TO <HIVE_USER>`

Note

If you are using Oracle 11, you might see the following error message:

```
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
optimize.mapjoin.mapreduce does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
heapsize does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
server2.enable.impersonation does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.
semantic.analyzer.factory.impl does not exist
14/11/17 14:11:38 WARN conf.HiveConf: HiveConf of name hive.auto.
convert.sortmerge.join.noconditionaltask does not exist
Metastore connection URL: jdbc:oracle:thin:@//ip-172-31-42-1.ec2.
internal:1521/XE
Metastore Connection Driver : oracle.jdbc.driver.OracleDriver
Metastore connection User: hiveuser
Starting upgrade metastore schema from version 0.13.0 to 0.14.0
Upgrade script upgrade-0.13.0-to-0.14.0.oracle.sql
Error: ORA-00955: name is already used by an existing object
 (state=42000,code=955)
Warning in pre-upgrade script pre-0-upgrade-0.13.0-to-0.14.0.
oracle.sql: Schema script failed, errorcode 2
Completed upgrade-0.13.0-to-0.14.0.oracle.sql
schemaTool completed
```

You can safely ignore this message. The error is in the pre-upgrade script and can be ignored; the schematool succeeded.

Note

Copy only the necessary configuration files. Do not copy the env.sh files, for example, hadoop-env.sh, hive-env.sh, and so forth. Additionally, all env.sh files must be properly configured.

5. Edit the hive-site.xml file and modify the properties based on your environment.

a. Edit the following properties in the hive-site.xml file:

```
<property>
 <name>fs.file.impl.disable.cache</name>
 <value>false</value>
 <description>Set to false or remove fs.file.impl.disable.cache</
description>
</property>

<property>
 <name>fs.hdfs.impl.disable.cache</name>
 <value>false</value>
 <description>Set to false or remove fs.hdfs.impl.disable.cache
 </description>
<property>
```

b. **Optional:** To enable the Hive buildin authorization mode, make the following changes. If you want to use the advanced authorization provided by Ranger, refer to the Ranger instructions.

Set the following Hive authorization parameters in the hive-site.xml file:

```
<property>
 <name>hive.server2.enable.doAs</name>
 <value>false</value>
</property>

<property>
 <name>hive.security.metastore.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.
   StorageBasedAuthorizationProvider,org.apache.hadoop.hive.ql.security.
authorization.MetaStoreAuthzAPIAuthorizeEmbedOnly</value>
</property>

<property>
 <name>hive.security.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.
SQLStdConfOnlyAuthorizeFactory</value>
</property>
```

Also set hive.users.in.admin.role to the list of comma-separated users who need to be added to admin role. A user who belongs to the admin role needs to run the "set role" command before getting the privileges of the admin role, as this role is not in the current roles by default.

Set the following in the hiveserver2-site.xml file.

```
<property>
 <name>hive.security.authenticator.manager</name>
 <value>org.apache.hadoop.hive.ql.security.
SessionStateUserAuthenticator</value>
</property>

<property>
 <name>hive.security..authorization.enabled</name>
 <value>true</value>
</property>
```

```
<property>
 <name>hive.security.authorization.manager</name>
 <value>org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.
SQLStdHiveAuthorizeFactory/value>
</property>
```

c. For a remote Hive metastore database, set the IP address (or fully-qualified domain name) and port of the metastore host using the following hive-site.xml property value.

```
<property>
 <name>hive.metastore.uris</name>
 <value>thrift://$metastore.server.full.hostname:9083</value>
 <description>URI for client to contact metastore server.
   To enable HiveServer2, leave the property value empty.
   </description>
</property>
```

You can further fine-tune your configuration settings based on node hardware specifications, using the HDP utility script.

6. Start Hive Metastore.

   On the Hive Metastore host machine, run the following command:

   ```
   su - hive -c "nohup /usr/hdp/current/hive-metastore/bin/hive
   --service metastore -hiveconf hive.log.file=hivemetastore.log
   >/var/log/hive/hivemetastore.out 2>/var/log/hive/
   hivemetastoreerr.log &"
   ```

7. Start Hive Server2.

   On the Hive Server2 host machine, run the following command:

   ```
   su - hive

   nohup /usr/hdp/current/hive-server2/bin/hiveserver2 -hiveconf
   hive.metastore.uris=" " -hiveconf hive.log.file=hiveserver2.log
   >/var/log/hive/hiveserver2.out 2> /var/log/hive/
   hiveserver2err.log &
   ```

3.16. Configure and Start Apache Oozie

Before you can upgrade Apache Oozie, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The duration of the Oozie upgrade is dependent on the amount of job history stored in ooziedb. This history must be backed up and restored during the upgrade process. Best practice when planning for upgrade is to backup ooziedb

from your production oozie server and restore it to a test or development oozie server. This can help you estimate the time that will be required to upgrade Oozie during your production upgrade.

Note

The `su` commands in this section use "hdfs" to represent the HDFS Service user and "oozie" to represent the Oozie Service user. If you are using another name for your HDFS Service user or your Oozie Service user, you will need to substitute your Service user names for "hdfs" or "oozie" in each of the `su` commands.

Upgrading Oozie is a complex process. Although the instructions are straightforward, set aside a dedicated block of time to upgrade oozie clients and servers.

Perform the following preparation steps on each oozie server host:

1. You must restore oozie-site.xml from your backup to the conf directory on each oozie server and client.
2. Copy the JDBC jar to libext-customer:
   a. Create the `/usr/hdp/current/oozie-server/libext-customer` directory.

      ```
      cd /usr/hdp/current/oozie-server mkdir libext-customer
      ```

   b. Grant read/write/execute access to all users for the libext-customer directory.

      ```
      chmod -R 777 /usr/hdp/current/oozie-server/libext-customer
      ```

3. Copy these files to the libext-customer directory

   ```
   cp /usr/hdp/current/hadoop/lib/hadooplzo*.jar /usr/hdp/current/
   oozie-server/libext-customer
   ```

   ```
   cp /usr/share/HDP-oozie/ext.zip /usr/hdp/current/oozie-server/
   libext-customer/
   ```

4. Extract share-lib.

   ```
   /usr/hdp/current/oozie-server/bin/oozie-setup.sh sharelib create
   -fs hdfs://<namenode>:8020
   ```

   To verify that the sharelibs extracted correctly, run the following command:

   ```
   oozie admin -oozie http://<oozie server host address>:11000/
   oozie -shareliblist
   ```

   There should be:

   - Available ShareLib
   - oozie
   - hive

- distcp
- hcatalog
- sqoop
- mapreduce-streaming
- pig

Change the ownership and permissions of the oozie directory:

```
su -l hdfs -c "hdfs dfs -chown oozie:hadoop /user/oozie"
```

```
su -l hdfs -c "hdfs dfs -chmod -R 755 /user/oozie"
```

5. If a previous version of Oozie was created using auto schema creation, you must run an SQL query.

   Use the `oozie-site.xml` properties:

   - `oozie.service.JPAService.jdbc.username`
   - `oozie.service.JPAService.jdbc.username`
   - `oozie.service.JPAService.jdbc.url`

   to obtain the password, username and db to run the query.

   Run the SQL query:

   ```
   insert into oozie_sys (name, data) values ('db.version', '2.5');
   ```

6. As the Oozie user (not root), run the upgrade.

   ```
   su - oozie -c "/usr/hdp/current/oozie-server/bin/ooziedb.sh
   upgrade -run"
   ```

7. As root, prepare the Oozie WAR file.

   ```
   chown oozie:oozie /usr/hdp/current/oozie-server/oozie-server/conf/server.xml
   su - oozie -c "/usr/hdp/current/oozie-server/bin/oozie-setup.sh prepare-war
    -d /usr/hdp/current/oozie-server/libext-customer"
   ```

   Look for console output to indicate success. For example, if you are using MySQL you should see something similar to:

   ```
   INFO: Adding extension: libext-customer/mysql-connector-java.jar
   New Oozie WAR file with added 'JARs' at /var/lib/oozie/oozie-server/webapps/
   oozie.war
   ```

8. Replace the content of /user/oozie/share in HDFS. On the Oozie server host:

   ```
   su - oozie -c "/usr/hdp/current/oozie-server/bin/oozie-setup.sh
   prepare-war -d /usr/hdp/current/oozie-server/libext-customer"
   ```

9. Add the following property to oozie-log4j.properties:

   ```
   log4j.appender.oozie.layout.ConversionPattern=%d{ISO8601} %5p
   %c{1}:%L - SERVER[${oozie.instance.id}] %m%n
   ```

   where ${oozie.instance.id} is determined by oozie, automatically.

10. If you have custom Oozie actions, you must define them in oozie-site.xml. Edit the `/etc/oozie/conf/oozie-site.xml` file and add the following property:

    ```
    <property>
          <name>oozie.service.SchemaService.wf.ext.schemas</name>
          <value>[Comma separated list of custom actions]</value>
        </property>
    ```

    For example, if you have added Spark Action, enter the following:

    ```
    <property>
          <name>oozie.service.SchemaService.wf.ext.schemas</name>
          <value>spark-action-0.1.xsd</value>
        </property>
    ```

11. Upgrade the Oozie database:

    ```
    su - oozie -c "bin/ooziedb.sh upgrade -run"
    ```

12. Start Oozie as the Oozie user:

    ```
    su - oozie -c "/usr/hdp/current/oozie-server/bin/oozied.sh
    start"
    ```

13. Check processes.

    ```
    ps -ef | grep -i oozie
    ```

3.17. Configure and Start Apache WebHCat

Before you can upgrade Apache WebHCat, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "hdfs" to represent the HDFS Service user and webhcat to represent the WebHCat Service user. If you are using another name for these Service users, you will need to substitute your Service user name for "hdfs" or "webhcat" in each of the `su` commands.

1. You must replace your configuration after upgrading. Copy `/etc/hive-webhcat/conf` from the template to the conf directory in webhcat hosts.

2. Modify the WebHCat configuration files.

a. Upload Pig, Hive and Sqoop tarballs to HDFS as the $HDFS_USER (in this example, hdfs):

```
su - hdfs -c “hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/pig/"

su - hdfs -c “hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/hive/"

su - hdfs -c “hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/sqoop/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/pig/pig.tar.gz /hdp/
apps/2.4.3.0-$BUILD/pig/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/hive/hive.tar.gz /
hdp/apps/2.4.3.0-$BUILD/hive/"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.4.3.0-$BUILD/sqoop/sqoop.tar.gz /
hdp/apps/2.4.3.0-$BUILD/sqoop/"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/pig"

su - hdfs - "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/pig/pig.tar.
gz"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/hive"

su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/hive/hive.
tar.gz"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.4.3.0-$BUILD/sqoop/
sqoop.tar.gz"

su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp"
```

b. Update the following properties in the webhcat-site.xml configuration file, as their values have changed:

```
<property>
 <name>templeton.pig.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/pig/pig.tar.gz</value>
</property>

<property>
 <name>templeton.hive.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/hive/hive.tar.gz</value>
</property>

<property>
 <name>templeton.streaming.jar</name>
 <value>hdfs:///hdp/apps/${hdp.version}/mapreduce/
   hadoop-streaming.jar</value>
 <description>The hdfs path to the Hadoop streaming jar file.</
description>
</property>

<property>
 <name>templeton.sqoop.archive</name>
 <value>hdfs:///hdp/apps/${hdp.version}/sqoop/sqoop.tar.gz</value>
```

```
 <description>The path to the Sqoop archive.</description>
</property>

<property>
 <name>templeton.sqoop.path</name>
 <value>sqoop.tar.gz/sqoop/bin/sqoop</value>
 <description>The path to the Sqoop executable.</description>
</property>

<property>
 <name>templeton.sqoop.home</name>
 <value>sqoop.tar.gz/sqoop</value>
 <description>The path to the Sqoop home in the exploded archive.
   </description>
</property>
```

Note

You do not need to modify ${hdp.version}.

c. Remove the following obsolete properties from webhcat-site.xml:

```
<property>
 <name>templeton.controller.map.mem</name>
 <value>1600</value>
 <description>Total virtual memory available to map tasks.</description>
</property>

<property>
 <name>hive.metastore.warehouse.dir</name>
 <value>/path/to/warehouse/dir</value>
</property>
```

d. Add new proxy users, if needed. In core-site.xml, make sure the following properties are also set to allow WebHcat to impersonate your additional HDP-2.4.3 groups and hosts:

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

<property>
 <name>hadoop.proxyuser.hcat.hosts</name>
 <value>*</value>
</property>
```

Where:

`hadoop.proxyuser.hcat.group`

Is a comma-separated list of the Unix groups whose users may be impersonated by 'hcat'.

`hadoop.proxyuser.hcat.hosts`

A comma-separated list of the hosts which are allowed to submit requests by 'hcat'.

3. Start WebHCat:

```
su - hcat -c "/usr/hdp/current/hive-webhcat/sbin/
webhcat_server.sh start"
```

4. Smoke test WebHCat.

   a. At the WebHCat host machine, run the following command:

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

   b. If you are using a secure cluster, run the following command:

```
curl --negotiate -u: http://cluster.$PRINCIPAL.$REALM:50111/
templeton/v1/status {"status":"ok","version":"v1"}
[machine@acme]$
```

3.18. Configure Apache Pig

Before you can upgrade Apache Pig, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

1. Replace your configuration after upgrading. Copy `/etc/pig/conf` from the template to the conf directory in pig hosts.

2. To validate the Pig upgrade, complete the following steps:

   a. On the host machine where Pig is installed, run the following commands:

```
su - $HDFS_USER -c "/usr/hdp/current/hadoop-client/bin/hadoop fs -
copyFromLocal /etc/passwd
      passwd"
```

   b. Create a Pig script file named `/tmp/id.pig` that contains the following Pig Latin commands:

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

   c. Run the Pig script:

```
su - $HDFS_USER
pig -l /tmp/pig.log /tmp/id.pig
```

3.19. Configure and Start Apache Sqoop

Before you can upgrade Apache Sqoop, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The su commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the su commands.

1. Replace your configuration after upgrading. Copy /etc/sqoop/conf from the template to the conf directory in sqoop hosts.
2. As the HDFS Service user, upload the Sqoop tarball to HDFS.

```
su - hdfs -c “hdfs dfs -mkdir -p /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chmod -R 555 /hdp/apps/2.4.3.0-$BUILD/sqoop"

su - hdfs -c "hdfs dfs -chown -R hdfs:hadoop /hdp/apps/2.4.3.0-$BUILD/
sqoop"

su - hdfs -c "hdfs dfs -put /usr/hdp/2.3-$BUILD/sqoop/sqoop.tar.gz /hdp/
apps/2.4.3.0-$BUILD/sqoop/sqoop.tar.gz"

su - hdfs -c "hdfs dfs -chmod 444 /hdp/apps/2.4.3.0-$BUILD/sqoop/sqoop.tar.
gz"
```

3. If you are using the MySQL database as a source or target, then the MySQL connector jar must be updated to 5.1.29 or later.

   Refer to the MySQL web site for information on updating the MySQL connector jar.
4. Because Sqoop is a client tool with no server component, you will need to run your own jobs to validate the upgrade.

3.20. Configure, Start, and Validate Apache Flume

Before you can upgrade Apache Flume, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

1. If you have not already done so, upgrade Flume. On the Flume host machine, run the following command:
   - For **RHEL/CentOS/Oracle Linux**:

     ```
     yum upgrade flume
     ```
   - For **SLES**:

     ```
     zypper update flume

     zypper remove flume

     zypper se -s flume
     ```

You should see Flume in the output.

Install Flume:

```
zypper install flume
```

- For **Ubuntu/Debian**:

  HDP support for Debian 6 is deprecated with HDP 2.4.2. Future versions of HDP will no longer be supported on Debian 6.

  ```
  apt-get install flume
  ```

2. To confirm that Flume is working correctly, create an example configuration file. The following snippet is a sample configuration that can be set using the properties file. For more detailed information, see the "Flume User Guide."

```
agent.sources = pstream
agent.channels = memoryChannel
agent.channels.memoryChannel.type = memory

agent.sources.pstream.channels = memoryChannel
agent.sources.pstream.type = exec
agent.sources.pstream.command = tail -f /etc/passwd

agent.sinks = hdfsSink
agent.sinks.hdfsSink.type = hdfs
agent.sinks.hdfsSink.channel = memoryChannel
agent.sinks.hdfsSink.hdfs.path = hdfs://tmp/flumetest
agent.sinks.hdfsSink.hdfs.fileType = SequenceFile
agent.sinks.hdfsSink.hdfs.writeFormat = Text
```

   The source here is defined as an exec source. The agent runs a given command on startup, which streams data to stdout, where the source gets it. The channel is defined as an in-memory channel and the sink is an HDFS sink.

3. Given this configuration, you can start Flume by navigating to FLUME_HOME and executing the following command:

```
$ bin/flume-ng agent --conf ./conf --conf-file example.conf --name a1 -
Dflume.root.logger=INFO,console
```

Note

The directory specified for `--conf agrument` would include a shell script flume-env.sh and potentially a log4j properties file. In this example, we pass a Java option to force Flume to log to the console and we go without a custom environment script.

4. After validating data in `hdfs://tmp/flumetest`, stop Flume and restore any backup files. Copy `/etc/flume/conf` to the conf directory in Flume hosts.

3.21. Configure and Validate Apache Mahout

Before you can upgrade Apache Mahout, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

Replace your configuration after upgrading. Copy `/etc/mahout/conf` from the backup if it existed to the conf directory in mahout hosts.

To validate mahout:

1. Create a test user named "testuser" in the Linux cluster and in HDFS, and log in as that user.

2. Export the required environment variables for Mahout:

```
export JAVA_HOME="your_jdk_home_install_location_here
export HADOOP_HOME=/usr/hdp/current/hadoop-client
export MAHOUT_HOME=/usr/hdp.current/mahout-client
export PATH="$PATH":$HADOOP_HOME/bin:$MAHOUT_HOME/bin
export CLASSPATH="$CLASSPATH":$MAHOUT_HOME
```

3. Upload a few megabytes of natural-language plain text to the Linux server as `/tmp/sample-test.txt`.

4. Transfer the sample-test.txt file to a subdirectory of the testusers's HDFS home directory.

```
hdfs dfs -mkdir /user/testuser/testdata
hdfs dfs -put /tmp/sample-test.txt /user/testuser/testdata
```

5. Enter the mahout command to convert the plain text file sample-test.txt into a sequence file stored in the output directory mahouttest:

```
mahout seqdirectory --input /user/testuser/testdata --output /user/testuser/
mahouttest -ow --charset utf-8
```

3.22. Configure and Start Hue

Before you can upgrade Hue, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

For HDP-2.4.3, use the Hue version shipped with HDP-2.4.3. If you have a previous version of Hue, use the following steps to upgrade Hue.

1. Migrate hue.ini setting from your old hue.ini configuration file to new hue.ini configuration file.
2. If you are using the embedded SQLite database, remember to restore your database after upgrade.

   To restore the database from a backup, make sure the destination database is empty before copying (if necessary, rename or remove the current destination database), then copy your backup to the destination database. For example:

   ```
   su - hdfs
   cd /var/lib/hue
   mv desktop.db desktop.db.old
   sqlite3 desktop.db < ~/hue_backup/desktop.bak
   exit
   ```

3. Synchronize Database

   ```
   cd /usr/lib/hue
   source ./build/env/bin/activate
   hue syncdb
   deactivate
   ```

4. Run the following script to pick up the new Hue version:

   ```
   bash /usr/lib/hue/tools/fill_versions.sh
   ```

5. Run the following script to pick up the new Hue version:

   ```
   bash /usr/lib/hue/tools/fill_versions.sh
   ```

6. Start Hue. As a root user, run the following command on the Hue Server:

   ```
   /etc/init.d/hue start
   ```

3.23. Configure and Start Apache Knox

Before you can upgrade Apache Knox, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

When working with the Knox Gateway in your Hadoop cluster, it is important you have the latest version of Knox installed so you can take advantage of new features and enhancements, in addition to ensuring your instance of Knox is in sync with other Hadoop

components (e.g. Ranger, Spark, Hive, Hue, etc.) for stability and performance. For example, if you need to upgrade your Hadoop cluster from 2.1 to 2.2.x, you should also make sure that your individual Hadoop components are also upgraded to the latest version.

HDP enables you to perform a rolling upgrade in 2.2.x. A rolling upgrade means that you can upgrade a component, or the entire Hadoop stack, without losing service, and your users can continue to use the cluster and run jobs with no application or server downtime. The main distinction between a rolling upgrade and a traditional upgrade implies the use of a Knox cluster for high availability capabilities. This means that you can require multiple instances of the gateway and a load balancer serving each gateway instance from a single URL.

The main distinction between a rolling upgrade and a traditional upgrade implies the use of a Knox cluster for high availability capabilities. This means that you can require multiple instances of the gateway and a load balancer serving each gateway instance from a single URL. Once the upgrade process is completed, you will be up and running with the latest version of Knox on each server you have designated as a Knox server.

Note

In this document, whenever you see a {} with a value inside, this denotes a value you must define.

3.23.1. Upgrade the Knox Gateway

If you are not currently using Ambari to manage your Hadoop cluster, you will need to upgrade Knox manually to the latest version. Because "rolling upgrades" are now supported in HDP-2.4.3, it is not important which version of Knox you are currently running, only that you have an instance of the Knox Gateway running.

Note

If you have not already installed Knox, refer to the "Install the Knox RPMs on the Knox Server" section of the Non-Ambari Cluster Installation Guide for instructions on how to install and configure the Knox Gateway.

Before upgrading the Knox Gateway, there are a several steps you must follow to ensure your configuration files, settings, and topology files can be copied to the new Knox Gateway instance when the upgrade is complete, which are described below.

1. Back up your existing `conf` directory.
2. Stop each Knox server.

```
su -l knox /usr/hdp/{the current Knox version}/knox/bin/gateway.sh stop
```

3. Select the HDP server version you are upgrading to after you have stopped each Knox server.

```
hdp-select set knox-server {the HDP server version}
```

4. Unzip your previously saved configuration directory (the `conf` directory you backed up in step 1) into the new */var/log/knox/gateway.conf* directory to import these files.

5. Create the Master Secret:

   ```
   su - knox -c “/usr/hdp/current/knox/bin/knoxcli.sh create-master”
   ```

6. Start the Gateway:

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

7. Restart the Knox server to complete the upgrade.

   ```
   su -l knox /usr/hdp/{the new HDP server version}/knox/bin/gateway.sh start
   ```

3.23.2. Verify the Knox Upgrade

To verify the upgrade was successful, follow the steps listed below.

1. Navigate to the */var/log/knox/gateway* directory and check the `gateway.log` file for errors and an acknowledgement that the server restart was successful.

2. Verify you have cluster access using the `LISTSTATUS` WebHDFS API call.

   ```
   curl -ivk -u {user}:{password} https://{knox host}:8443 /gateway/webhdfs/v1/tmp?op=LISTSTATUS
   ```

3. Verify the Knox version using the Knox Admin service and Version API.

   ```
   curl -ivk -u {adminuser}:{adminpassword} https://{knox host}:8443 /gateway/admin/v1/version
   ```

Note

The Admin API requires you to be a member of an Admin group, as specified in the `admin.xml` authorization provider.

When you have verified the Knox upgrade was successful, you can begin using Knox. If, however, the upgrade was unsuccessful, you will need to downgrade the Knox Gateway to the previous version. The steps to downgrade the Knox Gateway are described in the next section.

3.23.3. Downgrade the Knox Gateway to the Previous Version

If the Knox Gateway upgrade was unsuccessful, you will need to downgrade Knox to the previous version to ensure you have a working Knox Gateway for your cluster. To downgrade Knox, follow the steps listed below.

1. For each server running Knox, stop the server.

   ```
   su -l knox /usr/hdp/{current HDP server version}/knox/bin/gateway.sh stop
   ```

2. Select the HDP server version you want to use to downgrade your Knox Gateway.

   ```
   hdp-select set knox-server {previous HDP server version}
   ```

3. Restart the server.

```
su -l knox /usr/hdp/{previous HDP server version}/knox/bin/gateway.sh start
```

3.23.4. Verify the Knox Downgrade Was Successful

When the restart is complete, verify you are running an older version of Knox by following the steps listed below.

1. Navigate to the */var/log/knox/gateway* directory and check the `gateway.log`file for errors and an acknowledgement that the server restart was successful
2. Verify you have cluster access using the `LISTSTATUS`WebHDFS API call.
3. Check the Knox version using the Knox Admin service and Version API using the following command:

```
curl -ivk -u {adminuser}"{adminpassword} https://{knox host}:8443 /gateway/
admin/v1/version
```

3.24. Configure and Validate Apache Falcon

Before you can upgrade Apache Falcon, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

In HDP-2.4.3, if authorization is enabled (for example, in the startup.properties file with `*.falcon.security.authorization.enabled=true`") then Access Control List (ACL) is mandated for all entities.

Upgrade Falcon after you have upgraded HDFS, Hive, Oozie, and Pig. Stop Oozie jobs while running Falcon.

Important

We recommend upgrading Falcon from HDP 2.1 to HDP 2.2 and then upgrading from HDP 2.2 to HDP 2.3.

Upgrade Falcon from HDP 2.1 to HDP 2.2

1. Upgrade Falcon from HDP 2.1 to HDP 2.2.
   - **RHEL/CentOS/Oracle 6**
     a. Download the HDP 2.2.9 hdp.repo file:

     ```
     wget -nv http://public-repo-1.hortonworks.com/HDP/
     centos6/2.x/updates/2.2.9.0/hdp.repo -0 /etc/yum.repos.d/
     hdp.repo
     ```

     b. Install the HDP 2.2.9 version of Falcon:

```
yum install "falcon"
```

- **SLES 11 SP 1**

  a. Download the HDP 2.2.9 hdp.repo file:

  ```
  wget -nv http://public-repo-1.hortonworks.com/HDP/
  sles11sp1/2.x/updates/2.2.9.0/hdp.repo -O /etc/zypp/repos.d/
  hdp.repo
  ```

  b. Install the HDP 2.2.9 version of Falcon:

  ```
  zypper install "falcon"
  ```

- **SLES 11 SP3/SP4**

  a. Download the HDP 2.2.9 hdp.repo file:

  ```
  http://public-repo-1.hortonworks.com/HDP/suse11sp3/2.x/
  updates/2.2.9.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
  ```

  b. Install the HDP 2.2.9 version of Falcon:

  ```
  zypper install "falcon"
  ```

- **Ubuntu 12**

  a. Download the HDP 2.2.9 hdp.repo file:

  ```
  wget http://public-repo-1.hortonworks.com/HDP/ubuntu12/2.x/
  updates/2.2.9.0/hdp.list -O /etc/apt/sources.list.d/hdp.list
  ```

  b. Install the HDP 2.2.9 version of Falcon:

  ```
  apt-get install "falcon"
  ```

- **Debian 6 (Deprecated)**

  HDP support for Debian 6 is deprecated with HDP 2.4.2. Future versions of HDP will no longer be supported on Debian 6.

  a. Download the HDP 2.2.9 hdp.repo file:

  ```
  wget -nv http://public-repo-1.hortonworks.com/HDP/
  debian6/2.x/updates/2.2.9.0/hdp.list -O /etc/apt/
  sources.list.d/hdp.list
  ```

  b. Install the HDP 2.2.9 version of Falcon:

  ```
  apt-get install "falcon"
  ```

2. Verify start.properties. In Falcon 2.2, TLS is enabled by default. When TLS is enabled, Falcon starts on `https:/<falcon_host>:15443/`. You can disable TLS by adding the following line in startup.properties:

```
*.falcon.enableTLS=false
```

3. Verify the `client.properties` file. If TLS is disabled, make sure the property `falcon.url` is set as follows:

   ```
   falcon.url=http://localhost:15000/
   ```

4. Install the hotfix provided by Hortonworks customer support to migrate entities in bulk.

5. Restart Falcon.

   ```
   su - falcon -c "cd /usr/hdp/2.2.9.0-$BUILD/falcon/bin/falcon-start"
   ```

6. Run Falcon CLI to update Falcon entities from HDP 2.1 to HDP 2.2. This script also updates the ownership and permissions of staging and working directories of cluster entitles. Run this script as user falcon.

   ```
   falcon upgrade -owner hrt_qa -group users -configDir
   <falcon_config_store> -backupDir/tmp/
   ```

   Where:

   - `Owner` and `group` are used to set ACL for Falcon entities.
   - `configDir` refers to the location where Falcon stores its entities. This directory should be the same location as the value of property `*.config.store.uri` in the file `conf/startup.properties`.
   - `backupDir` is the location where Falcon 2.1 entitles are copied as backup. User can restore entities from this directory.

7. Restart Falcon.

   ```
   su - falcon -c "cd /usr/hdp/2.2.9.0-$BUILD/falcon/bin/falcon-start"
   ```

8. Replace your configuration after upgrading. Copy `/etc/falcon/conf` from the template to the conf directory in falcon hosts.

9. Check your Falcon entities. There should be no changes, but in some cases you may need to update your entities post-upgrade.

Upgrade Falcon from HDP 2.2 to HDP 2.3

1. **RHEL/CentOS/Oracle 6**

   a. On all Falcon hosts, clean the yum repository.

      ```
      yum clean all
      ```

   b. Remove the HDP 2.2 Falcon component. This command uninstalls the HDP 2.2 component. It leaves the user data, and metadata, but removes your configurations:

      ```
      yum erase "falcon*"
      ```

   c. Validate that all HDP 2.2 component binaries are uninstalled:

```
yum list installed | grep @HDP2.2
```

d. Remove the HDP 2.2 hdp.repo file:

```
rm /etc/yum.repos.d/hdp.repo
```

e. Install the HDP 2.4.3.0 repo:

- Download the hdp.repo file:

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

SLES 11 SP 1

a. On all Falcon hosts, clean the yum repository.

```
zypper clean -all
```

b. Remove the Falcon HDP 2.2 component. This command uninstalls the HDP 2.2 components. It leaves the user data, and metadata, but removes your configurations:

```
zypper rm "falcon*"
```

c. Validate that all HDP 2.2 component binaries are uninstalled:

```
yum list installed | grep @HDP2.2
```

d. Remove the HDP 2.2 hdp.repo file:

```
rm /etc/zypp/repos.d/hdp.repo
```

e. Download the HDP 2.4.3.0 hdp.repo file:

```
wget -nv http://public-repo-1.hortonworks.com/HDP/sles11sp1/2.x/updates/
2.4.3.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
```

f. Install the HDP 2.4.3.0 version of the Falcon component.

```
zypper install "falcon"
```

SLES 11 SP3/SP4

a. On all Falcon hosts, clean the zypper repository.

```
zypper clean -all
```

b. Remove HDP 2.2 Falcon component.

```
zypper rm "falcon*"
```

c. Validate that all HDP 2.2 component binaries are uninstalled:

```
zypper search --installed-only --repo HDP-2.2.6.0
```

d. Remove the HDP 2.2 hdp.repo file:

```
rm /etc/zypp/repos.d/hdp.repo
```

e. Download the HDP 2.4.3.0 hdp.repo file:

```
http://public-repo-1.hortonworks.com/HDP/suse11sp3/2.x/
updates/2.4.3.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
```

f. Install the HDP 2.4.3.0 version of the Falcon component. F

```
zypper install "falcon"
```

Ubuntu 12

a. On all Falcon hosts, clean the apt-get repository.

```
apt-get clean -&-all
```

b. Remove the HDP 2.2 Falcon component. This command uninstalls the HDP 2.2 component. It leaves the user data, and metadata, but removes your configurations:

```
apt-get remove "falcon*"
```

c. Validate that all HDP 2.2 component binaries are uninstalled:

```
yum list installed | grep @HDP2.2
```

d. Remove the HDP 2.2 hdp.repo file:

```
rm /etc/apt/sources.list.d/hdp.list
```

e. Download the HDP 2.4.3.0 hdp.repo file:

```
wget -nv http://public-repo-1.hortonworks.com/HDP/
ubuntu12/2.x/updates/2.4.3.0/hdp.list - O /etc/apt/
sources.list.d/hdp.list
```

f. Run an update:

```
apt-get update
```

g. Install the HDP 2.4.3.0 versions of the Falcon component.

```
apt-get install "falcon"
```

Debian 6 (Deprecated)

HDP support for Debian 6 is deprecated with HDP 2.4.2. Future versions of HDP will no longer be supported on Debian 6.

a. On all Falcon hosts, clean the apt-get repository.

```
apt-get clean
```

b. Remove the HDP 2.2 Falcon component. This command uninstalls the HDP 2.2 component. It leaves the user data, and metadata, but removes your configurations:

```
apt-get remove "falcon*"
```

c. Validate that all HDP 2.2 component binaries are uninstalled:

```
dpkg -l | grep "^ii" | grep hadoop
```

d. Remove the HDP 2.2 hdp.repo file:

```
rm /etc/apt/sources.list.d/hdp.list
```

e. Download the HDP 2.4.3.0 hdp.repo file:

```
wget -nv http://public-repo-1.hortonworks.com/HDP/debian6/2.x/
updates/2.4.3.0/hdp.list - O /etc/apt/sources.list.d/hdp.list
```

f. Run an update:

```
apt-get update
```

g. Install the HDP 2.4.3.0 versions of the Falcon component.

```
apt-get install "falcon"
```

2. Verify start.properties. In Falcon 2.4.3, TLS is enabled by default. When TLS is enabled, Falcon starts on `https:/<falcon_host>:15443/`. You can disable TLS by adding the following line in startup.properties:

```
*.falcon.enableTLS=false
```

3. Verify the `client.properties` file. If TLS is disabled, make sure the property `falcon.url` is set as follows:

```
falcon.url=http://localhost:15000/
```

4. Install the hotfix provided at `https://hortonworks.jire.com/browse/BUG-32579`.

5. Restart Falcon.

```
su - falcon -c "cd /usr/hdp/2.4.3.0-$BUILD/falcon/bin/falcon-start"
```

6. Run Falcon CLI to update Falcon entities from HDP 2.1 to HDP 2.2. This script also updates the ownership and permissions of staging and working directories of cluster entitles. Run this script as user falcon.

```
falcon upgrade -owner hrt_qa -group users -configDir
<falcon_config_store> -backupDir/tmp/
```

Where:

- `Owner` and `group` are used to set ACL for Falcon entities.
- `configDir` refers to the location where Falcon stores its entities. This directory should be the ssame location as the value of property `*.config.store.uri` in the file `conf/startup.properties`.

- `backupDir` is the location where Falcon 2.1 entitles are copied as backup. User can restore entities from this directory.

7. Restart Falcon.

```
su - falcon -c "cd /usr/hdp/2.4.3.0-$BUILD/falcon/bin/falcon-start"
```

8. Replace your configuration after upgrading. Copy `/etc/falcon/conf` from the template to the conf directory in falcon hosts.

9. Check your Falcon entities. There should be no changes, but in some cases you may need to update your entities post-upgrade.

3.25. Configure and Start Apache Storm

Before you can upgrade Apache Storm, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you have already upgraded your components for HDP 2.4.3. If you have not already completed these steps, return to Getting Ready to Upgrade and Upgrade 2.1 Components for instructions on how to upgrade your HDP components to 2.4.3.

Note

The `su` commands in this section use "zookeeper" to represent the ZooKeeper Service user. If you are using another name for your ZooKeeper Service user, you will need to substitute your ZooKeeper Service user name for "zookeeper" in each of the `su` commands.

Storm is fairly independent of changes to the HDP cluster, but you must upgrade Storm for rolling upgrade support in HDP-2.4.3 and be on the latest version of Storm.

1. Deactivate all running topologies.

2. Delete all states under zookeeper:

```
/usr/hdp/current/zookeeper-client/bin/zkCli.sh (optionally in
secure environment specify -server zk.server:port)

rmr /storm
```

3. Delete all states under the storm-local directory:

```
rm -rf <value of storm.local.dir>
```

4. Stop Storm services on the storm node.

5. Stop ZooKeeper services on the storm node.

```
su - zookeeper -c "export ZOOCFGDIR=/etc/zookeeper/conf ; export
ZOOCFG=zoo.cfg ;source /etc/zookeeper/conf/zookeeper-env.sh ; /
usr/lib/zookeeper/bin/zkServer.sh stop"
```

6. Remove Storm and ZooKeeper from the storm node and install the HDP-2.4.3 version:

- For **RHEL/CentOS/Oracle Linux**:

  ```
  yum erase storm

  yum erase zookeeper

  yum install storm

  yum install zookeeper
  ```

- For **SLES**:

  ```
  zypper rm storm

  zypper rm zookeeper

  zypper install storm

  zypper install zookeeper
  ```

- For **Ubuntu/Debian**:

  HDP support for Debian 6 is deprecated with HDP 2.4.2. Future versions of HDP will no longer be supported on Debian 6.

  ```
  apt-get remove storm --purge

  apt-get remove zookeeper --purge

  apt-get install storm

  apt-get install zookeeper
  ```

7. Replace your configuration after upgrading. Copy /etc/storm/conf from the template to the conf directory .

8. Replace your ZooKeeper configuration after upgrading. Replace the ZooKeeper template configuration in `/etc/zookeeper/conf`.

9. Start ZooKeeper. On the storm node, run the following command:

   ```
   su - zookeeper -c "source /etc/zookeeper/conf/zookeeper-env.sh; export
    ZOOCFGDIR=/etc/zookeeper/conf; /usr/hdp/current/zookeeper-server/bin/
   zkServer.sh start >> $ZOO_LOG_DIR/zoo.out\"
   ```

   where

   - $ZOO_LOG_DIR is the directory where ZooKeeper server logs are stored. For example, /var/log/zookeeper.

10. Start nimbus, then supervisor/ui/drpc/logviewer:

    `/usr/hdp/current/storm-nimbus/bin/storm nimbus`.

11. Start Storm, using a process controller, such as supervisor:

```
su - storm /usr/hdp/current/storm-supervisor/bin/storm
supervisor
```

You can use the same command syntax to start Storm using nimbus/ui and logviewer.

```
su - storm /usr/hdp/current/storm-supervisor/bin/storm nimbus

su - storm /usr/hdp/current/storm-supervisor/bin/storm ui

su - storm /usr/hdp/current/storm-supervisor/bin/storm logviewer

su - storm /usr/hdp/current/storm-supervisor/bin/storm drpc
```

3.26. Configure and Start Apache Ranger

Before you can upgrade the Apache Ranger service, you must have first upgraded your HDP components to the latest version (in this case, 2.4.3). This section assumes that you already have already performed the following tasks, however, if you have not already performed these steps, refer to the "*Upgrade HDP 2.1 Components*" section in this guide for instructions on how to upgrade your HDP components to 2.4.3.

Note

XA Secure was an add-on component in HDP-2.1. Ranger is the new name for XA Secure. In HDP-2.2 and subsequent releases, Ranger is installed with HDP.

3.26.1. Preparing Your Cluster to Upgrade Ranger

If you are not currently using Ambari to manage your Hadoop cluster, you will need to upgrade Ranger manually to the latest version. This section describes the steps you need to follow to prepare your cluster for the Ranger upgrade.

1. Back up the following Ranger configuration directories:
   - Ranger Policy Administration Service

     ```
     /usr/lib/xapolicymgr
     ```

   - Ranger UserSync

     ```
     /usr/lib/uxugsync or /etc/uxugsync (Depending on your installation)
     ```

   - Ranger Plugins:
     - Hadoop

       ```
       /etc/hadoop/conf
       ```

     - Hive

       ```
       /etc/hive/conf
       ```

     - HBase

```
/etc/hbase/conf
```

- Knox

```
/etc/knox/conf
```

- Storm

```
/etc/storm/conf
```

2. Backup the Ranger Policy and Audit databases. Make sure to take note of the following details in the `install.properties` file:
   - db_host
   - db_name
   - db_user
   - db_password
   - policy manager configuration
   - LDAP directory configuration
   - LDAP settings
   - LDAP AD domain
   - LDAP URL

3.26.2. Stop the Ranger Services

Now that you have prepared your cluster for the Ranger upgrade, you will need to stop the Ranger Admin and Ranger UserSync services. To stop the Ranger services, perform the steps described below.

1. Stop the Ranger Policy Admin service. When the service is stopped, you will receive an acknowledgement from the server that the service has been stopped.

```
service xapolicymgr stop
```

2. Stop the Ranger UserSync service. When the service is stopped, you will receive an acknowledgement from the server that the service has been stopped.

```
service uxugsync stop
```

3. Stop each individual Ranger plugin (HDFS, HBase, Knox, Storm). You will receive an acknowledgement from the server that the plugin has been stopped.

```
service <plugin name> stop
```

3.26.3. Install the Ranger Components

Next, you will need to re-install each Ranger component again to ensure you have the latest version. Because you have already upgraded your HDP stack, you only need to

follow the instructions in the Non-Ambari Cluster Installation Guide to install each Ranger component. The following components must be installed:

- Ranger Policy Admin service
- Ranger UserSync service
- Ranger Plugins:
  - HDFS
  - HBase
  - Hive
  - Knox
  - Storm

With this release, Ranger has also added support for the following components:

- Solr
- Kafka

  For more information about Kafka on HDP, see the Kafka Guide
- YARN

3.26.4. Restart the Ranger Services

Once you have re-installed each Ranger component, you will then need to restart these components to ensure the new configurations are loaded in your cluster. The Non-Ambari Cluster Installation Guide describes how you can start the following Ranger services:

- Ranger Policy Admin service

```
service ranger-admin start
```

- Ranger UserSync service

```
service ranger-usersync start
```

3.26.5. Remove Existing Startup Files and Symbolic Links

In order to ensure that your Ranger components are upgraded correctly, and there are no conflicts between versions, you should remove any existing startup files and symbolic links from the previous Ranger 2.2 version. The steps you need to follow to remove these files and links are described below.

1. Remove the Policy Manager startup files.

```
rm -f /etc/init.d/xapolicymgr
```

2. Remove the Policy Manager symbolic links.

```
rm -rf /etc/rc*.d/*xapolicymgr
```

3. Remove the UserSync startup files.

```
rm -f /etc/rc*.d/*uxugsync
```

4. Remove the UserSync symbolic links.

```
rm -rf /etc/rc*.d/uxugsync
```

5. Remove the Policy Manager library files.

```
rm -f /usr/lib/xapolicymgr
```

6. Remove the UserSync library files.

```
rm -f /usr/lib/uxugsync
```

3.26.6. Enable Ranger Plugins

The final step in the Ranger upgrade process requires you to re-enable the Ranger plugins. Although you are only required to enable HDFS in your cluster, you should re-enable all of the Ranger plugins because class names have changed for the 2.4.3 release.

Note

When you enable each Ranger plugin, make sure you remove all 2.1 class name values.

To re-enable the Ranger plugins, use the links listed below to access instructions in the *Non-Ambari Cluster Installation* guide that describe editing the `install.properties` file and enabling the Ranger plugins:

Important

Before enabling the HDFS plugin, remove `set-hdfs-plugin-env.sh` from `/etc/hadoop/conf`. You will need to re-enable this plugin after the upgrade is complete.

- HDFS Plugin
- YARN Plugin
- Kafka Plugin
- HBase Plugin
- Hive Plugin
- Knox Plugin
- Storm Plugin

3.27. Finalize the Upgrade

Note

The `su` commands in this section use keywords to represent the Service user. For example, "hdfs" is used to represent the HDFS Service user. If you are using another name for your Service users, you will need to substitute your Service user name in each of the `su` commands.

You can start HDFS without finalizing the upgrade. When you are ready to discard your backup, you can finalize the upgrade.

1. Verify your filesystem health before finalizing the upgrade. (After you finalize an upgrade, your backup will be discarded!)
2. As the $HDFS_USER, enter:

```
su - hdfs -c "dfsadmin -finalizeUpgrade"
```

3.28. Install New HDP Services

Install new HDP services introduced in HDP releases subsequent to HDP 2.1. See the Non-Ambari Cluster Installation Guide for details.