Ambari 1.7.0 Upgrade Guide

Hortonworks Data Platform
Ambari 1.7.0
2014-12-02

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.**

For more information on Hortonworks technology, Please visit the Hortonworks Data Platform page. 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

Upgrading Ambari

This section describes how to upgrade Ambari Server to 1.7.0, including how to upgrade the HDP stack to 2.2 and how to upgrade an older Ambari Server version to 1.2.5.

Upgrading Ambari Server to 1.7.0

This procedure upgrades Ambari Server from version 1.2.5 and above to 1.7.0. If your current Ambari Server version is 1.2.4 or below, you must upgrade the Ambari Server version to 1.2.5 before upgrading to version 1.7.0. Upgrading the Ambari Server version does not change the underlying Hadoop Stack.

Before Upgrading Ambari to 1.7.0, make sure that you perform the following actions:

- You ***must*** have root, administrative, or root-equivalent authorization on the Ambari server host and all servers in the cluster.
- You ***must*** know the location of the Nagios server before you begin the upgrade process.
  For example, to find the Nagios server using Ambari 1.6.0 or higher:
  - Browse `Ambari Web` > `Services` > `Summary`
  - Select the `Nagios Server` link
  - Scroll down to view summary information about the Nagios server host
- If you are using Ambari with Oracle, you ***must*** create an Ambari user in the Oracle database and grant that user all required permissions. Specifically, you must alter the Ambari database user and grant the SEQUENCE permission.
  For more information about creating users and granting required user permissions, see Using Ambari with Oracle.
- If you plan to upgrade your Stack, back up the configuration properties for your current Hadoop services.
  For more information about upgrading the Stack and locating the configuration files for your current services, see one of the following topics:
  - Upgrade from HDP 2.1 to HDP 2.2, Getting Ready to Upgrade
  - Upgrade from HDP 2.0 to HDP 2.2, Getting Ready to Upgrade
  - Upgrade from HDP 1.3 to HDP 2.2, Getting Ready to Upgrade

1. Stop the Nagios and Ganglia services.
   In **Ambari Web**:
   1. Browse to **Services** and select the Nagios service.
   2. Use **Service Actions** to stop the Nagios service.
   3. Wait for the Nagios service to stop.
   4. Browse to **Services** and select the Ganglia service.
   5. Use **Service Actions** to stop the Ganglia service.
   6. Wait for the Ganglia service to stop.
2. Stop the Ambari Server. On the Ambari Server host,

```
ambari-server stop
```

3 Stop all Ambari Agents. On each Ambari Agent host,

```
ambari-agent stop
```

4 Fetch the new Ambari repo using `wget` and replace the old repository file with the new repository file on all hosts in your cluster.

Check your current directory before you download the new repository file to make sure that there are no previous versions of the ambari.repo file. If you do not, and a previous version exists, the new download will be saved with a numeric extension, such as ambari.repo.1. Make sure that the version you copy is the new version.

Select the repository appropriate for your environment from the following list:

- **For RHEL/CentOS 6/Oracle Linux 6:**

```
wget -nv http://public-repo-
1.hortonworks.com/ambari/centos6/1.x/updates/1.7.0/ambari.repo -O
/etc/yum.repos.d/ambari.repo
```

- **For SLES 11:**

```
wget -nv http://public-repo-
1.hortonworks.com/ambari/suse11/1.x/updates/1.7.0/ambari.repo -O
/etc/zypp/repos.d/ambari.repo
```

- For Ubuntu 12:

```
wget -nv http://public-repo-
1.hortonworks.com/ambari/ubuntu12/1.x/updates/1.7.0/ambari.list -O
/etc/apt/sources/list.d/ambari.list
```

- **For RHEL/CentOS 5/Oracle Linux 5:** (DEPRECATED)

```
wget -nv http://public-repo-
1.hortonworks.com/ambari/centos5/1.x/updates/1.7.0/ambari.repo -O
/etc/yum.repos.d/ambari.repo
```


If your cluster does not have access to the Internet, set up a local repository with this data before you continue. See Configure the Local Repositories for more information.

5 Upgrade Ambari Server.

Ambari Server does not automatically turn off `iptables`. Check that your installation setup does not depend on **iptables** being disabled. After upgrading the server, you must either disable **iptables** manually or make sure that you have appropriate ports available on all cluster hosts. For more information about ports, see Configuring Network Port Numbers.

At the Ambari Server host:

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

```
yum clean all
yum upgrade ambari-server ambari-log4j
```

- **For SLES:**

```
zypper clean
zypper up ambari-server ambari-log4j
```

- **For Ubuntu:**

```
apt-get clean all
apt-get install ambari-server ambari-log4j
```

6 Check for upgrade success by noting progress during the Ambari server installation process you started in Step 5.

- As the process runs, the console displays output similar, although not identical, to the following:

```
Setting up Upgrade Process
Resolving Dependencies
--> Running transaction check
---> Package ambari-server.x86_64 0:1.2.2.3-1 will be updated
---> Package ambari-server.x86_64 0:1.2.2.4-1 will be updated ...
---> Package ambari-server.x86_64 0:1.2.2.5-1 will be an update ...
```

- If the upgrade fails, the console displays output similar to the following:

```
Setting up Upgrade Process
No Packages marked for Update
```

- A successful upgrade displays the following output:

```
Updated: ambari-log4j.noarch 0:1.7.0.111-1 ambari-server.noarch 0:1.7.0-111
Complete!
```

7 Upgrade the Ambari Server schema.

On the Ambari Server host:

```
ambari-server upgrade
```

8 Upgrade the Ambari Agent on all hosts.

At each Ambari Agent host:

For RHEL/CentOS/Oracle Linux:

```
yum upgrade ambari-agent ambari-log4j
```

For SLES:

```
zypper up ambari-agent ambari-log4j
```

Ignore the warning that begins with "There are some running programs that use files deleted by recent upgrade".

For Ubuntu:

```
apt-get update
apt-get install ambari-agent ambari-log4j
```

9 On each Agent host, check for a file named `/etc/ambari-agent/conf.save`. If that folder exists, rename it back to `/etc/ambari-agent/conf`.

```
mv /etc/ambari-agent/conf.save /etc/ambari-agent/conf
```

10 After the upgrade process completes, check each host to make sure the new 1.7.0 files have been installed:

```
rpm -qa | grep ambari
```

11 Start the Ambari Server. At the Ambari Server host,

```
ambari-server start
```

12 Start the Ambari Agents on all hosts. At each Ambari Agent host,

```
ambari-agent start
```

13 Open Ambari Web.

Point your browser to http://<your.ambari.server>:8080

where <your.ambari.server> is the name of your ambari server host. For example, c6401.ambari.apache.org.

Refresh your browser so that it loads the new version of the Ambari Web code. If you have problems, clear your browser cache manually, then restart Ambari Server.

14 Log in, using the Ambari administrator credentials that you have set up.

For example, the default name/password is `admin/admin`.

15 Start the Nagios and Ganglia services.

In Ambari Web,

1. Browse to **Services** and select each service.
2. Use **Service Actions** to start the service.

16 If you have customized logging properties, you will see a Restart indicator next to each service name after upgrading to Ambari 1.7.0.

Restarting a service pushes the configuration properties displayed in **Custom log4j.properties** to each host running components for that service.

To preserve any custom logging properties after upgrading, for each service:

1. Replace default logging properties with your custom logging properties, using **Service Configs > Custom log4j.properties**.
2. Restart all components in any services for which you have customized logging properties.

17. Review the HDP-UTILS repository Base URL setting.

As of Ambari 1.7.0, the HDP-UTILS repository Base URL is no longer set in the ambari.repo file. Browse to `Ambari Web` > `Admin` > `Repositories`, and confirm the value of the HDP-UTILS repository Base URL is correct for your environment. If you are using a local repository for HDP-UTILS, be sure to confirm the Base URL is correct for your locally hosted HDP-UTILS repository.

18. Review your Ambari LDAP authentication settings.

If you have configured Ambari for LDAP authentication, you must re-run "ambari-server setup-ldap". For further information, see Set Up LDAP or Active Directory Authentication.

Upgrading the HDP Stack from 2.1 to 2.2

The HDP Stack is the coordinated set of Hadoop components that you have installed on hosts in your cluster. Your set of Hadoop components and hosts is unique to your cluster. Before upgrading the Stack on your cluster, review all Hadoop services and hosts in your cluster. For example, use the Hosts and Services views in Ambari Web, which summarize and list the components installed on each Ambari host, to determine the components installed on each host. For more information about using Ambari to view components in your cluster, see Working with Hosts, and Viewing Components on a Host.

Upgrading the HDP Stack is a three-step procedure:

1. Prepare the 2.1 Stack for Upgrade
2. Upgrade the 2.1 Stack to 2.2
3. Complete the Upgrade of the 2.1 Stack to 2.2

If you plan to upgrade your existing JDK, do so after upgrading Ambari, before upgrading the Stack. The upgrade steps require that you remove HDP v2.1 components and install HDP v2.2 components.

As noted in that section, you should remove and install on each host, only the components on each host that you want to run on the HDP 2.2 stack. For example, if you want to run Storm or Falcon components on the HDP 2.2 stack, you will install those components and then configure their properties during the upgrade procedure.

In preparation for future HDP 2.2 releases to support rolling upgrades, the HDP RPM package version naming convention has changed to include the HDP 2.2 product version in file and directory names. HDP 2.2 marks the first release where HDP rpms, debs, and directories contain versions in the names to permit side-by-side installations of later HDP releases. To transition between previous releases and HDP 2.2, Hortonworks provides hdp-select, a script that symlinks your directories to `hdp/current` and lets you maintain using the same binary and configuration paths that you were using before.

The following instructions have you remove your older version HDP components, install hdp-select, and install HDP 2.2 components to prepare for rolling upgrade.

Prepare the 2.1 Stack for Upgrade

To prepare for upgrading the HDP Stack, perform the following tasks:

- Disable Security.

If your Stack has Kerberos Security turned on, turn it off before performing the upgrade. On `Ambari Web UI` > `Admin` > `Security`, click `Disable Security`. You can turn Kerberos Security on again after performing the upgrade.

- Checkpoint user metadata and capture the HDFS operational state.

  This step supports rollback and restore of the original state of HDFS data, if necessary.
- Backup Hive and Oozie metastore databases.

  This step supports rollback and restore of the original state of Hive and Oozie data, if necessary.
- Stop all HDP and Ambari services.
- Make sure to finish all current jobs running on the system before upgrading the stack.

Libraries will change during the upgrade. Any jobs remaining active that use the older version libraries will probably fail during the upgrade.

1. Use Ambari Web > Services > Service Actions to stop all services except HDFS and ZooKeeper.
2. Stop any client programs that access HDFS.

Perform steps 3 through 8 on the NameNode host. In a highly-available NameNode configuration, execute the following procedure on the primary NameNode.

To locate the primary NameNode in an Ambari-managed HDP cluster, browse Ambari Web > Services > HDFS. In Summary, click NameNode. Hosts > Summary displays the host name FQDN.

3. If HDFS is in a non-finalized state from a prior upgrade operation, you must finalize HDFS before upgrading further. Finalizing HDFS will remove all links to the metadata of the prior HDFS version. Do this only if you do not want to rollback to that prior HDFS version.

   On the NameNode host, as the HDFS user,

   ```
   su -l <HDFS_USER>
   hdfs dfsadmin -finalizeUpgrade
   ```

   where <HDFS_USER> is the HDFS Service user. For example, hdfs.
4. Check the NameNode directory to ensure that there is no snapshot of any prior HDFS upgrade.

   Specifically, using Ambari Web > HDFS > Configs > NameNode, examine the <dfs.namenode.name.dir> or the <dfs.name.dir> directory in the NameNode Directories property. Make sure that only a "\current" directory and no "\previous" directory exists on the NameNode host.
5. Create the following logs and other files.

   Creating these logs allows you to check the integrity of the file system, post-upgrade.

   As the HDFS user,

   ```
   su -l <HDFS_USER>
   ```

   where <HDFS_USER> is the HDFS Service user. For example, hdfs.

   1. Run fsck with the following flags and send the results to a log.

The resulting file contains a complete block map of the file system. You use this log later to confirm the upgrade.

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

2 Optional: Capture the complete namespace of the file system.

The following command does a recursive listing of the root file system:

```
hadoop dfs -ls -R / > dfs-old-lsr-1.log
```

3 Create a list of all the DataNodes in the cluster.

```
hdfs dfsadmin -report > dfs-old-report-1.log
```

4 Optional: Copy all unrecoverable data stored in HDFS to a local file system or to a backup instance of HDFS.

6 **Save the namespace.**

You must be the HDFS service user to do this and you must put the cluster in Safe Mode.

```
hdfs dfsadmin -safemode enter
hdfs dfsadmin -saveNamespace
```

In a highly-available NameNode configuration, the command `hdfs dfsadmin -saveNamespace` sets a checkpoint in the first NameNode specified in the configuration, in `dfs.ha.namenodes.[nameservice ID]`. You can also use the dfsadmin `-fs` option to specify which NameNode to connect.

For example, to force a checkpoint in namenode 2:

```
hdfs dfsadmin -fs hdfs://namenode2-hostname:namenode2-port -saveNamespace
```

7 Copy the checkpoint files located in `<dfs.name.dir/current>` into a backup directory.

Find the directory, using Ambari Web > HDFS > Configs > NameNode > NameNode Directories on your primary NameNode host.

In a highly-available NameNode configuration, the location of the checkpoint depends on where the saveNamespace command is sent, as defined in the preceding step.

8 Store the layoutVersion for the NameNode.

Make a copy of the file at <dfs.name.dir>/current/VERSION, where <dfs.name.dir> is the value of the config parameter NameNode directories. This file will be used later to verify that the layout version is upgraded.

9 Stop HDFS.

10 Stop ZooKeeper.

11 Using Ambari Web > Services > <service.name> > Summary, review each service and make sure that all services in the cluster are completely stopped.

12 At the Hive Metastore database host, stop the Hive metastore ***service***, if you have not done so already.

Make sure that the Hive metastore ***database*** is running. For more information about Administering the Hive metastore database, see the Hive Metastore Administrator documentation.

13 If you are upgrading Hive and Oozie, back up the Hive and Oozie metastore databases on the Hive and Oozie database host machines, respectively.

Make sure that your Hive database is updated to the minimum recommended version.

If you are using Hive with MySQL, we recommend upgrading your MySQL database to version 5.6.21 before upgrading the HDP Stack to v2.2.

For specific information, see Database Requirements.

1 Optional - Back up the Hive Metastore database.

These instructions are provided for your convenience. Please check your database documentation for the latest back up instructions.

Table 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

Oracle	Connect to the Oracle database using sqlplus export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database ile=input_file.dmp

2 Optional - Back up the Oozie Metastore database.

These instructions are provided for your convenience. Please check your database documentation for the latest back up instructions.

Table 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

14 Backup Hue.
If you are using the embedded SQLite database, you must perform a backup of the database before you upgrade Hue to prevent data loss. To make a backup copy of the database, stop Hue, then "dump" the database content to a file, as follows:

```
./etc/init.d/hue stop

su $HUE_USER
mkdir ~/hue_backup
cd /var/lib/hue
sqlite3 desktop.db .dump > ~/hue_backup/desktop.bak
```

For other databases, follow your vendor-specific instructions to create a backup.

15 On the Ambari Server host, stop Ambari Server and confirm that it is stopped.

```
ambari-server stop
ambari-server status
```

16 Stop all Ambari Agents.
On every host in your cluster known to Ambari,

```
ambari-agent stop
```

Upgrade the 2.1 Stack to 2.2

1 Upgrade the HDP repository on all hosts and replace the old repository file with the new file:

- **For RHEL/CentOS/Oracle Linux 6:**

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/centos6/2.x/GA/2.2.0.0/hdp.repo -O
/etc/yum.repos.d/HDP.repo
```

- **For SLES 11 SP3:**

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/suse11sp3/2.x/GA/2.2.0.0/hdp.repo -O
/etc/zypp/repos.d/HDP.repo
```

- **For SLES 11 SP1:**

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/sles11sp1/2.x/GA/2.2.0.0/hdp.repo -O
/etc/zypp/repos.d/HDP.repo
```

- **For UBUNTU12:**

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/ubuntu1/2.x/GA/2.2.0.0/hdp.list -O
/etc/apt/sourceslist.d/HDP.list
```

- **For RHEL/CentOS/Oracle Linux 5:** (DEPRECATED)

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/centos5/2.x/GA/2.2.0.0/hdp.repo -O
/etc/yum.repos.d/HDP.repo
```

Make sure to download the HDP.repo file under `/etc/yum.repos.d` on ALL hosts.

2 Update the Stack version in the Ambari Server database.

On the Ambari Server host, use the following command to update the Stack version to HDP-2.2:

```
ambari-server upgradestack HDP-2.2
```

3 Back up the files in following directories on the Oozie server host and make sure that all files, including *site.xml files are copied.

```
mkdir oozie-conf-bak
cp -R /etc/oozie/conf/* oozie-conf-bak
```

4 Remove the old oozie directories on all Oozie server and client hosts

- `rm -rf /etc/oozie/conf`
- `rm -rf /usr/lib/oozie/`
- `rm -rf /var/lib/oozie/`

5 Upgrade the Stack on all Ambari Agent hosts.

For each host, identify the HDP components installed on that host. Use Ambari Web, as described here, to view components on each host in your cluster. Based on the HDP components installed, edit the following upgrade commands for each host to upgrade only those components residing on that host.

For example, if you know that a host has ***no*** HBase service or client packages installed, then you can edit the command to ***not*** include HBase, as follows:

```
yum install "collectd*" "gccxml*" "pig*" "hdfs*" "sqoop*"
"zookeeper*" "hive*"
```

If you are writing to multiple systems using a script, do not use " " with the run command. You can use " " with pdsh -y.

- For RHEL/CentOS/Oracle Linux:

1 On all hosts, clean the yum repository.

```
yum clean all
```

2 Remove all HDP 2.1 components that you want to upgrade.

This command un-installs the HDP 2.1 component bits. It leaves the user data and metadata, but removes your configurations.

```
yum erase "hadoop*" "webhcat*" "hcatalog*" "oozie*" "pig*" "hdfs*" "sqoop*"
"zookeeper*" "hbase*" "hive*" "tez*" "storm*" "falcon*" "flume*" "phoenix*"
"accumulo*" "mahout*" "hue*" "hdp_mon_nagios_addons"
```

3 Remove your old hdp.repo and hdp-utils repo files.

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

4 Install all HDP 2.2 components that you want to upgrade.

```
yum install "hadoop_2_2_0_0_*" "oozie_2_2_0_0_*" "pig_2_2_0_0_*"
"sqoop_2_2_0_0_*" "zookeeper_2_2_0_0_*" "hbase_2_2_0_0_*" "hive_2_2_0_0_*"
"tez_2_2_0_0_*" "storm_2_2_0_0_*" "falcon_2_2_0_0_*" "flume_2_2_0_0_*"
"phoenix_2_2_0_0_*" "accumulo_2_2_0_0_*" "mahout_2_2_0_0_*"
rpm -e --nodeps hue-shell
yum install hue hue-common hue-beeswax hue-hcatalog hue-pig hue-oozie
```

5 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.

- For SLES:

1 On all hosts, clean the zypper repository.

```
zypper clean --all
```

2 Remove all HDP 2.1 components that you want to upgrade.

This command un-installs the HDP 2.1 component bits. It leaves the user data and metadata, but removes your configurations.

```
zypper remove "hadoop*" "webhcat*" "hcatalog*" "oozie*" "pig*" "hdfs*"
"sqoop*" "zookeeper*" "hbase*" "hive*" "tez*" "storm*" "falcon*" "flume*"
"phoenix*" "accumulo*" "mahout*" "hue*" "hdp_mon_nagios_addons"
```

3 Remove your old hdp.repo and hdp-utils repo files.

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

4 Install all HDP 2.2 components that you want to upgrade.

```
zypper install "hadoop\_2_2_0_0_*" "oozie\_2_2_0_0_*" "pig\_2_2_0_0_*"
"sqoop\_2_2_0_0_*" "zookeeper\_2_2_0_0_*" "hbase\_2_2_0_0_*"
"hive\_2_2_0_0_*" "tez\_2_2_0_0_*" "storm\_2_2_0_0_*" "falcon\_2_2_0_0_*"
"flume\_2_2_0_0_*" "phoenix\_2_2_0_0_*" "accumulo\_2_2_0_0_*"
"mahout\_2_2_0_0_*"
 rpm -e --nodeps hue-shell
zypper install hue hue-common hue-beeswax hue-hcatalog hue-pig hue-oozie
```

5 Verify that the components were upgraded.

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

No component files names should appear in the returned list.

6 If any components were not upgraded, upgrade them as follows:

```
yast --update hdfs hcatalog hive
```

6 Symlink directories, using `hdp-select`.

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.

Check that the `hdp-select` package installed:

```
rpm -qa | grep hdp-select
```

You should see: `hdp-select-2.2.0.0-2041.el6.noarch` for the HDP 2.2 release.

If not, then run:

```
yum install hdp-select
```

Run `hdp-select` as root, on every node.

```
hdp-select set all 2.2.0.0-<$version>
```

where `$version` is the build number. For the HDP 2.2 release <$version> = 2041.

7 On the Hive Metastore database host, stop the Hive Metastore ***service***, if you have not done so already. Make sure that the Hive Metastore ***database*** is running.

8 Upgrade the Hive metastore database schema from v13 to v14, using the following instructions:

- Set java home:

  ```
  export JAVA_HOME=/path/to/java
  ```

- Copy (rewrite) old Hive configurations to new conf dir:

  ```
  cp -R /etc/hive/conf.server/* /etc/hive/conf/
  ```

- Copy jdbc connector to `/usr/hdp/<$version>/hive/lib`, if it is not already in that location.

- `<HIVE_HOME>/bin/schematool -upgradeSchema -dbType<databaseType>`

  where `<HIVE_HOME>` is the Hive installation directory.

  For example, on the Hive Metastore host:

  /usr/hdp/2.2.0.0-`<$version>`/hive/bin/schematool -upgradeSchema -dbType `<databaseType>`

  where `<$version>` is the 2.2.0 build number and `<databaseType>` is derby, mysql, oracle, or postgres.

Complete the Upgrade of the 2.1 Stack to 2.2

1 Start Ambari Server.

On the Ambari Server host,

```
ambari-server start
```

2 Start all Ambari Agents.

At each Ambari Agent host,

```
ambari-agent start
```

3 Update the repository Base URLs in Ambari Server for the HDP-2.2 stack.

Browse to **Ambari Web > Admin > Repositories,** then set the values for the HDP and HDP-UTILS repository Base URLs. For more information about viewing and editing repository Base URLs, see Viewing Cluster Stack Version and Repository URLs.

For a remote, accessible, public repository, the HDP and HDP-UTILS Base URLs are the same as the baseurl=values in the HDP.repo file downloaded in Upgrade the Stack: Step 1. For a local repository, use the local repository Base URL that you configured for the HDP Stack. For links to download the HDP repository files for your version of the Stack, see HDP Stack Repositories.

4 Using the Ambari Web UI, add the Tez service if if it has not been installed already. For more information about adding a service, see Adding a Service.

5 Using the Ambari Web UI, add any new services that you want to run on the HDP 2.2 stack. You must add a Service before editing configuration properties necessary to complete the upgrade.

6 Using the `Ambari Web UI` > `Services`, start the ZooKeeper service.

7 Copy (rewrite) old hdfs configurations to new conf directory, on all Datanode and Namenode hosts,

```
cp /etc/hadoop/conf.empty/hdfs-site.xml.rpmsave
/etc/hadoop/conf/hdfs-site.xml;
cp /etc/hadoop/conf.empty/hadoop-env.sh.rpmsave
/etc/hadoop/conf/hadoop-env.sh.xml;
cp /etc/hadoop/conf.empty/log4j.properties.rpmsave
/etc/hadoop/conf/log4j.properties;
cp /etc/hadoop/conf.empty/core-site.xml.rpmsave /etc/hadoop/conf/core-
site.xml
```

8 If you are upgrading from an HA NameNode configuration, start all JournalNodes.

At each JournalNode host, run the following command:

```
su -l <HDFS_USER> -c "/usr/hdp/2.2.0.0-<$version>/hadoop/sbin/hadoop-
daemon.sh start journalnode"
```

where `<HDFS_USER>` is the HDFS Service user. For example, hdfs.

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 will fail.

9 Because the file system version has now changed, you must start the NameNode manually. On the active NameNode host, as the HDFS user,

```
su -l <HDFS_USER> -c "export HADOOP_LIBEXEC_DIR=/usr/hdp/2.2.0.0-
<$version>/hadoop/libexec && /usr/hdp/2.2.0.0-
<$version>/hadoop/sbin/hadoop-daemon.sh start namenode -upgrade"
```

where <HDFS_USER> is the HDFS Service user. For example, hdfs.

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

In a NameNode HA configuration, this NameNode does not enter the standby state as usual. Rather, this NameNode immediately enters the active state, upgrades its local storage directories, and upgrades the shared edit log. At this point, the standby NameNode in the HA pair is still down, and not synchronized with the upgraded, active NameNode.

To re-establish HA, you must synchronize the active and standby NameNodes. To do so, bootstrap the standby NameNode by running the NameNode with the '-bootstrapStandby' flag. Do NOT start the standby NameNode with the '-upgrade' flag.

At the Standby NameNode,

```
su -l <HDFS_USER> -c "hdfs namenode -bootstrapStandby -force"
```

where <HDFS_USER> is the HDFS Service user. For example, hdfs.

The bootstrapStandby command downloads the most recent fsimage from the active NameNode into the <dfs.name.dir> directory on the standby NameNode. Optionally, you can access that directory to make sure the fsimage has been successfully downloaded. After verifying, start the ZKFailoverController, then start the standby NameNode using **Ambari Web > Hosts > Components**.

10 Start all DataNodes.

At each DataNode, as the HDFS user,

```
su -l <HDFS_USER> -c "/usr/hdp/2.2.0.0-<$version>/hadoop/sbin/hadoop-
daemon.sh --config /etc/hadoop/conf start datanode"
```

where <HDFS_USER> is the HDFS Service user. For example, hdfs.

The NameNode sends an upgrade command to DataNodes after receiving block reports.

11 Update HDFS Configuration Properties for HDP 2.2

Using `Ambari Web UI > Services > HDFS > Configs > core-site.xml`:

- Add

Name	Value
hadoop.http.authentication.simple.anonymous.allowed	true

`Using Ambari Web UI > Services > HDFS > Configs > hdfs-site.xml`:

- Add

Name	Value
dfs.namenode.startup.delay.block.deletion.sec	3600

- Modify

Name	Value
dfs.datanode.max.transfer.threads	4096

12 Restart HDFS.

1 Open the Ambari Web GUI. If the browser in which Ambari is running has been open throughout the process, clear the browser cache, then refresh the browser.

2 Choose **Ambari Web > Services > HDFS > Service Actions > Restart All**.

⚠ In a cluster configured for NameNode High Availability, use the following procedure to restart NameNodes. Using the following procedure preserves HA when upgrading the cluster.

1. Using Ambari Web > Services > HDFS, choose Active NameNode.
   This shows the host name of the current, active NameNode.
2. Write down (or copy, or remember) the host name of the active NameNode.
   You need this host name for step 4.
3. Using Ambari Web > Services > HDFS > Service Actions > choose Stop.
   This stops all of the HDFS Components, including both NameNodes.
4. Using Ambari Web > Hosts > choose the host name you noted in Step 2, then start that NameNode component, using Host Actions > Start.
   This causes the original, active NameNode to re-assume its role as the active NameNode.
5. Using Ambari Web > Services > HDFS > Service Actions, choose Re-Start All.

3 Choose **Service Actions > Run Service Check**. Makes sure the service check passes.

13 After the DataNodes are started, HDFS exits SafeMode. To monitor the status, run the following command, on each DataNode:

```
sudo su -l <HDFS_USER> -c "hdfs dfsadmin -safemode get"
```

where `<HDFS_USER>` is the HDFS Service user. For example, hdfs.

When HDFS exits SafeMode, the following message displays:

```
Safe mode is OFF
```

14 Make sure that the HDFS upgrade was successful.
Optionally, repeat step 5 in Prepare the 2.1 Stack for Upgrade to create new versions of the logs and reports, substituting "-new " for "-old " in the file names as necessary.

- Compare the old and new versions of the following log files:
- dfs-old-fsck-1.log versus dfs-new-fsck-1.log.

The files should be identical unless the hadoop fsck reporting format has changed in the new version.

- dfs-old-lsr-1.log versus dfs-new-lsr-1.log.

The files should be identical unless the format of hadoop fs -lsr reporting or the data structures have changed in the new version.

- dfs-old-report-1.log versus fs-new-report-1.log

Make sure that all DataNodes in the cluster before upgrading are up and running.

15 Upgrade Application Timeline Server (ATS) components for YARN.

- If upgrading your HDP Stack version from 2.1.1 or 2.1.2, you must modify the following YARN configuration property:

  Browse to **Ambari Web > Services > YARN Configs > Application Timeline Server**, and set
  yarn.timeline-service.store-class=org.apache.hadoop.yarn.server.timeline.LeveldbTimelineStore
- If YARN is installed in your HDP 2.1 stack, and the ATS components are NOT, then you must create and install ATS service and host components using the API.

  Run the following commands on the server that will host the YARN ATS in your cluster. Be sure to replace <your_ATS_component_hostname> with a host name appropriate for your environment.

  1 Create the ATS Service Component.

```
curl --user admin:admin -H "X-Requested-By: ambari" -i -X POST
http://localhost:8080/api/v1/clusters/<your_cluster_name>/services/YARN/compo
nents/APP_TIMELINE_SERVER
```

  2 Create the ATS Host Component.

```
curl --user admin:admin -H "X-Requested-By: ambari" -i -X POST
http://localhost:8080/api/v1/clusters/<your_cluster_name>/hosts/<your_ATS_com
ponent_hostname>/host_components/APP_TIMELINE_SERVER
```

  3 Install the ATS Host Component.

```
curl --user admin:admin -H "X-Requested-By: ambari" -i -X PUT -d
'{"HostRoles": { "state":  "INSTALLED"}}'
http://localhost:8080/api/v1/clusters/<your_cluster_name>/hosts/<your_ATS_com
ponent_hostname>/host_components/APP_TIMELINE_SERVER
```

curl commands use the default username/password = admin/admin. To run the curl commands using non-default credentials, modify the --user option to use your Ambari administrator credentials.

For example: --user <ambari_admin_username>:<ambari_admin_password>.

16 Prepare MR2 and Yarn for work. Execute HDFS commands on any host.

- Create mapreduce dir in hdfs.

```
su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.2.0.0-
<$version>/mapreduce/"
```

- Copy new mapreduce.tar.gz to HDFS mapreduce dir.

```
su -l <HDFS_USER> -c "hdfs dfs -copyFromLocal /usr/hdp/2.2.0.0-
<$version>/hadoop/mapreduce.tar.gz /hdp/apps/2.2.0.0-
<$version>/mapreduce/."
```

- Grant permissions for created mapreduce dir in hdfs.

```
su -l <HDFS_USER> -c "hdfs dfs -chown -R <HDFS_USER>:<HADOOP_GROUP> /hdp";
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.2.0.0-
<$version>/mapreduce";
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.2.0.0-
<$version>/mapreduce/mapreduce.tar.gz"
```

- Update YARN Configuration Properties for HDP 2.2

On ambari-server host,
`cd /var/lib/ambari-server/resources/scripts`

↳ then run the following scripts:

```
./configs.sh set localhost <your.cluster.name> capacity-scheduler
"yarn.scheduler.capacity.resource-calculator"
"org.apache.hadoop.yarn.util.resource.DefaultResourceCalculator";
./configs.sh set localhost <your.cluster.name> capacity-scheduler
"yarn.scheduler.capacity.root.accessible-node-labels" "*";
./configs.sh set localhost <your.cluster.name> capacity-scheduler
"yarn.scheduler.capacity.root.accessible-node-labels.default.capacity" "-1";
./configs.sh set localhost <your.cluster.name> capacity-scheduler
"yarn.scheduler.capacity.root.accessible-node-labels.default.maximum-
capacity" "-1";
./configs.sh set localhost <your.cluster.name> capacity-scheduler
"yarn.scheduler.capacity.root.default-node-label-expression" ""
```

Using Ambari Web UI > Service > Yarn > Configs > Advanced > yarn-site:

↳ Add

Name	Value
yarn.application.classpath	$HADOOP_CONF_DIR,/usr/hdp/current/hadoop-client/*, /usr/hdp/current/hadoop-client/lib/*, /usr/hdp/current/hadoop-hdfs-client/*, /usr/hdp/current/hadoop-hdfs-client/lib/*, /usr/hdp/current/hadoop-yarn-client/* ,/usr/hdp/current/hadoop-yarn-client/lib/*
hadoop.registry.zk.quorum	<!--List of hostname:port pairs defining the zookeeper quorum binding for the registry-->
hadoop.registry.rm.enabled	false
yarn.client.nodemanager-connect.max-wait-ms	900000
yarn.client.nodemanager-connect.retry-interval-ms	10000
yarn.node-labels.fs-store.retry-policy-spec	2000, 500
yarn.node-labels.fs-store.root-dir	/system/yarn/node-labels
yarn.node-labels.manager-class	org.apache.hadoop.yarn.server.resourcemanager.nodelabels.MemoryRMNodeLabelsManager
yarn.nodemanager.bind-host	0.0.0.0
yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage	90
yarn.nodemanager.disk-health-checker.min-free-space-per-disk-mb	1000
yarn.nodemanager.linux-container-executor.cgroups.hierarchy	hadoop-yarn
yarn.nodemanager.linux-container-executor.cgroups.mount	false
yarn.nodemanager.linux-container-executor.cgroups.strict-resource-usage	false
yarn.nodemanager.linux-container-executor.resources-handler.class	org.apache.hadoop.yarn.server.nodemanager.util.DefaultLCEResourcesHandler
yarn.nodemanager.log-aggregation.debug-enabled	false
yarn.nodemanager.log-aggregation.num-log-files-per-app	30
yarn.nodemanager.log-aggregation.roll-monitoring-interval-seconds	-1
yarn.nodemanager.recovery.dir	/var/log/hadoop-yarn/nodemanager/recovery-state

yarn.nodemanager.recovery.enabled	false
yarn.nodemanager.resource.cpu-vcores	1
yarn.nodemanager.resource.percentage-physical-cpu-limit	100
yarn.resourcemanager.bind-host	0.0.0.0
yarn.resourcemanager.connect.max-wait.ms	900000
yarn.resourcemanager.connect.retry-interval.ms	30000
yarn.resourcemanager.fs.state-store.retry-policy-spec	2000, 500
yarn.resourcemanager.fs.state-store.uri	<enter a "space" as the property value>
yarn.resourcemanager.ha.enabled	false
yarn.resourcemanager.recovery.enabled	false
yarn.resourcemanager.state-store.max-completed-applications	${yarn.resourcemanager.max-completed-applications}
yarn.resourcemanager.store.class	org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore
yarn.resourcemanager.system-metrics-publisher.dispatcher.pool-size	10
yarn.resourcemanager.system-metrics-publisher.enabled	true
yarn.resourcemanager.webapp.delegation-token-auth-filter.enabled	false
yarn.resourcemanager.work-preserving-recovery.enabled	false
yarn.resourcemanager.work-preserving-recovery.scheduling-wait-ms	10000
yarn.resourcemanager.zk-acl	world:anyone:rwcda
yarn.resourcemanager.zk-address	localhost:2181
yarn.resourcemanager.zk-num-retries	1000
yarn.resourcemanager.zk-retry-interval-ms	1000
yarn.resourcemanager.zk-state-store.parent-path	/rmstore
yarn.resourcemanager.zk-timeout-ms	10000
yarn.timeline-service.bind-host	0.0.0.0

yarn.timeline-service.client.max-retries	30
yarn.timeline-service.client.retry-interval-ms	1000
yarn.timeline-service.http-authentication.simple.anonymous.allowed	true
yarn.timeline-service.http-authentication.type	simple
yarn.timeline-service.leveldb-timeline-store.read-cache-size	104857600
yarn.timeline-service.leveldb-timeline-store.start-time-read-cache-size	10000
yarn.timeline-service.leveldb-timeline-store.start-time-write-cache-size	10000

↳ Modify

Name	Value
yarn.timeline-service.webapp.address	<PUT_THE_FQDN_OF_ATS_HOST_NAME_HERE>:8188
yarn.timeline-service.webapp.https.address	<PUT_THE_FQDN_OF_ATS_HOST_NAME_HERE>:8190
yarn.timeline-service.address	<PUT_THE_FQDN_OF_ATS_HOST_NAME_HERE>:10200

- Update MapReduce2 Configuration Properties for HDP 2.2

  Using Ambari Web UI > Services > MapReduce2 > Configs > mapred-site.xml:

↳ Add

Name	Value
mapreduce.job.emit-timeline-data	false
mapreduce.jobhistory.bind-host	0.0.0.0
mapreduce.reduce.shuffle.fetch.retry.enabled	1
mapreduce.reduce.shuffle.fetch.retry.interval-ms	1000
mapreduce.reduce.shuffle.fetch.retry.timeout-ms	30000
mapreduce.application.framework.path	/hdp/apps/${hdp.version}/mapreduce/mapreduce.tar.gz#mr-framework

↳ Modify

Name	Value
mapreduce.admin.map.child.java.opts	-server -XX:NewRatio=8 -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.version}
mapreduce.admin.reduce.child.java.opts	-server -XX:NewRatio=8 -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.version}

yarn.app.mapreduce.am.admin-command-opts	-Dhdp.version=${hdp.version}
yarn.app.mapreduce.am.command-opts	-Xmx546m -Dhdp.version=${hdp.version}
mapreduce.application.classpath	$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.${hdp.version}.jar:/etc/hadoop/conf/secure$
mapreduce.admin.user.env	LD_LIBRARY_PATH=/usr/hdp/${hdp.version}/hadoop/lib/native:/usr/hdp/${hdp.version}/hadoop/lib/native/Linux-amd64-64

- Update HBase Configuration Properties for HDP 2.2

  Using Ambari Web UI > Services > HBase > Configs > hbase-site.xml:

  - Add

Name	Value
hbase.hregion.majorcompaction.jitter	0.50

  - Modify

Name	Value
hbase.hregion.majorcompaction	604800000
hbase.hregion.memstore.block.multiplier	4

  - Remove

Name	Value
hbase.hstore.flush.retries.number	120

- Update Hive Configuration Properties for HDP 2.2

  Using Ambari Web UI > Services > Hive > Configs > hive-site.xml:

  - Add

Name	Value
hive.cluster.delegation.token.store.zookeeper.connectString	<!-- The ZooKeeper token store connect string. -->
hive.auto.convert.sortmerge.join.to.mapjoin	false
hive.cbo.enable	true
hive.cli.print.header	false
hive.cluster.delegation.token.store.class	org.apache.hadoop.hive.thrift.ZooKeeperTokenStore
hive.cluster.delegation.token.store.zookeeper.znode	/hive/cluster/delegation

hive.conf.restricted.list	hive.security.authenticator.manager,hive.security.authorization.manager,hive.users.in.admin.role
hive.convert.join.bucket.mapjoin.tez	false
hive.exec.compress.intermediate	false
hive.exec.compress.output	false
hive.exec.dynamic.partition	true
hive.exec.dynamic.partition.mode	nonstrict
hive.exec.max.created.files	100000
hive.exec.max.dynamic.partitions	5000
hive.exec.max.dynamic.partitions.pernode	2000
hive.exec.orc.compression.strategy	SPEED
hive.exec.orc.default.compress	ZLIB
hive.exec.orc.default.stripe.size	67108864
hive.exec.parallel	false
hive.exec.parallel.thread.number	8
hive.exec.reducers.bytes.per.reducer	67108864
hive.exec.reducers.max	1009
hive.exec.scratchdir	/tmp/hive
hive.exec.submit.local.task.via.child	true
hive.exec.submitviachild	false
hive.fetch.task.aggr	false
hive.fetch.task.conversion	more
hive.fetch.task.conversion.threshold	1073741824
hive.map.aggr.hash.force.flush.memory.threshold	0.9
hive.map.aggr.hash.min.reduction	0.5
hive.map.aggr.hash.percentmemory	0.5
hive.mapjoin.optimized.hashtable	true
hive.merge.mapfiles	true
hive.merge.mapredfiles	false
hive.merge.orcfile.stripe.level	true
hive.merge.rcfile.block.level	true
hive.merge.size.per.task	256000000
hive.merge.smallfiles.avgsize	16000000
hive.merge.tezfiles	false
hive.metastore.authorization.storage.checks	false
hive.metastore.client.connect.retry.delay	5s
hive.metastore.connect.retries	24
hive.metastore.failure.retries	24
hive.metastore.server.max.threads	100000
hive.optimize.constant.propagation	true
hive.optimize.metadataonly	true
hive.optimize.null.scan	true
hive.optimize.sort.dynamic.partition	false
hive.orc.compute.splits.num.threads	10
hive.prewarm.enabled	false
hive.prewarm.numcontainers	10

hive.security.metastore.authenticator.manager	org.apache.hadoop.hive.ql.security.HadoopDefaultMetastoreAuthenticator
hive.security.metastore.authorization.auth.reads	true
hive.server2.allow.user.substitution	true
hive.server2.logging.operation.enabled	true
hive.server2.logging.operation.log.location	${system:java.io.tmpdir}/${system:user.name}/operation_logs
hive.server2.table.type.mapping	CLASSIC
hive.server2.thrift.http.path	cliservice
hive.server2.thrift.http.port	10001
hive.server2.thrift.max.worker.threads	500
hive.server2.thrift.sasl.qop	auth
hive.server2.transport.mode	binary
hive.server2.use.SSL	false
hive.smbjoin.cache.rows	10000
hive.stats.dbclass	fs
hive.stats.fetch.column.stats	false
hive.stats.fetch.partition.stats	true
hive.support.concurrency	false
hive.tez.auto.reducer.parallelism	false
hive.tez.cpu.vcores	-1
hive.tez.dynamic.partition.pruning	true
hive.tez.dynamic.partition.pruning.max.data.size	104857600
hive.tez.dynamic.partition.pruning.max.event.size	1048576
hive.tez.log.level	INFO
hive.tez.max.partition.factor	2.0
hive.tez.min.partition.factor	0.25
hive.tez.smb.number.waves	0.5
hive.user.install.directory	/user/
hive.vectorized.execution.reduce.enabled	false
hive.zookeeper.client.port	2181
hive.zookeeper.namespace	hive_zookeeper_namespace
hive.zookeeper.quorum	<!-- List of zookeeper server to talk to -->

- Modify

Name	Value
hive.metastore.client.socket.timeout	1800s
hive.optimize.reducededuplication.min.reducer	4
hive.security.authorization.manager	org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQLStdConfOnlyAuthorizerFactory

hive.security.metastore.authorization.manager	org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvider, org.apache.hadoop.hive.ql.security.authorization.MetaStoreAuthzAPIAuthorizerEmbedOnly
hive.server2.support.dynamic.service.discovery	true
hive.vectorized.groupby.checkinterval	4096
fs.file.impl.disable.cache	true
fs.hdfs.impl.disable.cache	true

17 Using Ambari Web > Services > Service Actions, start YARN.

18 Using Ambari Web > Services > Service Actions, start MapReduce2.

19 Using Ambari Web > Services > Service Actions, start HBase and ensure the service check passes.

20 Using Ambari Web > Services > Service Actions, start the Hive service.

21 Upgrade Oozie.

1 Perform the following preparation steps on each Oozie server host:

You must replace your Oozie configuration after upgrading.

1 Copy configurations from **oozie-conf-bak** to the **/etc/oozie/conf** directory on each Oozie server and client.

2 Create `/usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22` directory.

```
mkdir /usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22
```

3 Copy the JDBC jar of your Oozie database to both `/usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22` and `/usr/hdp/2.2.0.0-<$version>/oozie/libtools`.

For example, if you are using MySQL, copy your `mysql-connector-java.jar`.

4 Copy these files to `/usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22` directory

```
cp /usr/lib/hadoop/lib/hadoop-lzo*.jar /usr/hdp/2.2.0.0-
<$version>/oozie/libext-upgrade22;
cp /usr/share/HDP-oozie/ext-2.2.zip /usr/hdp/2.2.0.0-<$version>/oozie/libext-
upgrade22;
cp /usr/share/HDP-oozie/ext-2.2.zip /usr/hdp/2.2.0.0-<$version>/oozie/libext
```

5 Grant read/write access to the Oozie user.

```
chmod -R 777 /usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22
```

2 Upgrade steps:

1 On the Services view, make sure that YARN and MapReduce2 services are running.

2. Make sure that the Oozie service is stopped.
3. In `/etc/oozie/conf/oozie-env.sh`, comment out `CATALINA_BASE` property, also do the same using Ambari Web UI in `Services` > `Oozie` > `Configs` > `Advanced oozie-env`.
4. Upgrade Oozie.

   At the Oozie database host, as the Oozie service user:

```
sudo su -l <OOZIE_USER> -c"/usr/hdp/2.2.0.0-<$version>/oozie/bin/ooziedb.sh upgrade -run"
```

where `<OOZIE_USER>` is the Oozie service user. For example, oozie.

Make sure that the output contains the string "Oozie DB has been upgraded to Oozie version `<OOZIE_Build_Version>`.

5. Prepare the Oozie WAR file.

The Oozie server must be **not** running for this step. If you get the message "ERROR: Stop Oozie first", it means the script still thinks it's running. Check, and if needed, remove the process id (pid) file indicated in the output. You may see additional "File Not Found" error messages during a successful upgrade of Oozie.

On the Oozie server, as the Oozie user

```
sudo su -l <OOZIE_USER> -c "/usr/hdp/2.2.0.0-<$version>/oozie/bin/oozie-setup.sh prepare-war -d /usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22"
```

where `<OOZIE_USER>` is the Oozie service user. For example, oozie.
Make sure that the output contains the string "New Oozie WAR file added".

6. Using **Ambari Web**, choose `Services` > `Oozie` > `Configs`, expand `oozie-log4j`, then add the following property:

```
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.

7. Using `Ambari Web` > `Services` > `Oozie` > `Configs`, expand **Advanced oozie-site**, then edit the following properties:

A. In `oozie.service.coord.push.check.requeue.interval`, ***replace*** the existing property value with the following one:

```
30000
```

B. In `oozie.service.URIHandlerService.uri.handlers`, ***append*** to the existing property value the following string, if is it is not already present:

```
org.apache.oozie.dependency.FSURIHandler,org.apache.oozie.dependency.HCatURIHandler
```

C. In `oozie.services`, make sure all the following properties are present:

```
org.apache.oozie.service.SchedulerService,
org.apache.oozie.service.InstrumentationService,
org.apache.oozie.service.MemoryLocksService,
org.apache.oozie.service.UUIDService,
org.apache.oozie.service.ELService,
org.apache.oozie.service.AuthorizationService,
org.apache.oozie.service.UserGroupInformationService,
org.apache.oozie.service.HadoopAccessorService,
org.apache.oozie.service.JobsConcurrencyService,
org.apache.oozie.service.URIHandlerService,
org.apache.oozie.service.DagXLogInfoService,
org.apache.oozie.service.SchemaService,
org.apache.oozie.service.LiteWorkflowAppService,
org.apache.oozie.service.JPAService,
org.apache.oozie.service.StoreService,
org.apache.oozie.service.CoordinatorStoreService,
org.apache.oozie.service.SLAStoreService,
org.apache.oozie.service.DBLiteWorkflowStoreService,
org.apache.oozie.service.CallbackService,
org.apache.oozie.service.ActionService,
org.apache.oozie.service.ShareLibService,
org.apache.oozie.service.CallableQueueService,
org.apache.oozie.service.ActionCheckerService,
org.apache.oozie.service.RecoveryService,
org.apache.oozie.service.PurgeService,
org.apache.oozie.service.CoordinatorEngineService,
org.apache.oozie.service.BundleEngineService,
org.apache.oozie.service.DagEngineService,
org.apache.oozie.service.CoordMaterializeTriggerService,
org.apache.oozie.service.StatusTransitService,
org.apache.oozie.service.PauseTransitService,
org.apache.oozie.service.GroupsService,
org.apache.oozie.service.ProxyUserService,
org.apache.oozie.service.XLogStreamingService,
org.apache.oozie.service.JvmPauseMonitorService
```

D Add the `oozie.service.AuthorizationService.security.enabled` property with the following property value: `false`

Specifies whether security (user name/admin role) is enabled or not. If disabled any user can manage Oozie system and manage any job.

E Add the `oozie.service.HadoopAccessorService.kerberos.enabled` property with the following property value: `false`

Indicates if Oozie is configured to use Kerberos.

F In `oozie.services.ext`, ***append*** to the existing property value the following string, if is it is not already present:

```
org.apache.oozie.service.PartitionDependencyManagerService,org.apache.oozie.service.HCatAccessorService
```

G After modifying all properties on the Oozie Configs page, choose **Save** to update `oozie.site.xml`, using the modified configurations.

8 Replace the content of `/usr/oozie/share` in HDFS.

On the Oozie server host:

1 Extract the Oozie sharelib into a `tmp` folder.

```
mkdir -p /tmp/oozie_tmp;
cp /usr/hdp/2.2.0.0-<$version>/oozie/oozie-sharelib.tar.gz /tmp/oozie_tmp;
cd /tmp/oozie_tmp;
tar xzvf oozie-sharelib.tar.gz;
```

2 Back up the `/user/oozie/share` folder in HDFS and then delete it.

If you have any custom files in this folder, back them up separately and then add them to the /share folder after updating it.

```
mkdir /tmp/oozie_tmp/oozie_share_backup;
chmod 777 /tmp/oozie_tmp/oozie_share_backup;

su -l <HDFS_USER> -c "hdfs dfs -copyToLocal /user/oozie/share
/tmp/oozie_tmp/oozie_share_backup";
su -l <HDFS_USER> -c "hdfs dfs -rm -r /user/oozie/share";
```

where `<HDFS_USER>` is the HDFS service user. For example, hdfs.

3 Add the latest share libs that you extracted in step 1. After you have added the files, modify ownership and acl.

```
su -l <HDFS_USER> -c "hdfs dfs -copyFromLocal /tmp/oozie_tmp/share
/user/oozie/.";
su -l <HDFS_USER> -c "hdfs dfs -chown -R <OOZIE_USER>:<HADOOP_GROUP>
/user/oozie";
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 755 /user/oozie";
```

where `<HDFS_USER>` is the HDFS service user. For example, hdfs.

3 Update Oozie Configuration Properties for HDP 2.2

Using `Ambari Web UI > Services > Oozie > Configs > oozie-site.xml`:

- Add

Name	Value
oozie.authentication.simple.anonymous.allowed	true
oozie.service.coord.check.maximum.frequency	false
oozie.service.HadoopAccessorService.kerberos.enabled	false

- Modify

Name	Value
oozie.service.SchemaService.wf.ext.schemas	shell-action-0.1.xsd,shell-action-0.2.xsd,shell-action-0.3.xsd,email-action-0.1.xsd,email-action-0.2.xsd, hive-action-0.2.xsd,hive-action-0.3.xsd,hive-action-0.4.xsd,hive-action-0.5.xsd,sqoop-action-0.2.xsd, sqoop-action-0.3.xsd,sqoop-action-0.4.xsd,ssh-action-0.1.xsd,ssh-action-0.2.xsd,distcp-action-0.1.xsd, distcp-action-0.2.xsd,oozie-sla-0.1.xsd,oozie-sla-0.2.xsd
oozie.services.ext	org.apache.oozie.service.JMSAccessorService,org.apache.oozie.service.PartitionDependencyManagerService, org.apache.oozie.service.HCatAccessorService

22 Use the Ambari Web UI > Services view to start the Oozie service.

Make sure that ServiceCheck passes for Oozie.

23 Update WebHCat.

A Modify the webhcat-site config type.

Using Ambari Web > Services > WebHCat, modify the following configuration:

Action	Property Name	Property Value
Modify	templeton.storage.class	org.apache.hive.hcatalog.templeton.tool.ZooKeeperStorage

B Expand Advanced > webhcat-site.xml.

Check if property templeton.port exists. If not, then add it using the Custom webhcat-site panel. The default value for templeton.port = 50111.

C On each WebHCat host, update the Pig and Hive tar bundles, by updating the following files:

- /apps/webhcat/pig.tar.gz
- /apps/webhcat/hive.tar.gz

Find these files only on a host where WebHCat is installed.

For example, to update a *.tar.gz file:

1 Move the file to a local directory.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -copyToLocal
/apps/webhcat/*.tar.gz <local_backup_dir>"
```

2 Remove the old file.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -rm
/apps/webhcat/*.tar.gz"
```

3 Copy the new file.

```
su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal
/usr/hdp/2.2.0.0-<$version>/hive/hive.tar.gz /apps/webhcat/"; su -l
<HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal
/usr/hdp/2.2.0.0-<$version>/pig/pig.tar.gz /apps/webhcat/";
```

where <HCAT_USER> is the HCatalog service user. For example, hcat.

D On each WebHCat host, update /app/webhcat/hadoop-streaming.jar file.

1 Move the file to a local directory.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -copyToLocal
/apps/webhcat/hadoop-streaming*.jar <local_backup_dir>"
```

2 Remove the old file.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -rm
/apps/webhcat/hadoop-streaming*.jar"
```

3 Copy the new hadoop-streaming.jar file.

```
su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal
/usr/hdp/2.2.0.0-<$version>/hadoop-mapreduce/hadoop-streaming*.jar
/apps/webhcat"
```

where <HCAT_USER> is the HCatalog service user. For example, hcat.

24 If Tez was not installed during the upgrade, you must prepare Tez for work, using the following steps:

The Tez client should be available on the same host with Pig.

If you use Tez as the Hive execution engine, and if the variable hive.server2.enabled.doAs is set to true, you must create a scratch directory on the NameNode host for the username that will run the HiveServer2 service. If you installed Tez before upgrading the Stack, use the following commands:

```
sudo su -c "hdfs -makedir /tmp/hive- <username> "
sudo su -c "hdfs -chmod 777 /tmp/hive- <username> "
```

where <username> is the name of the user that runs the HiveServer2 service.

- Update Tez Configuration Properties for HDP 2.2

Using Ambari Web UI > Services > Tez > Configs > tez-site.xml:

↳ Add

Name	Value
tez.am.container.idle.release-timeout-max.millis	20000
tez.am.container.idle.release-timeout-min.millis	10000
tez.am.launch.cluster-default.cmd-opts	-server -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.version}
tez.am.launch.cmd-opts	-XX:+PrintGCDetails -verbose:gc -XX:+PrintGCTimeStamps -XX:+UseNUMA -XX:+UseParallelGC
tez.am.launch.env	LD_LIBRARY_PATH=/usr/hdp/${hdp.version}/hadoop/lib/native:/usr/hdp/${hdp.version}/hadoop/lib/native/Linux-amd64-64
tez.am.max.app.attempts	2

tez.am.maxtaskfailures.per.node	10
tez.cluster.additional.classpath.prefix	/usr/hdp/${hdp.version}/hadoop/lib/hadoop-lzo-0.6.0.${hdp.version}.jar:/etc/hadoop/conf/secure
tez.counters.max	2000
tez.counters.max.groups	1000
tez.generate.debug.artifacts	false
tez.grouping.max-size	1073741824
tez.grouping.min-size	16777216
tez.grouping.split-waves	1.7
tez.history.logging.service.class	org.apache.tez.dag.history.logging.ats.ATSHistoryLoggingService
tez.runtime.compress	true
tez.runtime.compress.codec	org.apache.hadoop.io.compress.SnappyCodec
tez.runtime.io.sort.mb	272
tez.runtime.unordered.output.buffer.size-mb	51
tez.shuffle-vertex-manager.max-src-fraction	0.4
tez.shuffle-vertex-manager.min-src-fraction	0.2
tez.task.am.heartbeat.counter.interval-ms.max	4000
tez.task.launch.cluster-default.cmd-opts	-server -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.version}
tez.task.launch.cmd-opts	-XX:+PrintGCDetails -verbose:gc -XX:+PrintGCTimeStamps -XX:+UseNUMA -XX:+UseParallelGC
tez.task.launch.env	LD_LIBRARY_PATH=/usr/hdp/${hdp.version}/hadoop/lib/native:/usr/hdp/${hdp.version}/hadoop/lib/native/Linux-amd64-64
tez.task.max-events-per-heartbeat	500
tez.task.resource.memory.mb	682

- Modify

Name	Value
tez.am.container.reuse.non-local-fallback.enabled	false
tez.am.resource.memory.mb	1364
tez.lib.uris	/hdp/apps/${hdp.version}/tez/tez.tar.gz
tez.session.client.timeout.secs	-1

- Remove

Name	Value
tez.am.container.session.delay-allocation-millis	10000
tez.am.env	LD_LIBRARY_PATH=/usr/hdp/2.2.0.0-1947/hadoop/lib/native:/usr/hdp/2.2.0.0-1947/hadoop/lib/native/Linux-amd64-64
tez.am.grouping.max-size	1073741824

tez.am.grouping.min-size	16777216
tez.am.grouping.split-waves	1.4
tez.am.java.opt	-server -Xmx546m -Djava.net.preferIPv4Stack=true -XX:+UseNUMA -XX:+UseParallelGC
tez.am.shuffle-vertex-manager.max-src-fraction	0.4
tez.am.shuffle-vertex-manager.min-src-fraction	0.2
tez.runtime.intermediate-input.compress.codec	org.apache.hadoop.io.compress.SnappyCodec
tez.runtime.intermediate-input.is-compressed	false
tez.runtime.intermediate-output.compress.codec	org.apache.hadoop.io.compress.SnappyCodec
tez.runtime.intermediate-output.should-compress	false
tez.yarn.ats.enabled	true

- Put Tez libraries in hdfs. Execute at any host:

```
su -l hdfs -c "hdfs dfs -mkdir -p /hdp/apps/2.2.0.0-<$version>/tez/"
su -l hdfs -c "hdfs dfs -copyFromLocal /usr/hdp/2.2.0.0-
<$version>/tez/lib/tez.tar.gz /hdp/apps/2.2.0.0-<$version>/tez/."
su -l hdfs -c "hdfs dfs -chown -R <HDFS_USER>:<HADOOP_GROUP> /hdp" su -l hdfs
-c "hdfs dfs -chmod -R 555 /hdp/apps/2.2.0.0-1899/tez" su -l hdfs -c "hdfs
dfs -chmod -R 444 /hdp/apps/2.2.0.0-1899/tez/tez.tar.gz"
```

25 Prepare the Storm service properties.

- Edit nimbus.childopts.

  Using Ambari Web UI > Services > Storm > Configs > Nimbus > find nimbus.childopts. Update the path for the jmxetric-1.0.4.jar to: /usr/hdp/current/storm-nimbus/contrib/storm-jmxetric/lib/jmxetric-1.0.4.jar. If nimbus.childopts property value contains "-Djava.security.auth.login.config=/path/to/storm_jaas.conf", remove this text.

- Edit supervisor.childopts.

  Using Ambari Web UI > Services > Storm > Configs > Supervisor > find supervisor.childopts. Update the path for the jmxetric-1.0.4.jar to: /usr/hdp/current/storm-nimbus/contrib/storm-jmxetric/lib/jmxetric-1.0.4.jar. If supervisor.childopts property value contains "-Djava.security.auth.login.config=/etc/storm/conf/storm_jaas.conf", remove this text.

- Edit worker.childopts.

Using Ambari Web UI > Services > Storm > Configs > Advanced > storm-site find worker.childopts. Update the path for the jmxetric-1.0.4.jar to: /usr/hdp/current/storm-nimbus/contrib/storm-jmxetric/lib/jmxetric-1.0.4.jar.

Check if the _storm.thrift.nonsecure.transport property exists. If not, add it, _storm.thrift.nonsecure.transport = backtype.storm.security.auth.SimpleTransportPlugin, using the Custom storm-site panel.

- Remove the `storm.local.dir` from every host where the Storm component is installed.

  You can find this property in the Storm > Configs > General tab.

  ```
  rm -rf <storm.local.dir>
  ```

- If you are planning to enable secure mode, navigate to `Ambari Web UI` > `Services` > `Storm` > `Configs` > `Advanced storm-site` and add the following property:

```
_storm.thrift.secure.transport=backtype.storm.security.auth.kerberos.KerberosSaslTransportPlugin
```

26 Upgrade Pig.

Copy the the Pig configuration files to /etc/pig/conf.

```
cp /etc/pig/conf.dist/pig.properties.rpmsave
/etc/pig/conf/pig.properties;
cp /etc/pig/conf.dist/pig-env.sh /etc/pig/conf/;
cp /etc/pig/conf.dist/log4j.properties.rpmsave
/etc/pig/conf/log4j.properties
```

27 Using `Ambari Web UI` > `Services` > `Storm`, start the Storm service.

28 Prepare the Falcon service properties:

- Update Falcon Configuration Properties for HDP 2.2

  Using `Ambari Web UI` > `Services` > `Falcon` > `Configs` > `falcon startup properties`:

  Add

Name	Value
*.application.services	org.apache.falcon.security.AuthenticationInitializationService,\ org.apache.falcon.workflow.WorkflowJobEndNotificationService,\ org.apache.falcon.service.ProcessSubscriberService,\ org.apache.falcon.entity.store.ConfigurationStore,\ org.apache.falcon.rerun.service.RetryService,\ org.apache.falcon.rerun.service.LateRunService,\ org.apache.falcon.service.LogCleanupService

Using `Ambari Web UI` > `Services` > `Falcon` > `Configs` > `advanced falcon-startup`:

↳ Add

Name	Value
*.dfs.namenode.kerberos.principal	nn/_HOST@EXAMPLE.COM
*.falcon.enableTLS	false
*.falcon.http.authentication.cookie.domain	EXAMPLE.COM
*.falcon.http.authentication.kerberos.keytab	/etc/security/keytabs/spnego.service.keytab
*.falcon.http.authentication.kerberos.principal	HTTP/_HOST@EXAMPLE.COM
*.falcon.security.authorization.admin.groups	falcon
*.falcon.security.authorization.admin.users	falcon,ambari-qa
*.falcon.security.authorization.enabled	false
*.falcon.security.authorization.provider	org.apache.falcon.security.DefaultAuthorizationProvider
*.falcon.security.authorization.superusergroup	falcon
*.falcon.service.authentication.kerberos.keytab	/etc/security/keytabs/falcon.service.keytab
*.falcon.service.authentication.kerberos.principal	falcon/_HOST@EXAMPLE.COM
*.journal.impl	org.apache.falcon.transaction.SharedFileSystemJournal
prism.application.services	org.apache.falcon.entity.store.ConfigurationStore
prism.configstore.listeners	org.apache.falcon.entity.v0.EntityGraph,\ org.apache.falcon.entity.ColoClusterRelation,\ org.apache.falcon.group.FeedGroupMap

29 Using `Ambari Web` > `Services` > `Service Actions`, re-start all stopped services.

30 The upgrade is now fully functional but not yet finalized. Using the `finalize` command removes the previous version of the NameNode and DataNode storage directories.

After the upgrade is finalized, the system cannot be rolled back. Usually this step is not taken until a thorough testing of the upgrade has been performed.

The upgrade must be finalized before another upgrade can be performed.

Directories used by Hadoop 1 services set in `/etc/hadoop/conf/taskcontroller.cfg` are not automatically deleted after upgrade. Administrators can choose to delete these directories after the upgrade.

To finalize the upgrade, execute the following command once, on the primary NameNode host in your HDP cluster,

```
sudo su -l <HDFS_USER> -c "hdfs dfsadmin -finalizeUpgrade"
```

where `<HDFS_USER>` is the HDFS service user. For example, hdfs.

Upgrading the HDP Stack from 2.0 to 2.2

The HDP Stack is the coordinated set of Hadoop components that you have installed on hosts in your cluster. Your set of Hadoop components and hosts is unique to your cluster. Before upgrading the Stack on your cluster, review all Hadoop services and hosts in your cluster to confirm the location of Hadoop components. For example, use the **Hosts** and **Services** views in Ambari Web, which summarize and list the components installed on each Ambari host, to determine the components installed on each host. For more information about using Ambari to view components in your cluster, see Working with Hosts, and Viewing Components on a Host.

Complete the following procedures to upgrade the Stack from version 2.0 to version 2.2 on your current, Ambari-installed-and-managed cluster.

1. Prepare the 2.0 Stack for Upgrade
2. Upgrade the 2.0 Stack
3. Complete the Upgrade of the 2.0 Stack to 2.2

If you plan to upgrade your existing JDK, do so after upgrading Ambari, before upgrading the Stack. The upgrade steps require that you remove HDP v2.0 components and install HDP v2.2 components. As noted in that section, you should remove and install on each host, only the components on each host that you want to run on the HDP 2.2 stack.

For example, if you want to run Storm or Falcon components on the HDP 2.2 stack, you will install those components and then configure their properties during the upgrade procedure.

In preparation for future HDP 2.2 releases to support rolling upgrades, the HDP RPM package version naming convention has changed to include the HDP 2.2 product version in file and directory names. HDP 2.2 marks the first release where HDP rpms, debs, and directories contain versions in the names to permit side-by-side installations of later HDP releases. To transition between previous releases and HDP 2.2, Hortonworks provides hdp-select, a script that symlinks your directories to `hdp/current` and lets you maintain using the same binary and configuration paths that you were using before.

The following instructions have you remove your old versions of HDP, install hdp-select, and install HDP 2.2 to prepare for rolling upgrade.

Prepare the 2.0 Stack for Upgrade

To prepare for upgrading the HDP Stack, this section describes how to perform the following tasks:

- Disable Security.

If your Stack has Kerberos Security turned on, turn it off before performing the upgrade. On `Ambari Web UI` > `Admin` > `Security` click `Disable Security`. You can re-enable Security after performing the upgrade.

- Checkpoint user metadata and capture the HDFS operational state.
  This step supports rollback and restore of the original state of HDFS data, if necessary.
- Backup Hive and Oozie metastore databases.
  This step supports rollback and restore of the original state of Hive and Oozie data, if necessary.
- Stop all HDP and Ambari services.
- Make sure to finish all current jobs running on the system before upgrading the stack.

Libraries will change during the upgrade. Any jobs remaining active that use the older version libraries will probably fail during the upgrade.

1 Use `Ambari Web` > `Services` > `Service Actions` to stop all services except HDFS and ZooKeeper.

2 Stop any client programs that access HDFS.

Perform steps 3 through 8 on the NameNode host. In a highly-available NameNode configuration, execute the following procedure on the primary NameNode.

To locate the primary NameNode in an Ambari-managed HDP cluster, browse `Ambari Web` > `Services` > `HDFS`. In Summary, click NameNode. `Hosts` > `Summary` displays the host name FQDN.

3 If HDFS is in a non-finalized state from a prior upgrade operation, you must finalize HDFS before upgrading further. Finalizing HDFS will remove all links to the metadata of the prior HDFS version - do this only if you do not want to rollback to that prior HDFS version.

On the NameNode host, as the HDFS user,

```
su -l <HDFS_USER>
hdfs dfsadmin -finalizeUpgrade
```

where `<HDFS_USER>` is the HDFS Service user. For example, hdfs.

4 Check the NameNode directory to ensure that there is no snapshot of any prior HDFS upgrade.

Specifically, using **Ambari Web > HDFS > Configs > NameNode**, examine the `<$dfs.namenode.name.dir>` or the `<$dfs.name.dir>` directory in the NameNode Directories property. Make sure that only a "\current" directory and no "\previous" directory exists on the NameNode host.

5 Create the following logs and other files.

Creating these logs allows you to check the integrity of the file system, post-upgrade.

As the HDFS user,

```
su -l <HDFS_USER>
```

where <HDFS_USER> is the HDFS Service user. For example, hdfs.

1. Run `fsck` with the following flags and send the results to a log. The resulting file contains a complete block map of the file system. You use this log later to confirm the upgrade.

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

2. Optional: Capture the complete namespace of the filesystem.

   The following command does a recursive listing of the root file system:

   ```
   hadoop dfs -ls -R / > dfs-old-lsr-1.log
   ```

3. Create a list of all the DataNodes in the cluster.

   ```
   hdfs dfsadmin -report > dfs-old-report-1.log
   ```

4. Optional: Copy all unrecoverable data stored in HDFS to a local file system or to a backup instance of HDFS.

6. **Save the namespace.**

   You must be the HDFS service user to do this and you must put the cluster in Safe Mode.

   ```
   hdfs dfsadmin -safemode enter
   ```

   ```
   hdfs dfsadmin -saveNamespace
   ```

   

   In a highly-available NameNode configuration, the command `hdfs dfsadmin -saveNamespace` sets a checkpoint in the first NameNode specified in the configuration, in `dfs.ha.namenodes.[nameservice ID]`. You can also use the dfsadmin `-fs` option to specify which NameNode to connect.

   For example, to force a checkpoint in namenode 2:

   ```
   hdfs dfsadmin -fs hdfs://namenode2-hostname:namenode2-port -saveNamespace
   ```

7. Copy the checkpoint files located in <$dfs.name.dir/current> into a backup directory.

Find the directory, using Ambari Web > HDFS > Configs > NameNode > NameNode Directories on your primary NameNode host.

In a highly-available NameNode configuration, the location of the checkpoint depends on where the saveNamespace command is sent, as defined in the preceding step.

8 Store the layoutVersion for the NameNode.

Make a copy of the file at `<dfs.name.dir>/current/VERSION` where `<dfs.name.dir>` is the value of the config parameter `NameNode directories`. This file will be used later to verify that the layout version is upgraded.

9 Stop HDFS.

10 Stop ZooKeeper.

11 Using `Ambari Web` > `Services` > `<service.name>` > `Summary`, review each service and make sure that all services in the cluster are completely stopped.

12 On the Hive Metastore database host, stop the Hive metastore ***service***, if you have not done so already.

Make sure that the Hive metastore ***database*** is running. For more information about Administering the Hive metastore database, see the Hive Metastore Administrator documentation.

13 If you are upgrading Hive and Oozie, back up the Hive and Oozie metastore databases on the Hive and Oozie database host machines, respectively.

Make sure that your Hive database is updated to the minimum recommended version. **If you are using Hive with MySQL, we recommend upgrading your MySQL database version to 5.6.21 before upgrading the HDP Stack to v2.2.** For specific information, see Database Requirements.

1 Optional - Back up the Hive Metastore database.

These instructions are provided for your convenience. Please check your database documentation for the latest back up instructions.

Table 3. 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
Oracle	Connect to the Oracle database using sqlplus export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database ile=input_file.dmp

2 Optional - Back up the Oozie Metastore database.

These instructions are provided for your convenience. Please check your database documentation for the latest back up instructions.

Table 4. 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

14 On the Ambari Server host, stop Ambari Server and confirm that it is stopped.

```
ambari-server stop
ambari-server status
```

15 Stop all Ambari Agents.

At every host in your cluster known to Ambari,

```
ambari-agent stop
```

Upgrade the 2.0 Stack to 2.2

1. Upgrade the HDP repository on all hosts and replace the old repository file with the new file:
   - **For RHEL/CentOS/Oracle Linux 6:**

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/centos6/2.x/GA/2.2.0.0/hdp.repo -O
/etc/yum.repos.d/HDP.repo
```

   - **For SLES 11 SP3:**

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/suse11sp3/2.x/GA/2.2.0.0/hdp.repo -O
/etc/zypp/repos.d/HDP.repo
```

   - **For SLES 11 SP1:**

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/sles11sp1/2.x/GA/2.2.0.0/hdp.repo -O
/etc/zypp/repos.d/HDP.repo
```

   - **For UBUNTU:**

```

wget -nv http://public-repo-
1.hortonworks.com/HDP/ubuntu1/2.x/GA/2.2.0.0/hdp.list -O
/etc/apt/sourceslist.d/HDP.list
```

   - **For RHEL/CentOS/Oracle Linux 5:**

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/centos5/2.x/GA/2.2.0.0/hdp.repo -O
/etc/yum.repos.d/HDP.repo
```

Make sure to download the HDP.repo file under /etc/yum.repos on ALL hosts.

2. Update the Stack version in the Ambari Server database.
   On the Ambari Server host, use the following command to update the Stack version to HDP-2.2:

```
ambari-server upgradestack HDP-2.2
```

3. Back up the files in following directories on the Oozie server host and make sure that all files, including *site.xml files are copied.

```
mkdir oozie-conf-bak
cp -R /etc/oozie/conf/* oozie-conf-bak
```

4. Remove the old oozie directories on all Oozie server and client hosts.
   - `rm -rf /etc/oozie/conf`
   - `rm -rf /usr/lib/oozie/`
   - `rm -rf /var/lib/oozie/`

5 Upgrade the Stack on all Ambari Agent hosts.

For each host, identify the HDP components installed on each host. Use Ambari Web, as described here, to view components on each host in your cluster.

Based on the HDP components installed, tailor the following upgrade commands for each host to upgrade only components residing on that host. For example, if you know that a host has ***no*** HBase service or client packages installed, then you can adapt the command to ***not*** include HBase, as follows:

```
yum install "collectd*" "gccxml*" "pig*" "hadoop*" "sqoop*"
"zookeeper*" "hive*"
```

If you are writing to multiple systems using a script, do not use " " with the run command. You can use " " with pdsh -y.

- For RHEL/CentOS/Oracle Linux:

1 On all hosts, clean the yum repository.

```
yum clean all
```

2 Remove all components that you want to upgrade. At least, WebHCat, HCatlaog, and Oozie components.
This command un-installs the HDP 2.0 component bits. It leaves the user data and metadata, but removes your configurations.

```
yum erase "hadoop*" "webhcat*" "hcatalog*" "oozie*" "pig*" "hdfs*" "sqoop*"
"zookeeper*" "hbase*" "hive*" "phoenix*" "accumulo*" "mahout*" "hue*"
"flume*" "hdp_mon_nagios_addons"
```

3 Remove your old hdp.repo and hdp-utils repo files.

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

4 Install the following components:

```
yum install "hadoop_2_2_0_0_*"  "zookeeper_2_2_0_0_*" "hive_2_2_0_0_*"
"flume_2_2_0_0_*" "phoenix_2_2_0_0_*" "accumulo_2_2_0_0_*" "mahout_2_2_0_0_*"
rpm -e --nodeps hue-shell
yum install hue hue-common hue-beeswax hue-hcatalog hue-pig hue-oozie
```

5 Verify that the components were upgraded.

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

Nothing should appear in the returned list.

- For SLES:

1 On all hosts, clean the zypper repository.

```
zypper clean --all
```

2 Remove WebHCat, HCatalog, and Oozie components.
This command uninstalls the HDP 2.0 component bits. It leaves the user data and metadata, but removes your configurations.

```
zypper remove "hadoop*" "webhcat*" "hcatalog*" "oozie*" "pig*" "hdfs*"
"sqoop*" "zookeeper*" "hbase*" "hive*" "phoenix*" "accumulo*" "mahout*"
"hue*" "flume*" "hdp_mon_nagios_addons"
```

3 Remove your old hdp.repo and hdp-utils repo files.

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

4 Install the following components:

```
zypper install "hadoop\_2_2_0_0_*" "oozie\_2_2_0_0_*" "pig\_2_2_0_0_*"
"sqoop\_2_2_0_0_*" "zookeeper\_2_2_0_0_*" "hbase\_2_2_0_0_*"
"hive\_2_2_0_0_*" "flume\_2_2_0_0_*" "phoenix\_2_2_0_0_*"
"accumulo\_2_2_0_0_*" "mahout\_2_2_0_0_*"
rpm -e --nodeps hue-shell
zypper install hue hue-common hue-beeswax hue-hcatalog hue-pig hue-oozie
```

5 Verify that the components were upgraded.

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

No 2.0 components should appear in the returned list.

6 If components were not upgraded, upgrade them as follows:

```
yast --update hadoop hcatalog hive
```

6 Symlink directories, using `hdp-select`.

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.

Check that the `hdp-select` package installed:

```
rpm -qa | grep hdp-select
```

You should see: `hdp-select-2.2.0.0-2041.el6.noarch`

If not, then run:

```
yum install hdp-select
```

Run `hdp-select` as root, on every node. In `/usr/bin`:

```
hdp-select set all 2.2.0.0-<$version>
```

where `<$version>` is the build number. For the HDP 2.2 release <$version> = 2041.

Check that the `hdp-select` package installed:

```
rpm -qa | grep hdp-select
```

You should see: `hdp-select-2.2.0.0-2041.el6.noarch`

If not, then run:

```
yum install hdp-select
```

Run `hdp-select` as root, on every node. In `/usr/bin`:

```
hdp-select set all 2.2.0.0-<$version>
```

where `<$version>` is the build number. For the HDP 2.2 release <$version> = 2041.

7 On the Hive Metastore database host, stop the Hive Metastore ***service***, if you have not done so already. Make sure that the Hive Metastore ***database*** is running.

8 Upgrade the Hive metastore database schema from v12 to v14, using the following instructions:

- Set java home:

  ```
  export JAVA_HOME=/path/to/java
  ```

- Copy (rewrite) old Hive configurations to new conf dir:

  ```
  cp -R /etc/hive/conf.server/* /etc/hive/conf/
  ```

- Copy the jdbc connector to `/usr/hdp/<$version>/hive/lib`, if it not there, yet.

- `<HIVE_HOME>/bin/schematool -upgradeSchema -dbType<databaseType>`

  where <HIVE_HOME> is the Hive installation directory.

  For example, on the Hive Metastore host:

  /usr/hdp/2.2.0.0-`<$version>`/hive/bin/schematool -upgradeSchema -dbType `<databaseType>`

  where `<$version>` is the 2.2.0 build number and `<databaseType>` is derby, mysql, oracle, or postgres.

Complete the Upgrade of the 2.0 Stack to 2.2

1 Start Ambari Server.

On the Server host,

```
amber-server start
```

2 Start all Ambari Agents.

On each Ambari Agent host,

```
ambari-agent start
```

3 Update the repository Base URLs in the Ambari Server for the HDP 2.2 stack.

Browse to Ambari Web > Admin > Repositories, then set the value of the HDP and HDP-UTILS repository Base URLs. For more information about viewing and editing repository Base URLs, see Viewing Cluster Stack Version and Repository URLs.

For a remote, accessible, public repository, the HDP and HDP-UTILS Base URLs are the same as the baseurl=values in the HDP.repo file downloaded in Upgrade the Stack: Step 1. For a local repository, use the local repository Base URL that you configured for the HDP Stack. For links to download the HDP repository files for your version of the Stack, see HDP Repositories.

4 Using the **Ambari Web UI > Services**, start the ZooKeeper service.

5 At all Datanode and Namenode hosts, copy (rewrite) old hdfs configurations to new conf directory:

```
cp /etc/hadoop/conf.empty/hdfs-site.xml.rpmsave  /etc/hadoop/conf/hdfs-site.xml;

cp /etc/hadoop/conf.empty/hadoop-env.sh.rpmsave /etc/hadoop/conf/hadoop-env.sh;

cp /etc/hadoop/conf.empty/log4j.properties.rpmsave /etc/hadoop/conf/log4j.properties;

cp /etc/hadoop/conf.empty/core-site.xml.rpmsave /etc/hadoop/conf/core-site.xml
```

6 If you are upgrading from an HA NameNode configuration, start all JournalNodes.

On each JournalNode host, run the following command:

```
su -l <HDFS_USER> -c "/usr/hdp/2.2.0.0-<$version>/hadoop/sbin/hadoop-daemon.sh start journalnode"
```

where <HDFS_USER> is the HDFS Service user. For example, hdfs.

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 will fail.

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

On the active NameNode host, as the HDFS user:

```
su -l <HDFS_USER> -c "export HADOOP_LIBEXEC_DIR=/usr/hdp/2.2.0.0-<$version>/hadoop/libexec && /usr/hdp/2.2.0.0-<$version>/hadoop/sbin/hadoop-daemon.sh start namenode -upgrade"
```

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

In a NameNode HA 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.

As the HDFS user:

```
su -l <HDFS_USER> -c "hdfs namenode -bootstrapStandby -force"w
```

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 via Ambari, then start the standby NameNode via Ambari. You can check the status of both NameNodes using the Web UI.

8 Start all DataNodes.

On each DataNode, as the HDFS user,

```
su -l <HDFS_USER> -c "/usr/hdp/2.2.0.0-<$version>/hadoop/sbin/hadoop-
daemon.sh --config /etc/hadoop/conf start datanode"
```

where <HDFS_USER> is the HDFS Service user. For example, hdfs.

The NameNode sends an upgrade command to DataNodes after receiving block reports.

9 Update HDFS Configuration Properties for HDP 2.2

Using Ambari Web UI > Services > HDFS > Configs > core-site.xml:

- Add

Name	Value
hadoop.proxyuser.falcon.groups	users
hadoop.proxyuser.falcon.hosts	*

Using Ambari Web UI > Services > HDFS > Configs > hdfs-site.xml:

- Add

Name	Value
dfs.namenode.startup.delay.block.deletion.sec	3600

- Modify

Name	Value
dfs.datanode.max.transfer.threads	4096

10 Restart HDFS.

1 Open the Ambari Web GUI. If the browser in which Ambari is running has been open throughout the process, clear the browser cache, then refresh the browser.

2 Choose `Ambari Web` > `Services` > `HDFS` > `Service Actions` > `Restart All.`

In a cluster configured for NameNode High Availability, use the following procedure to restart NameNodes. Using the following procedure preserves HA when upgrading the cluster.

1 Using Ambari Web > Services > HDFS, choose Active NameNode.
This shows the host name of the current, active NameNode.
2 Write down (or copy, or remember) the host name of the active NameNode.
You need this host name for step 4.
3 Using Ambari Web > Services > HDFS > Service Actions > choose Stop.
This stops all of the HDFS Components, including both NameNodes.
4 Using Ambari Web > Hosts > choose the host name you noted in Step 2, then start that NameNode component, using Host Actions > Start.
This causes the original, active NameNode to re-assume its role as the active NameNode.
5 Using Ambari Web > Services > HDFS > Service Actions, choose Re-Start All.

3 Choose `Service Actions` > `Run Service Check`. Makes sure the service checks pass.

11 After the DataNodes are started, HDFS exits safe mode. Monitor the status, by running the following command, as the HDFS user:

```
sudo su -l <HDFS_USER> -c "hdfs dfsadmin -safemode get"
```

When HDFS exits safe mode, the following message displays:

```
Safe mode is OFF
```

12 Make sure that the HDFS upgrade was successful.

- Compare the old and new versions of the following log files:
- `dfs-old-fsck-1.log` **versus** `dfs-new-fsck-1.log`.

The files should be identical unless the hadoop fsck reporting format has changed in the new version.

- `dfs-old-lsr-1.log` **versus** `dfs-new-lsr-1.log`.

The files should be identical unless the format of hadoop fs -lsr reporting or the data structures have changed in the new version.

- `dfs-old-report-1.log` **versus** `fs-new-report-1.log`.

Make sure that all DataNodes in the cluster before upgrading are up and running.

13 Update HBase Configuration Properties for HDP 2.2

Using `Ambari Web UI` > `Services` > `HBase` > `Configs` > `hbase-site.xml`:

↳ Add

Name	Value
hbase.hregion.majorcompaction.jitter	0.50

↳ Modify

Name	Value
hbase.hregion.majorcompaction	604800000
hbase.hregion.memstore.block.multiplier	4

↳ Remove

Name	Value
hbase.hstore.flush.retries.number	120

14 Using Ambari Web, navigate to `Services` > `Hive` > `Configs` > `Advanced` and verify that the following properties are set to their default values:

```
Hive (Advanced)
hive.security.authorization.manager=org.apache.hadoop.hive.ql.security.author
ization.StorageBasedAuthorizationProvider
hive.security.metastore.authorization.manager=org.apache.hadoop.hive.ql.secur
ity.authorization.StorageBasedAuthorizationProvider
hive.security.authenticator.manager=org.apache.hadoop.hive.ql.security.ProxyU
serAuthenticator
```

The Security Wizard enables Hive authorization. The default values for these properties changed in Hive-0.12. If you are upgrading Hive from 0.12 to 0.13 in a secure cluster, you should not need to change the values. If upgrading from Hive-older than version 0.12 to Hive-0.12 or greater in a secure cluster, you will need to correct the values.

15 Update Hive Configuration Properties for HDP 2.2

Using `Ambari Web UI` > `Services` > `Hive` > `Configs` > `hive-site.xml`:

↳ Add

Name	Value
hive.cluster.delegation.token.store.zookeeper.connectString	<!-- The ZooKeeper token store connect string. -->

datanucleus.cache.level2.type	none
hive.auto.convert.sortmerge.join.to.mapjoin	false
hive.cbo.enable	true
hive.cli.print.header	false
hive.cluster.delegation.token.store.class	org.apache.hadoop.hive.thrift.ZooKeeperTokenStore
hive.cluster.delegation.token.store.zookeeper.znode	/hive/cluster/delegation
hive.compactor.abortedtxn.threshold	1000
hive.compactor.check.interval	300L
hive.compactor.delta.num.threshold	10
hive.compactor.delta.pct.threshold	0.1f
hive.compactor.initiator.on	false
hive.compactor.worker.threads	0
hive.compactor.worker.timeout	86400L
hive.compute.query.using.stats	true
hive.conf.restricted.list	hive.security.authenticator.manager,hive.security.authorization.manager,hive.users.in.admin.role
hive.convert.join.bucket.mapjoin.tez	false
hive.enforce.sortmergebucketmapjoin	true
hive.exec.compress.intermediate	false
hive.exec.compress.output	false
hive.exec.dynamic.partition	true
hive.exec.dynamic.partition.mode	nonstrict
hive.exec.max.created.files	100000
hive.exec.max.dynamic.partitions	5000
hive.exec.max.dynamic.partitions.pernode	2000
hive.exec.orc.compression.strategy	SPEED
hive.exec.orc.default.compress	ZLIB
hive.exec.orc.default.stripe.size	67108864
hive.exec.parallel	false
hive.exec.parallel.thread.number	8
hive.exec.reducers.bytes.per.reducer	67108864
hive.exec.reducers.max	1009
hive.exec.scratchdir	/tmp/hive
hive.exec.submit.local.task.via.child	true
hive.exec.submitviachild	false
hive.fetch.task.aggr	false
hive.fetch.task.conversion	more
hive.fetch.task.conversion.threshold	1073741824
hive.limit.optimize.enable	true
hive.limit.pushdown.memory.usage	0.04
hive.map.aggr.hash.force.flush.memory.threshold	0.9
hive.map.aggr.hash.min.reduction	0.5
hive.map.aggr.hash.percentmemory	0.5
hive.mapjoin.optimized.hashtable	true
hive.merge.mapfiles	true

hive.merge.mapredfiles	false
hive.merge.orcfile.stripe.level	true
hive.merge.rcfile.block.level	true
hive.merge.size.per.task	256000000
hive.merge.smallfiles.avgsize	16000000
hive.merge.tezfiles	false
hive.metastore.authorization.storage.checks	false
hive.metastore.client.connect.retry.delay	5s
hive.metastore.connect.retries	24
hive.metastore.failure.retries	24
hive.metastore.kerberos.keytab.file	/etc/security/keytabs/hive.service.keytab
hive.metastore.kerberos.principal	hive/_HOST@EXAMPLE.COM
hive.metastore.server.max.threads	100000
hive.optimize.constant.propagation	true
hive.optimize.metadataonly	true
hive.optimize.null.scan	true
hive.optimize.sort.dynamic.partition	false
hive.orc.compute.splits.num.threads	10
hive.orc.splits.include.file.footer	false
hive.prewarm.enabled	false
hive.prewarm.numcontainers	10
hive.security.metastore.authenticator.manager	org.apache.hadoop.hive.ql.security.HadoopDefaultMetastoreAuthenticator
hive.security.metastore.authorization.auth.reads	true
hive.server2.allow.user.substitution	true
hive.server2.authentication.spnego.keytab	HTTP/_HOST@EXAMPLE.COM
hive.server2.authentication.spnego.principal	/etc/security/keytabs/spnego.service.keytab
hive.server2.logging.operation.enabled	true
hive.server2.logging.operation.log.location	${system:java.io.tmpdir}/${system:user.name}/operation_logs
hive.server2.table.type.mapping	CLASSIC
hive.server2.tez.default.queues	default
hive.server2.tez.sessions.per.default.queue	1
hive.server2.thrift.http.path	cliservice
hive.server2.thrift.http.port	10001
hive.server2.thrift.max.worker.threads	500
hive.server2.thrift.sasl.qop	auth
hive.server2.transport.mode	binary
hive.server2.use.SSL	false
hive.smbjoin.cache.rows	10000
hive.stats.autogather	true
hive.stats.dbclass	fs
hive.stats.fetch.column.stats	false

hive.stats.fetch.partition.stats	true
hive.support.concurrency	false
hive.tez.auto.reducer.parallelism	false
hive.tez.cpu.vcores	-1
hive.tez.dynamic.partition.pruning	true
hive.tez.dynamic.partition.pruning.max.data.size	104857600
hive.tez.dynamic.partition.pruning.max.event.size	1048576
hive.tez.input.format	org.apache.hadoop.hive.ql.io.HiveInputFormat
hive.tez.log.level	INFO
hive.tez.max.partition.factor	2.0
hive.tez.min.partition.factor	0.25
hive.tez.smb.number.waves	0.5
hive.txn.manager	org.apache.hadoop.hive.ql.lockmgr.DummyTxnManager
hive.txn.max.open.batch	1000
hive.txn.timeout	300
hive.user.install.directory	/user/
hive.vectorized.execution.reduce.enabled	false
hive.vectorized.groupby.checkinterval	4096
hive.vectorized.groupby.flush.percent	0.1
hive.vectorized.groupby.maxentries	100000
hive.zookeeper.client.port	2181
hive.zookeeper.namespace	hive_zookeeper_namespace
hive.zookeeper.quorum	<!-- List of zookeeper server to talk to -->

- Modify

Name	Value
hive.auto.convert.join.noconditionaltask.size	238026752
hive.metastore.client.socket.timeout	1800s
hive.optimize.reducededuplication.min.reducer	4
hive.security.authorization.manager	org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQLStdConfOnlyAuthorizerFactory
hive.security.metastore.authorization.manager	org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvider,org.apache.hadoop.hive.ql.security.authorization.MetaStoreAuthzAPIAuthorizerEmbedOnly
hive.server2.support.dynamic.service.discovery	true
hive.tez.container.size	682
hive.tez.java.opts	-server -Xmx546m -Djava.net.preferIPv4Stack=true -XX:NewRatio=8 -XX:+UseNUMA -XX:+UseParallelGC -XX:+PrintGCDetails -verbose:gc -XX:+PrintGCTimeStamps

fs.file.impl.disable.cache	true
fs.hdfs.impl.disable.cache	true

16 If YARN is installed in your HDP 2.0 stack, and the Application Timeline Server (ATS) components are **NOT**, then you must create and install ATS service and host components via API by running the following commands on the server that will host the YARN application timeline server in your cluster. Be sure to replace `<your_ATS_component_hostname>` with a host name appropriate for your envrionment.

Ambari does not currently support ATS in a kerberized cluster. If you are upgrading YARN in a kerberized cluster, skip this step.

1 Create the ATS Service Component.

```
curl --user admin:admin -H "X-Requested-By: ambari" -i -X POST
http://localhost:8080/api/v1/clusters/<your_cluster_name>/services/YARN/compo
nents/APP_TIMELINE_SERVER
```

2 Create the ATS Host Component.

```
curl --user admin:admin -H "X-Requested-By: ambari" -i -X POST
http://localhost:8080/api/v1/clusters/<your_cluster_name>/hosts/<your_ATS_com
ponent_hostname>/host_components/APP_TIMELINE_SERVER
```

3 Install the ATS Host Component.

```
curl --user admin:admin -H "X-Requested-By: ambari" -i -X PUT -d '{
"HostRoles": { "state":  "INSTALLED"}}'
http://localhost:8080/api/v1/clusters/<your_cluster_name>/hosts/<your_ATS_com
ponent_hostname>/host_components/APP_TIMELINE_SERVER
```

curl commands use the default `username/password = admin/admin`. To run the curl commands using non-default credentials, modify the --user option to use your Ambari administrator credentials. For example: --user `<ambari_admin_username>`:`<ambari_admin_password>`.

17 Make the following config changes required for Application Timeline Server. Use the Ambari web UI to navigate to the service dashboard and add/modify the following configurations:

```
YARN (Custom yarn-site.xml)
yarn.timeline-service.leveldb-timeline-store.path=/var/log/hadoop-
yarn/timeline
yarn.timeline-service.leveldb-timeline-store.ttl-interval-ms=300000
yarn.timeline-service.store-
class=org.apache.hadoop.yarn.server.timeline.LeveldbTimelineStore
yarn.timeline-service.ttl-enable=true
yarn.timeline-service.ttl-ms=2678400000
yarn.timeline-service.generic-application-history.store-
class=org.apache.hadoop.yarn.server.applicationhistoryservice.NullApplication
HistoryStore
yarn.timeline-
service.webapp.address=<PUT_THE_FQDN_OF_ATS_HOST_NAME_HERE>:8188
yarn.timeline-
service.webapp.https.address=<PUT_THE_FQDN_OF_ATS_HOST_NAME_HERE>:8190
yarn.timeline-service.address=<PUT_THE_FQDN_OF_ATS_HOST_NAME_HERE>:10200


HIVE (hive-site.xml)
hive.execution.engine=mr
hive.exec.failure.hooks=org.apache.hadoop.hive.ql.hooks.ATSHook
hive.exec.post.hooks=org.apache.hadoop.hive.ql.hooks.ATSHook
hive.exec.pre.hooks=org.apache.hadoop.hive.ql.hooks.ATSHook
hive.tez.container.size=<map-container-size>
```

```
      *If mapreduce.map.memory.mb > 2GB then set it equal to
      mapreduce.map.memory. Otherwise, set it equal to
mapreduce.reduce.memory.mb*
hive.tez.java.opts="-server -Xmx" + Math.round(0.8 * map-container-size) + "m
-Djava.net.preferIPv4Stack=true -XX:NewRatio=8 -XX:+UseNUMA -
XX:+UseParallelGC"
```


Use configuration values appropriate for your environment. For example, the value "800" in the preceding example is shown only for illustration purposes.

18 **Prepare MR2 and Yarn for work. Execute hdfs commands on any host.**

- **Create mapreduce dir in hdfs.**

```
su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.2.0.0-
<$version>/mapreduce/"
```

- **Copy new mapreduce.tar.gz to hdfs mapreduce dir.**

```
su -l <HDFS_USER> -c "hdfs dfs -copyFromLocal /usr/hdp/2.2.0.0-
<$version>/hadoop/mapreduce.tar.gz /hdp/apps/2.2.0.0-<$version>/mapreduce/."
```

- **Grant permissions for created mapreduce dir in hdfs.**

```
su -l <HDFS_USER> -c "hdfs dfs -chown -R <HDFS_USER>:<HADOOP_GROUP> /hdp";
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.2.0.0-
<$version>/mapreduce";
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.2.0.0-
<$version>/mapreduce/mapreduce.tar.gz"
```

- Using Ambari Web UI > Service > Mapreduce2 > Configs > Advanced > mapred-site:
- Add

Name	Value
mapreduce.job.emit-timeline-data	false
mapreduce.jobhistory.bind-host	0.0.0.0
mapreduce.reduce.shuffle.fetch.retry.enabled	1
mapreduce.reduce.shuffle.fetch.retry.interval-ms	1000
mapreduce.reduce.shuffle.fetch.retry.timeout-ms	30000

- Modify

Name	Value
mapreduce.admin.map.child.java.opts	-server -XX:NewRatio=8 -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.version}
mapreduce.admin.reduce.child.java.opts	-server -XX:NewRatio=8 -Djava.net.preferIPv4Stack=true -Dhdp.version=${hdp.version}
mapreduce.map.java.opts	-Xmx546m
mapreduce.map.memory.mb	682
mapreduce.reduce.java.opts	-Xmx546m
mapreduce.task.io.sort.mb	273
yarn.app.mapreduce.am.admin-command-opts	-Dhdp.version=${hdp.version}
yarn.app.mapreduce.am.command-opts	-Xmx546m -Dhdp.version=${hdp.version}
yarn.app.mapreduce.am.resource.mb	682
mapreduce.application.framework.path	/hdp/apps/${hdp.version}/mapreduce/mapreduce.tar.gz#mr-framework
mapreduce.application.classpath	$PWD/mr-framework/hadoop/share/hadoop/mapreduce/*:$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
mapreduce.admin.user.env	LD_LIBRARY_PATH=/usr/hdp/${hdp.version}/hadoop/lib/native:/usr/hdp/${hdp.version}/hadoop/lib/native/Linux-amd64-64

- Using Ambari Web UI > Service > Yarn > Configs > Advanced > yarn-site. Add/modify the following property:

Name	Value
hadoop.registry.zk.quorum	<!--List of hostname:port pairs defining the zookeeper quorum binding for the registry-->
hadoop.registry.rm.enabled	false
yarn.client.nodemanager-connect.max-wait-ms	900000
yarn.client.nodemanager-connect.retry-interval-ms	10000
yarn.node-labels.fs-store.retry-policy-spec	2000, 500
yarn.node-labels.fs-store.root-dir	/system/yarn/node-labels
yarn.node-labels.manager-class	org.apache.hadoop.yarn.server.resourcemanager.nodelabels.MemoryRMNodeLabelsManager
yarn.nodemanager.bind-host	0.0.0.0
yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage	90
yarn.nodemanager.disk-health-checker.min-free-space-per-disk-mb	1000
yarn.nodemanager.linux-container-executor.cgroups.hierarchy	hadoop-yarn
yarn.nodemanager.linux-container-executor.cgroups.mount	false
yarn.nodemanager.linux-container-executor.cgroups.strict-resource-usage	false
yarn.nodemanager.linux-container-executor.resources-handler.class	org.apache.hadoop.yarn.server.nodemanager.util.DefaultLCEResourcesHandler
yarn.nodemanager.log-aggregation.debug-enabled	false
yarn.nodemanager.log-aggregation.num-log-files-per-app	30
yarn.nodemanager.log-aggregation.roll-monitoring-interval-seconds	-1
yarn.nodemanager.recovery.dir	/var/log/hadoop-yarn/nodemanager/recovery-state
yarn.nodemanager.recovery.enabled	false
yarn.nodemanager.resource.cpu-vcores	1
yarn.nodemanager.resource.percentage-physical-cpu-limit	100

yarn.resourcemanager.bind-host	0.0.0.0
yarn.resourcemanager.connect.max-wait.ms	900000
yarn.resourcemanager.connect.retry-interval.ms	30000
yarn.resourcemanager.fs.state-store.retry-policy-spec	2000, 500
yarn.resourcemanager.fs.state-store.uri	<enter a "space" as the property value>
yarn.resourcemanager.ha.enabled	false
yarn.resourcemanager.recovery.enabled	false
yarn.resourcemanager.state-store.max-completed-applications	${yarn.resourcemanager.max-completed-applications}
yarn.resourcemanager.store.class	org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore
yarn.resourcemanager.system-metrics-publisher.dispatcher.pool-size	10
yarn.resourcemanager.system-metrics-publisher.enabled	true
yarn.resourcemanager.webapp.delegation-token-auth-filter.enabled	false
yarn.resourcemanager.work-preserving-recovery.enabled	false
yarn.resourcemanager.work-preserving-recovery.scheduling-wait-ms	10000
yarn.resourcemanager.zk-acl	world:anyone:rwcda
yarn.resourcemanager.zk-address	localhost:2181
yarn.resourcemanager.zk-num-retries	1000
yarn.resourcemanager.zk-retry-interval-ms	1000
yarn.resourcemanager.zk-state-store.parent-path	/rmstore
yarn.resourcemanager.zk-timeout-ms	10000
yarn.timeline-service.bind-host	0.0.0.0
yarn.timeline-service.client.max-retries	30
yarn.timeline-service.client.retry-interval-ms	1000
yarn.timeline-service.enabled	true

yarn.timeline-service.http-authentication.simple.anonymous.allowed	true
yarn.timeline-service.http-authentication.type	simple
yarn.timeline-service.leveldb-timeline-store.read-cache-size	104857600
yarn.timeline-service.leveldb-timeline-store.start-time-read-cache-size	10000
yarn.timeline-service.leveldb-timeline-store.start-time-write-cache-size	10000

19 Using Ambari Web > Services > Service Actions, start YARN.

20 Using Ambari Web > Services > Service Actions, start MapReduce2.

21 Using Ambari Web > Services > Service Actions, start HBase and ensure the service check passes.

22 Using Ambari Web > Services > Service Actions, start the Hive service.

23 Upgrade Oozie.

1 Perform the following preparation steps on each Oozie server host:

You must replace your Oozie configuration after upgrading.

1 Copy configurations from **oozie-conf-bak** to the **/etc/oozie/conf** directory on each Oozie server and client.

2 Create `/usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22` directory.

```
mkdir /usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22
```

3 Copy the JDBC jar of your Oozie database to both `/usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22` and `/usr/hdp/2.2.0.0-<$version>/oozie/libtools`.
For example, if you are using MySQL, copy your `mysql-connector-java.jar`.

4 Copy these files to `/usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22` directory

```
cp /usr/lib/hadoop/lib/hadoop-lzo*.jar /usr/hdp/2.2.0.0-
<$version>/oozie/libext-upgrade22;
cp /usr/share/HDP-oozie/ext-2.2.zip /usr/hdp/2.2.0.0-<$version>/oozie/libext-
upgrade22;
cp /usr/share/HDP-oozie/ext-2.2.zip /usr/hdp/2.2.0.0-<$version>/oozie/libext
```

5 Grant read/write access to the Oozie user.

```
chmod -R 777 /usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22
```

2 Upgrade steps:

1 On the Services view, make sure that YARN and MapReduce2 services are running.
2 Make sure that the Oozie service is stopped.
3 In `oozie-env.sh`, comment out `CATALINA_BASE` property, also do the same using Ambari Web UI in `Services` > `Oozie` > `Configs` > `Advanced oozie-env`.
4 Upgrade Oozie.

At the Oozie server host, as the Oozie service user:

```
sudo su -l <OOZIE_USER> -c"/usr/hdp/2.2.0.0-<$version>/oozie/bin/ooziedb.sh
upgrade -run"
```

where `<OOZIE_USER>` is the Oozie service user. For example, oozie.

Make sure that the output contains the string "Oozie DB has been upgraded to Oozie version `<OOZIE_Build_Version>`.

5 Prepare the Oozie WAR file.

The Oozie server must be **not** running for this step. If you get the message "ERROR: Stop Oozie first", it means the script still thinks it's running. Check, and if needed, remove the process id (pid) file indicated in the output.

At the Oozie server, as the Oozie user

```
sudo su -l <OOZIE_USER> -c "/usr/hdp/2.2.0.0-<$version>/oozie/bin/oozie-
setup.sh prepare-war -d /usr/hdp/2.2.0.0-<$version>/oozie/libext-upgrade22"
```

where `<OOZIE_USER>` is the Oozie service user. For example, oozie.

Make sure that the output contains the string "New Oozie WAR file added".

6 Using **Ambari Web**, choose **Services > Oozie > Configs, expand oozie-log4j**, then add the following property:

```
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.

7 Using Ambari Web, choose `Services` > `Oozie` > `Configs`, expand **Advanced oozie-site**, then edit the following properties:

A In `oozie.service.coord.push.check.requeue.interval`, ***replace*** the existing property value with the following one:

```
30000
```

B In `oozie.service.SchemaService.wf.ext.schemas`, ***append*** (using copy/paste) to the existing property value the following string, if is it is not already present:

```
shell-action-0.1.xsd,shell-action-0.2.xsd,shell-action-0.3.xsd,email-action-
0.1.xsd,email-action-0.2.xsd,hive-action-0.2.xsd,hive-action-0.3.xsd,hive-
action-0.4.xsd,hive-action-0.5.xsd,sqoop-action-0.2.xsd,sqoop-action-
0.3.xsd,sqoop-action-0.4.xsd,ssh-action-0.1.xsd,ssh-action-0.2.xsd,distcp-
action-0.1.xsd,distcp-action-0.2.xsd,oozie-sla-0.1.xsd,oozie-sla-0.2.xsd
```

If you have customized schemas, append this string to your custom schema name string.

Do not overwrite custom schemas.

If you have no customized schemas, you can replace the existing string with the following one:

shell-action-0.1.xsd,email-action-0.1.xsd,hive-action-0.2.xsd,sqoop-action-0.2.xsd,ssh-action-0.1.xsd,distcp-action-0.1.xsd,shell-action-0.2.xsd,oozie-sla-0.1.xsd,oozie-sla-0.2.xsd,hive-action-0.3.xsd

C In `oozie.service.URIHandlerService.uri.handlers`, ***append*** to the existing property value the following string, if is it is not already present:

```
org.apache.oozie.dependency.FSURIHandler,org.apache.oozie.dependency.HCatURIH
andler
```

D In `oozie.services`, make sure all the following properties are present:

```
org.apache.oozie.service.SchedulerService,
org.apache.oozie.service.InstrumentationService,
org.apache.oozie.service.MemoryLocksService,
org.apache.oozie.service.UUIDService,
org.apache.oozie.service.ELService,
org.apache.oozie.service.AuthorizationService,
org.apache.oozie.service.UserGroupInformationService,
org.apache.oozie.service.HadoopAccessorService,
org.apache.oozie.service.JobsConcurrencyService,
org.apache.oozie.service.URIHandlerService,
org.apache.oozie.service.DagXLogInfoService,
org.apache.oozie.service.SchemaService,
org.apache.oozie.service.LiteWorkflowAppService,
org.apache.oozie.service.JPAService,
org.apache.oozie.service.StoreService,
org.apache.oozie.service.CoordinatorStoreService,
org.apache.oozie.service.SLAStoreService,
org.apache.oozie.service.DBLiteWorkflowStoreService,
org.apache.oozie.service.CallbackService,
org.apache.oozie.service.ActionService,
org.apache.oozie.service.ShareLibService,
org.apache.oozie.service.CallableQueueService,
org.apache.oozie.service.ActionCheckerService,
org.apache.oozie.service.RecoveryService,
org.apache.oozie.service.PurgeService,
org.apache.oozie.service.CoordinatorEngineService,
org.apache.oozie.service.BundleEngineService,
org.apache.oozie.service.DagEngineService,
org.apache.oozie.service.CoordMaterializeTriggerService,
org.apache.oozie.service.StatusTransitService,
org.apache.oozie.service.PauseTransitService,
org.apache.oozie.service.GroupsService,
org.apache.oozie.service.ProxyUserService,
org.apache.oozie.service.XLogStreamingService,
org.apache.oozie.service.JvmPauseMonitorService
```

E Add the `oozie.services.coord.check.maximum.frequency` property with the following property value: `false`

If you set this property to true, Oozie rejects any coordinators with a frequency faster than 5 minutes. It is not recommended to disable this check or submit coordinators with frequencies faster than 5 minutes: doing so can cause unintended behavior and additional system stress.

F Add the `oozie.service.AuthorizationService.security.enabled` property with the following property value: `false`

Specifies whether security (user name/admin role) is enabled or not. If disabled any user can manage Oozie system and manage any job.

G Add the `oozie.service.HadoopAccessorService.kerberos.enabled` property with the following property value: `false`

Indicates if Oozie is configured to use Kerberos.

H Add the `oozie.authentication.simple.anonymous.allowed` property with the following property value: `true`

Indicates if anonymous requests are allowed. This setting is meaningful only when using 'simple' authentication.

- In `oozie.services.ext`, ***append*** to the existing property value the following string, if is it is not already present:

`org.apache.oozie.service.PartitionDependencyManagerService,org.apache.oozie.service.HCatAccessorService`

- Update Oozie Configuration Properties for HDP 2.2

Using `Ambari Web UI > Services > Oozie > Configs > oozie-site.xml`:

- Add

Name	Value
oozie.authentication.simple.anonymous.allowed	true
oozie.service.coord.check.maximum.frequency	false
oozie.service.ELService.ext.functions.coord-action-create	now=org.apache.oozie.extensions.OozieELExtensions#ph2_now, today=org.apache.oozie.extensions.OozieELExtensions#ph2_today, yesterday=org.apache.oozie.extensions.OozieELExtensions#ph2_yesterday, currentMonth=org.apache.oozie.extensions.OozieELExtensions#ph2_currentMonth, lastMonth=org.apache.oozie.extensions.OozieELExtensions#ph2_lastMonth, currentYear=org.apache.oozie.extensions.OozieELExtensions#ph2_currentYear, lastYear=org.apache.oozie.extensions.OozieELExtensions#ph2_lastYear, latest=org.apache.oozie.coord.CoordELFunctions#ph2_coord_latest_echo, future=org.apache.oozie.coord.CoordELFunctions#ph2_coord_future_echo, formatTime=org.apache.oozie.coord.CoordELFunctions#ph2_coord_formatTime, user=org.apache.oozie.coord.CoordELFunctions#coord_user

oozie.service.ELService.ext.functions.coord-action-create-inst	now=org.apache.oozie.extensions.OozieELExtensions#ph2_now_inst, today=org.apache.oozie.extensions.OozieELExtensions#ph2_today_inst, yesterday=org.apache.oozie.extensions.OozieELExtensions#ph2_yesterday_inst, currentMonth=org.apache.oozie.extensions.OozieELExtensions#ph2_currentMonth_inst, lastMonth=org.apache.oozie.extensions.OozieELExtensions#ph2_lastMonth_inst, currentYear=org.apache.oozie.extensions.OozieELExtensions#ph2_currentYear_inst, lastYear=org.apache.oozie.extensions.OozieELExtensions#ph2_lastYear_inst, latest=org.apache.oozie.coord.CoordELFunctions#ph2_coord_latest_echo, future=org.apache.oozie.coord.CoordELFunctions#ph2_coord_future_echo, formatTime=org.apache.oozie.coord.CoordELFunctions#ph2_coord_formatTime, user=org.apache.oozie.coord.CoordELFunctions#coord_user
oozie.service.ELService.ext.functions.coord-action-start	now=org.apache.oozie.extensions.OozieELExtensions#ph2_now, today=org.apache.oozie.extensions.OozieELExtensions#ph2_today, yesterday=org.apache.oozie.extensions.OozieELExtensions#ph2_yesterday, currentMonth=org.apache.oozie.extensions.OozieELExtensions#ph2_currentMonth, lastMonth=org.apache.oozie.extensions.OozieELExtensions#ph2_lastMonth, currentYear=org.apache.oozie.extensions.OozieELExtensions#ph2_currentYear, lastYear=org.apache.oozie.extensions.OozieELExtensions#ph2_lastYear, latest=org.apache.oozie.coord.CoordELFunctions#ph3_coord_latest, future=org.apache.oozie.coord.CoordELFunctions#ph3_coord_future, dataIn=org.apache.oozie.extensions.OozieELExtensions#ph3_dataIn, instanceTime=org.apache.oozie.coord.CoordELFunctions#ph3_coord_nominalTime, dateOffset=org.apache.oozie.coord.CoordELFunctions#ph3_coord_dateOffset, formatTime=org.apache.oozie.coord.CoordELFunctions#ph3_coord_formatTime, user=org.apache.oozie.coord.CoordELFunctions#coord_user

oozie.service.ELService.ext.functions.coord-job-submit-data	now=org.apache.oozie.extensions.OozieELExtensions#ph1_now_echo, today=org.apache.oozie.extensions.OozieELExtensions#ph1_today_echo, yesterday=org.apache.oozie.extensions.OozieELExtensions#ph1_yesterday_echo, currentMonth=org.apache.oozie.extensions.OozieELExtensions#ph1_currentMonth_echo, lastMonth=org.apache.oozie.extensions.OozieELExtensions#ph1_lastMonth_echo, currentYear=org.apache.oozie.extensions.OozieELExtensions#ph1_currentYear_echo, lastYear=org.apache.oozie.extensions.OozieELExtensions#ph1_lastYear_echo, dataIn=org.apache.oozie.extensions.OozieELExtensions#ph1_dataIn_echo, instanceTime=org.apache.oozie.coord.CoordELFunctions#ph1_coord_nominalTime_echo_wrap, formatTime=org.apache.oozie.coord.CoordELFunctions#ph1_coord_formatTime_echo, dateOffset=org.apache.oozie.coord.CoordELFunctions#ph1_coord_dateOffset_echo, user=org.apache.oozie.coord.CoordELFunctions#coord_user
oozie.service.ELService.ext.functions.coord-job-submit-instances	now=org.apache.oozie.extensions.OozieELExtensions#ph1_now_echo, today=org.apache.oozie.extensions.OozieELExtensions#ph1_today_echo, yesterday=org.apache.oozie.extensions.OozieELExtensions#ph1_yesterday_echo, currentMonth=org.apache.oozie.extensions.OozieELExtensions#ph1_currentMonth_echo, lastMonth=org.apache.oozie.extensions.OozieELExtensions#ph1_lastMonth_echo, currentYear=org.apache.oozie.extensions.OozieELExtensions#ph1_currentYear_echo, lastYear=org.apache.oozie.extensions.OozieELExtensions#ph1_lastYear_echo, formatTime=org.apache.oozie.coord.CoordELFunctions#ph1_coord_formatTime_echo, latest=org.apache.oozie.coord.CoordELFunctions#ph2_coord_latest_echo, future=org.apache.oozie.coord.CoordELFunctions#ph2_coord_future_echo
oozie.service.ELService.ext.functions.coord-sla-create	instanceTime=org.apache.oozie.coord.CoordELFunctions#ph2_coord_nominalTime, user=org.apache.oozie.coord.CoordELFunctions#coord_user
oozie.service.ELService.ext.functions.coord-sla-submit	instanceTime=org.apache.oozie.coord.CoordELFunctions#ph1_coord_nominalTime_echo_fixed, user=org.apache.oozie.coord.CoordELFunctions#coord_user

oozie.service.HadoopAccessorService .kerberos.enabled	false
oozie.service.HadoopAccessorService .supported.filesystems	*

↳ Modify

Name	Value
oozie.service.SchemaService.wf.ext.schemas	shell-action-0.1.xsd,shell-action-0.2.xsd,shell-action-0.3.xsd,email-action-0.1.xsd,email-action-0.2.xsd,hive-action-0.2.xsd,hive-action-0.3.xsd,hive-action-0.4.xsd,hive-action-0.5.xsd,sqoop-action-0.2.xsd,sqoop-action-0.3.xsd,sqoop-action-0.4.xsd,ssh-action-0.1.xsd,ssh-action-0.2.xsd,distcp-action-0.1.xsd,distcp-action-0.2.xsd,oozie-sla-0.1.xsd,oozie-sla-0.2.xsd
oozie.services.ext	org.apache.oozie.service.JMSAccessorService,org.apache.oozie.service.PartitionDependencyManagerService,org.apache.oozie.service.HCatAccessorService

C After modifying all properties on the Oozie Configs page, choose **Save** to update `oozie.site.xml`, using the updated configurations.

4 Replace the content of `/usr/oozie/share` in HDFS.
On the Oozie server host:

1 Extract the Oozie sharelib into a `tmp` folder.

```
mkdir -p /tmp/oozie_tmp;
cp /usr/hdp/2.2.0.0-<$version>/oozie/oozie-sharelib.tar.gz /tmp/oozie_tmp;
cd /tmp/oozie_tmp;
tar xzvf oozie-sharelib.tar.gz;
```

2 Back up the `/user/oozie/share` folder in HDFS and then delete it. If you have any custom files in this folder, back them up separately and then add them to the `/share` folder after updating it.

```
mkdir /tmp/oozie_tmp/oozie_share_backup;
chmod 777 /tmp/oozie_tmp/oozie_share_backup;

su -l <HDFS_USER> -c "hdfs dfs -copyToLocal /user/oozie/share
/tmp/oozie_tmp/oozie_share_backup";
su -l <HDFS_USER> -c "hdfs dfs -rm -r /user/oozie/share";
```

where `<HDFS_USER>` is the HDFS service user. For example, hdfs.

3 Add the latest share libs that you extracted in step 1. After you have added the files, modify ownership and acl.

```
su -l <HDFS_USER> -c "hdfs dfs -copyFromLocal /tmp/oozie_tmp/share
/user/oozie/.";
su -l <HDFS_USER> -c "hdfs dfs -chown -R <OOZIE_USER>:<HADOOP_GROUP>
/user/oozie";
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 755 /user/oozie";
```

where `<HDFS_USER>` is the HDFS service user. For example, hdfs.

4 Use the `Ambari Web UI` > `Services` view to start the Oozie service. Make sure that ServiceCheck passes for Oozie.

24 Update WebHCat.

A Modify the `webhcat-site` config type.

Using **Ambari Web**, navigate to `Services` > `WebHCat` and modify the following configuration:

Action	Property Name	Property Value
Modify	templeton.storage.class	org.apache.hive.hcatalog.templeton.tool.ZooKeeperStorage

B Expand `Advanced` > `webhcat-site.xml`.

Check if property `templeton.port` exists. If not, then add it using the Custom webhcat-site panel. The default value for templeton.port = 50111.

C On each WebHCat host, update the Pig and Hive tar bundles, by updating the following files:

- `/apps/webhcat/pig.tar.gz`
- `/apps/webhcat/hive.tar.gz`

Find these files only on a host where WebHCat is installed.

For example, to update a *.tar.gz file:

1 Move the file to a local directory.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -copyToLocal
/apps/webhcat/*.tar.gz <local_backup_dir>"
```

2 Remove the old file.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -rm
/apps/webhcat/*.tar.gz"
```

3 Copy the new file.

```
su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal
/usr/hdp/2.2.0.0-<$version>/hive/hive.tar.gz /apps/webhcat/"; su -l
<HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal
/usr/hdp/2.2.0.0-<$version>/pig/pig.tar.gz /apps/webhcat/";
```

where `<HCAT_USER>` is the HCatalog service user. For example, hcat.

D On each WebHCat host, update `/app/webhcat/hadoop-streaming.jar` file.

1 Move the file to a local directory.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -copyToLocal
/apps/webhcat/hadoop-streaming*.jar <local_backup_dir>"
```

2 Remove the old file.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -rm
/apps/webhcat/hadoop-streaming*.jar"
```

3 Copy the new hadoop-streaming.jar file.

```
su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal
/usr/hdp/2.2.0.0-<$version>/hadoop-mapreduce/hadoop-streaming*.jar
/apps/webhcat"
```

where <HCAT_USER> is the HCatalog service user. For example, hcat.

25 Prepare Tez for work.
Add the Tez service to your cluster using the Ambari Web UI, if Tez was not installed earlier.

Configure Tez.

```
cd /var/lib/ambari-server/resources/scripts/;
./configs.sh set localhost <your-cluster-name> cluster-env
"tez_tar_source" "/usr/hdp/current/tez-client/lib/tez.tar.gz";
./configs.sh set localhost <your-cluster-name> cluster-env
"tez_tar_destination_folder" "hdfs:///hdp/apps/{{ hdp_stack_version
}}/tez/"
```

If you use Tez as the Hive execution engine, and if the variable hive.server2.enabled.doAs is set to true, you must create a scratch directory on the NameNode host for the username that will run the HiveServer2 service. For example, use the following commands:

```
sudo su -c "hdfs -makedir /tmp/hive- <username> "
sudo su -c "hdfs -chmod 777 /tmp/hive- <username> "
```

where <username> is the name of the user that runs the HiveServer2 service.

26 Using the **Ambari Web UI> Services > Hive**, start the Hive service.

27 If you use Tez as the Hive execution engine, and if the variable `hive.server2.enabled.doAs` is set to `true`, you must create a scratch directory on the NameNode host for the username that will run the HiveServer2 service. For example, use the following commands:

```
sudo su -c "hdfs -makedir /tmp/hive-<username>"

sudo su -c "hdfs -chmod 777 /tmp/hive-<username>"
```

where <username> is the name of the user that runs the HiveServer2 service.

28 Using **Ambari Web > Services**, re-start the remaining services.

29 The upgrade is now fully functional but not yet finalized. Using the `finalize` command removes the previous version of the NameNode and DataNode storage directories.

After the upgrade is finalized, the system cannot be rolled back. Usually this step is not taken until a thorough testing of the upgrade has been performed.

The upgrade must be finalized before another upgrade can be performed.

Directories used by Hadoop 1 services set in /etc/hadoop/conf/taskcontroller.cfg are not automatically deleted after upgrade. Administrators can choose to delete these directories after the upgrade.

To finalize the upgrade, execute the following command once, on the primary NameNode host in your HDP cluster:

```
sudo su -l <HDFS_USER> -c "hdfs dfsadmin -finalizeUpgrade"
```

Upgrading the HDP Stack from 1.3 to 2.2

The Stack is the coordinated set of Hadoop components that you have installed. Use the following instructions to upgrade a current, Ambari-installed and managed instance of a version 1.3 Stack to a version 2.2 Stack. This procedure causes the upgraded stack to be managed by Ambari.

If your Stack has Kerberos Security turned on, you should turn it off before performing the upgrade. On `Ambari Web UI` > `Admin` > `Security`, click `Disable Security`. You can re-enable Security after performing the upgrade.

If you are upgrading from any other 1.x version of the stack, you must upgrade to 1.3 or later before you can upgrade to 2.2. Upgrades from previous 1.x versions are not supported.

In preparation for future HDP 2.2 releases to support rolling upgrades, the HDP RPM package version naming convention has changed to include the HDP 2.2 product version in file and directory names. HDP 2.2 marks the first release where HDP rpms, debs, and directories contain versions in the names to permit side-by-side installations of later HDP releases. To transition between previous releases and HDP 2.2, Hortonworks provides hdp-select, a script that symlinks your directories to `hdp/current` and lets you maintain using the same binary and configuration paths that you were using before. The following instructions have you remove your old versions of HDP, install hdp-select, and install HDP 2.2 to prepare for rolling upgrade.

Preparing the 1.3 Stack for the Upgrade to 2.2

To prepare for upgrading the HDP Stack, this section describes how to perform the following tasks:

- Checkpoint user metadata and capture the HDFS operational state.
  This step supports rollback and restore of the original state of HDFS data, if necessary.
- Backup Hive and Oozie metastore databases.
  This step supports rollback and restore of the original state of Hive and Oozie data, if necessary.
- Stop all HDP and Ambari services.

Perform steps 1 through 8 on the NameNode host. In a highly-available NameNode configuration, you should execute the following procedure on the primary NameNode.

To locate the primary NameNode in an Ambari-managed HDP cluster, browse `Ambari Web` > `Services` > `HDFS`. In Summary, click NameNode. `Hosts` > `Summary` displays the host name FQDN.

1. Stop all services except HDFS and ZooKeeper. Also stop any client programs that access HDFS.
2. If HDFS is in a non-finalized state from a prior upgrade operation, you must finalize HDFS before upgrading further. Finalizing HDFS will remove all links to the metadata of the prior HDFS version - do this only if you do not want to rollback to that prior HDFS version.

   For example, as the HDFS user:
   ```
   sudo -u <HDFS_USER> hadoop dfsadmin -finalizeUpgrade
   ```

   You can check the namenode directory to ensure that there is no snapshot of any prior HDFS upgrade. Specifically, examine the $dfs.namenode.name.dir (or $dfs.name.dir) on the NameNode. Make sure that only a 'current', not a 'previous' directory exists.
3. Create the following logs and other files.

   Creating these logs lets you to check the integrity of the file system after upgrading.

   1. Run `fsck` with the following flags and send the results to a log. The resulting file contains a complete block map of the file system. You use this log later to confirm the upgrade.

      ```
      sudo -u <HDFS_USER> hadoop fsck / -files -blocks -locations >
      dfs-old-fsck-1.log
      ```
   2. Optional: Capture the complete namespace of the filesystem. (The following command does a recursive listing of the root file system.)

      ```
      sudo -u <HDFS_USER> hadoop dfs -lsr / > dfs-old-lsr-1.log
      ```
   3. Create a list of all the DataNodes in the cluster.

      ```
      sudo -u <HDFS_USER> hadoop dfsadmin -report > dfs-old-report-
      1.log
      ```
   4. Optional: copy all or unrecoverable only data stored in HDFS to a local file system or to a backup instance of HDFS.
4. **Save the namespace**. You must be the HDFS service user to do this and you must put the cluster in Safe Mode.

This is a critical step. If you do not do this step before you do the upgrade, the NameNode will not start afterwards.

As the HDFS user:

```
hadoop dfsadmin -safemode enter
hadoop dfsadmin -saveNamespace
```

In a HA NameNode configuration, the command `hdfs dfsadmin -saveNamespace` does checkpoint in the first NameNode specified in the configuration, in `dfs.ha.namenodes.[nameservice ID]`. You can also use the dfsadmin `-fs` option to specify which NameNode to connect. For example, to force a checkpoint in namenode 2:

```
hdfs dfsadmin -fs hdfs://namenode2-hostname:namenode2-port -saveNamespace
```

5 Copy the following checkpoint files into a backup directory. You can find the directory by using the **Services View** in the Ambari Web UI. Select `HDFS` > `Configs`. In the Namenode section, look up the property **NameNode Directories** on your NameNode host.

```
<dfs.name.dir>/current
```

In a HA NameNode configuration, the location of the checkpoint depends on where the saveNamespace command is sent, as defined in the preceding step.

6 Store the layoutVersion for the NameNode. Make a copy of the file at `<dfs.name.dir>/current/VERSION` where `<dfs.name.dir>` is the value of the config parameter `NameNode directories`. This file will be used later to verify that the layout version is upgraded.

7 Stop HDFS. Make sure all services in the cluster are completely stopped.

8 If you are upgrading Hive and Oozie, back up the Hive database and the Oozie database on the Hive database host and Oozie database host machines, respectively.

Make sure that your Hive database is updated to the minimum recommended version. **If you are using Hive with MySQL, we recommend upgrading your MySQL database version to 5.6.21 before upgrading the HDP Stack to v2.2.** For specific information, see Database Requirements.

a Optional - Backup the Hive Metastore database.

These instructions are provided for your convenience. Please check your database documentation for the latest back up instructions.

Table 5. 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
Oracle	Connect to the Oracle database using sqlplus export the database: exp username/password@database full=yes file=output_file.dmp	Import the database: imp username/password@database ile=input_file.dmp

b Optional - Backup the Oozie Metastore database.

These instructions are provided for your convenience. Please check your database documentation for the latest back up instructions.

Table 6. 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

9 On every host in your cluster known to Ambari, stop all Ambari Agents.

```
ambari-agent stop
```

Upgrading the 1.3 Stack to 2.2

This stack upgrade involves removing the HDP 1.x version of MapReduce and replacing it with the HDP 2.x YARN and MapReduce2 components. This process is somewhat long and complex. To help you, a Python script is provided to automate some of the upgrade steps.

Prepare the 1.3 Stack for Upgrade to 2.2

1 Make sure that you completed the system preparation procedure; most importantly, save the namespace.

2 Stage the upgrade script:

1 Create an "Upgrade Folder", for example `/work/upgrade_hdp_2`, on a host that can communicate with Ambari Server. The Ambari Server host would be a suitable candidate.

2 Copy the upgrade script to the Upgrade Folder. The script is available here: `/var/lib/ambari-server/resources/scripts/UpgradeHelper_HDP2.py` on the Ambari Server host.

3 Make sure that Python is available on the host and that the version is 2.6 or higher:

```
python --version
```

For RHEL/Centos/Oracle Linux 5, you **must** use Python 2.6.

3 Start the Ambari Server only if it is stopped. On the Ambari Server host:

```
ambari-server status
```

If status is "stopped", then:

```
ambari-server start
```

4 Back up current configuration settings and the component host mappings from MapReduce:

1 Go to the Upgrade Folder.

2 Execute the `backup-configs` action:

```
python UpgradeHelper_HDP2.py --hostname <HOSTNAME> --user
<USERNAME> --password <PASSWORD> --clustername <CLUSTERNAME>
backup-configs
```

Where

- `<HOSTNAME>` is the name of the Ambari Server host
- `<USERNAME>` is the admin user for Ambari Server
- `<PASSWORD>` is the password for the admin user
- `<CLUSTERNAME>` is the name of the cluster

This step produces a set of files named TYPE_TAG, where TYPE is the configuration type and TAG is the tag. These files contain copies of the various configuration settings for the current (pre-upgrade) cluster. You can use these files as a reference later.

3 Execute the `save-mr-mapping` action:

```
python UpgradeHelper_HDP2.py --hostname <HOSTNAME> --user
<USERNAME> --password <PASSWORD> --clustername <CLUSTERNAME> save-
mr-mapping
```

This step produces a file named `mr_mapping` that stores the host level mapping of MapReduce components such as MapReduce JobTracker/TaskTracker/Client.

5 Delete all the MapReduce server components installed on the cluster.

1. If you are not already there, go to the Upgrade Folder.
2. Execute the `delete-mr` action.

```
python UpgradeHelper_HDP2.py --hostname <HOSTNAME> --user
<USERNAME> --password <PASSWORD> --clustername <CLUSTERNAME>
delete-mr
```

Optionally, execute the delete script with the -n option to view, verify, and validate API calls, if necessary.

Running the delete script with the -n option exposes API calls but does not remove installed components. Use the -n option for validation purposes only.

3. The script asks you to confirm that you have executed the `save-mr-mapping` action and that you have a file named `mr_mapping` in the Upgrade Folder.

6. On the Ambari Server host, stop Ambari Server and confirm that it is stopped.

```
ambari-server stop
ambari-server status
```

Upgrade the 1.3 Stack to 2.2

1. Stop the Ambari Server only if it is started . On the Ambari Server host:

```
ambari-server status
```

If status is "started", then:

```
ambari-server stop
```

2. Make sure that the old version of MapReduce deleted successfully.
3. Update the stack version in the Ambari Server database.
Use the command appropriate for a remote, or local repository, as follows:

```
ambari-server upgradestack HDP-2.2
```

4. Upgrade the HDP repository on all hosts and replace the old repo file with the new file:

The file you download is named `hdp.repo`. To function properly in the system, it must be named `HDP.repo`. Once you have completed the "mv" of the new repo file to the repos.d folder, make sure there is no file named `hdp.repo` anywhere in your repos.d folder.

- For RHEL/CentOS/Oracle Linux 6:

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/centos6/2.x/GA/2.2.0.0/hdp.repo -O
/etc/yum.repos.d/HDP.repo
```

- For SLES 11:

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/centos5/2.x/GA/2.2.0.0/hdp.repo -O
/etc/yum.repos.d/HDP.repo
```

- For RHEL/CentOS/Oracle Linux 5: (DEPRECATED)

```
wget -nv http://public-repo-1.hortonworks.com/HDP-
LABS/Projects/Champlain-Preview/2.2.0.0-9/centos5/hdp.repo -O
/etc/yum.repos.d/HDP.repo
```

5 Back up the files in following directories on the Oozie server host and make sure that all files, including *site.xml files are copied.

```
mkdir oozie-conf-bak
cp -R /etc/oozie/conf/* oozie-conf-bak
```

6 Remove the old **/oozie** directories on all Oozie server and client hosts

- `rm -rf /etc/oozie/conf`
- `rm -rf /usr/lib/oozie/`
- `rm -rf /var/lib/oozie/`

7 Upgrade the stack on all Agent hosts. Skip any components your installation does not use:

- For RHEL/CentOS/Oracle Linux:

1 On all hosts, clean the yum repository.

```
yum clean all
```

2 Remove remaining MapReduce, and WebHCat, HCatalog, and Oozie components on all hosts.
This command uninstalls these HDP 1.3 component bits. It leaves the user data and metadata, but removes your configurations.

```
yum erase hadoop-pipes hadoop-sbin hadoop-native "webhcat*" "hcatalog*"
"oozie*" "hadoop*" "hadoop-libhdfs*" "hbase*" "hcatalog*" "hive*" "oozie*"
"oozie-client*" "pig*" "sqoop*" "webhcat-tar-hive*" "webhcat-tar-pig*"
"zookeeper*"
```

3 Remove your old hdp.repo and hdp-utils repo files.

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

4 Install the following components:

```
yum install "hadoop_2_2_0_0_*" "oozie_2_2_0_0_*" "pig_2_2_0_0_*"
"sqoop_2_2_0_0_*" "zookeeper_2_2_0_0_*" "hbase_2_2_0_0_*" "hive_2_2_0_0_*"
```

5 Verify that the components were upgraded:

```
yum list installed | grep HDP-$old-stack-version-number
```

No 1.3 components should appear in the returned list.

- For SLES:

1 On all hosts, clean the zypper repository.

```
zypper clean --all
```

2 Remove remaining MapReduce, and WebHCat, HCatalog, and Oozie components on all hosts.
This command uninstalls these HDP 1.3 component bits. It leaves the user data and metadata, but removes your configurations.

```
zypper remove hadoop-pipes hadoop-sbin hadoop-native webhcat\* hcatalog\*
oozie\*
```

3 Remove your old hdp.repo and hdp-utils repo files.

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

4 Install the following components:

```
zypper install "collectd*" "epel-release*" "gccxml*" "pig*" "hadoop*"
"sqoop*" "zookeeper*" "hbase*" "hive*" hdp_mon_nagios_addons

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

zypper install oozie oozie-client
```

5 Verify that the components were upgraded.

```
rpm -qa | grep hadoop, rpm -qa | grep hive and rpm -qa | grep hcatalog
```

No 1.3 components should appear in the returned list.

6 If components were not upgraded, upgrade them as follows:

```
yast --update hadoop hcatalog hive
```

8 Symlink directories, using `hdp-select`.

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.

Check that the `hdp-select` package installed:

```
rpm -qa | grep hdp-select
```

You should see: `hdp-select-2.2.0.0-2041.el6.noarch`

If not, then run:

```
yum install hdp-select
```

Run `hdp-select` as root, on your NameNode(s) and all your DataNodes. In `/usr/bin`:

```
hdp-select set all 2.2.0.0-<$version>
```

where `<$version>` is the build number. For the HDP 2.2 release <$version> = 2041.

9 On the Hive Metastore database host, stop the Hive Metastore ***service***, if you have not done so already. Make sure that the Hive Metastore ***database*** is running.

10 Upgrade Hive v11 to v14 and upgrade the Hive metastore database schema, using the following instructions:

- Set java home:

  ```
  export JAVA_HOME=/path/to/java
  ```

- Copy old hive configurations to new conf dir:

  ```
  cp -R /etc/hive/conf.server/* /etc/hive/conf/
  ```

- `<HIVE_HOME> /bin/schematool -upgradeSchemaFrom 0.11.0 -dbType <databaseType>`
  where `<HIVE_HOME>` is the Hive installation directory.

  For example, on the Hive Metastore host:
  `/usr/hdp/2.2.0.0-<$version>/hive/bin/schematool -upgradeSchemaFrom 0.11.0 -dbType <databaseType>`
  where `<$version>` is the 2.2.0 build number and `<databaseType>` is derby, mysql, oracle, or postgres.

Complete Upgrade of the 1.3 Stack to 2.2

1 Start Ambari Server and Ambari Agents.

On the Server host:

```
ambari-server start
```

On all of the Agent hosts:

```
ambari-agent start
```

2 Update the repository Base URLs in Ambari Server for the HDP-2.2 stack. Browse to `Ambari Web` > `Admin` > `Repositories` and set the value of the HDP and HDP-UTILS repository Base URLs. For more information about viewing and editing repository Base URLs, see Viewing Cluster Stack Version and Repository URLs.

For a remote, accessible, public repository, the HDP and HDP-UTILS Base URLs are the same as the baseurl=values in the HDP.repo file downloaded in Upgrade the Stack: Step 1 For a local repository, use the local repository Base URL that you configured for the HDP Stack. For links to download the HDP repository files for your version of the Stack, see HDP Stack Repositories.

3 Add YARN and MapReduce2 services:

- If you are not already there, go to the Upgrade Folder.
- Execute the add-yarn-mr2 action:

  ```
  python UpgradeHelper_HDP2.py --hostname $HOSTNAME --user $USERNAME --password $PASSWORD --clustername $CLUSTERNAME add-yarn-mr2
  ```

If desired, you can use the -n option to see the API calls as they are being made so that you can verify them.

4 Update the respective configurations:

- If you are not already there, go to the Upgrade Folder.
- Execute the update-configs action:

```
python UpgradeHelper_HDP2.py --hostname $HOSTNAME --user
$USERNAME --password $PASSWORD --clustername $CLUSTERNAME update-
configs
```

5 Install the YARN and MapReduce2 services:

- If you are not already there, go to the Upgrade Folder.
- Execute the install-yarn-mr2 action:

```
python UpgradeHelper_HDP2.py --hostname $HOSTNAME --user
$USERNAME --password $PASSWORD --clustername $CLUSTERNAME
install-yarn-mr2
```

6 Using the Ambari Web UI, add the Tez service if if it has not been installed already. For more information about adding a service, see Adding a Service.

7 Using the Ambari Web UI, add any new services that you want to run on the HDP 2.2 stack. You must add a Service before editing configuration properties necessary to complete the upgrade.

8 Using the Ambari Web Services view, start the ZooKeeper service.

9 If you are upgrading from an HA NameNode configuration, start all JournalNodes. On each JournalNode host, run the following command as the HDFS user:

```
su -l <HDFS_USER> -c "/usr/hdp/2.2.0.0-<$version>/hadoop/sbin/hadoop-
daemon.sh start journalnode"
```

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 will fail.

10 Because the file system version has now changed you must start the NameNode manually. On the NameNode host, as the HDFS User:

```
su -l -c "export HADOOP_LIBEXEC_DIR=/usr/hdp/2.2.0.0-
<$version>/hadoop/libexec && /usr/hdp/2.2.0.0-
<$version>/hadoop/sbin/hadoop-daemon.sh start namenode -upgrade"
```

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

In a NameNode HA 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 standbyNameNode by running the NameNode with the '-bootstrapStandby' flag. **Do NOT** start this standby NameNode with the '-upgrade' flag.

```
su -l <HDFS_USER> -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 via Ambari, then start the standby NameNode via Ambari. You can check the status of both NameNodes using the Web UI.

11 Start all DataNodes.

```
su -l <HDFS_USER> -c "/usr/hdp/2.2.0.0-<$version>/hadoop/sbin/hadoop-
daemon.sh --config /etc/hadoop/conf start datanode"
```

The NameNode will send an upgrade command to DataNodes after receiving block reports.

12 Prepare the NameNode to work with Ambari:

1. Open the Ambari Web GUI. If it has been open throughout the process, do a hard reset on your browser to force a reload.
2. On the Services view, click **HDFS** to open the HDFS service.
3. Click **View Host** to open the NameNode host details page.
4. Use the drop-down menu to stop the NameNode.
5. On the Services view, restart the HDFS service. Make sure it passes the Service Check. It is now under Ambari's control.

In a cluster configured for NameNode High Availability, use the following procedure to restart NameNodes. Using the following procedure preserves HA when upgrading the cluster.

1. Using Ambari Web > Services > HDFS, choose Active NameNode.
   This shows the host name of the current, active NameNode.
2. Write down (or copy, or remember) the host name of the active NameNode.
   You need this host name for step 4.
3. Using Ambari Web > Services > HDFS > Service Actions > choose Stop.
   This stops all of the HDFS Components, including both NameNodes.
4. Using Ambari Web > Hosts > choose the host name you noted in Step 2, then start that NameNode component, using Host Actions > Start.
   This causes the original, active NameNode to re-assume its role as the active NameNode.
5. Using Ambari Web > Services > HDFS > Service Actions, choose Re-Start All.

13 After the DataNodes are started, HDFS exits safemode. To monitor the status, run the following command:

```
sudo su -l <HDFS_USER> -c "hdfs dfsadmin -safemode get"
```

Depending on the size of your system, a response may not display for up to 10 minutes. When HDFS exits safemode, the following message displays:

```
Safe mode is OFF
```

14 Make sure that the HDFS upgrade was successful. Execute step 3 in Preparing for the Upgrade to create new versions of the logs and reports. Substitute " `new` " for " `old` " in the file names as necessary.

- Compare the old and new versions of the following:
- `dfs-old-fsck-1.log` **versus** `dfs-new-fsck-1.log`.

The files should be identical unless the hadoop fsck reporting format has changed in the new version.

- `dfs-old-lsr-1.log` **versus** `dfs-new-lsr-1.log`.

The files should be identical unless the format of hadoop fs -lsr reporting or the data structures have changed in the new version.

- `dfs-old-report-1.log` **versus** `fs-new-report-1.log`.

Make sure all DataNodes previously belonging to the cluster are up and running.

15 Update the configuration properties required for HDFS. Using Ambari Web, navigate to **Services > HDFS > Configs** and add/modify the following configurations:

- Change the io.compression.codecs property to:

```
org.apache.hadoop.io.compress.GzipCodec,com.hadoop.compression.lzo.LzoCodec,org.apache.hadoop.io
```

- Add to core-site.xml, the following property:

```
<name>io.compression.codec.lzo.class</name>
<value>com.hadoop.compression.lzo.LzoCodec</value>
```

16. Using Ambari Web > Services > Service Actions, start YARN.
17. Using Ambari Web > Services > Service Actions, start MapReduce2.
18. Upgrade HBase.
    1. Make sure that all HBase components - RegionServers and HBase Master - are stopped.
    2. Using Ambari Web > Services, start the ZooKeeper service. Wait until the ZK service is up and running.
    3. On the HBase Master host, make these configuration changes:
    1. In HBASE_CONFDIR/hbase-site.xml, set the property dfs.client.read.shortcircuit to false.
    2. In the configuration file, find the value of the hbase.tmp.dir property and make sure that the directory exists and is readable and writeable for the HBase service user and group.

```
chown -R <HBASE_USER>:<HADOOP_GROUP><HBASE.TMP.DIR>
```

    3. Go to the Upgrade Folder and check in the saved global configuration file named global_<$TAG> for the value of the property hbase_pid_dir and hbase_log_dir. Make sure that the directories are readable and writeable for the HBase service user and group.

```
chown -R <HBASE_USER>:<HADOOP_GROUP><hbase_pid_dir>
chown -R <HBASE_USER>:<HADOOP_GROUP><hbase_log_dir>
```

Do this on **every** host where a RegionServer is installed as well as on the HBase Master host.

    4. Check for HFiles in V1 format. HBase 0.96.0 discontinues support for HFileV1. Before the actual upgrade, run the following command to check if there are HFiles in V1 format: hbase upgrade -check HFileV1 was a common format prior to HBase 0.94. You may see output similar to:

```
Tables Processed:

hdfs://localhost:41020/myHBase/.META.
hdfs://localhost:41020/myHBase/usertable
hdfs://localhost:41020/myHBase/TestTable
hdfs://localhost:41020/myHBase/t

Count of HFileV1: 2
HFileV1:
hdfs://localhost:41020/myHBase/usertable/fa02dac1f38d03577bd0f7e666f12812/fam
ily/249450144068442524
hdfs://localhost:41020/myHBase/usertable/ecdd3eaee2d2fcf8184ac025555bb2af/fam
ily/249450144068442512

Count of corrupted files: 1
Corrupted Files:
hdfs://localhost:41020/myHBase/usertable/fa02dac1f38d03577bd0f7e666f12812/fam
ily/1
Count of Regions with HFileV1: 2
Regions to Major Compact:
hdfs://localhost:41020/myHBase/usertable/fa02dac1f38d03577bd0f7e666f12812
hdfs://localhost:41020/myHBase/usertable/ecdd3eaee2d2fcf8184ac025555bb2af
```

When you run the upgrade check, if "Count of HFileV1" returns any files, start the hbase shell to use major compaction for regions that have HFileV1 format. For example in the sample output above, you must compact the `fa02dac1f38d03577bd0f7e666f12812` and `ecdd3eaee2d2fcf8184ac025555bb2af` regions.

5. Upgrade HBase. As the HBase service user:

```
su -l  <HBASE_USER>  -c "hbase upgrade -execute"
```

Make sure that the output contains the string "Successfully completed Znode upgrade".

6. Use the Services view to start the HBase service. Make sure that Service Check passes.

19. Upgrade Oozie.

1. Perform the following preparation steps on each oozie server host:

You must replace your Oozie configuration after upgrading.

1. Copy **/etc/oozie/conf** from the template to the **/conf** directory on each Oozie server and client.
2. Create `/usr/lib/oozie/libext-upgrade22` directory.

```
mkdir /usr/lib/oozie/libext-upgrade22
```

3. Copy the JDBC jar of your Oozie database to both `/usr/lib/oozie/libext-upgrade22` and `/usr/lib/oozie/libtools`.

For example, if you are using MySQL, copy your `mysql-connector-java.jar`.

4 Copy these files to `/usr/lib/oozie/libext-upgrade22` directory

```
cp /usr/lib/hadoop/lib/hadoop-lzo*.jar /usr/lib/oozie/libext-upgrade22

cp /usr/share/HDP-oozie/ext-2.2.zip /usr/lib/oozie/libext-upgrade22
```

5 Grant read/write access to the Oozie user.

```
chmod -R 777 /usr/lib/oozie/libext-upgrade22
```

2 Upgrade steps:

1 On the Services view, make sure YARN and MapReduce2 are running.
2 Make sure that the Oozie service is stopped.
3 Upgrade Oozie. On the Oozie server host, as the Oozie service user:

```
su -l <OOZIE_USER> -c "/usr/lib/oozie/bin/ooziedb.sh upgrade -run"
```

Make sure that the output contains the string "Oozie DB has been upgraded to Oozie version `<OOZIE_build_version>`.

4 Prepare the Oozie WAR file.

The Oozie server must be **not** running for this step. If the message "ERROR: Stop Oozie first" displays, find and remove the process id (pid) file listed in the output. This prevent the script from "seeing" the (old) Oozie server process.

As the root user:

```
sudo su -l <OOZIE_USER> -c "/usr/lib/oozie/bin/oozie-setup.sh prepare-war -d
/usr/lib/oozie/libext-upgrade22"
```

Make sure that the output contains the string "New Oozie WAR file added".

5 Using Ambari Web UI `Services` > `Oozie` > `Configs`, edit the following configuration properties:

1 Add the following configuration properties in `oozie-site.xml`.

Action	Property Name	Property Value
Add	oozie.service.URIHandlerService.uri.handlers	org.apache.oozie.dependency.FSURIHandler,org.apache.oozie.dependency.HCatURIHandler
Add	oozie.service.coord.push.check.requeue.interval	30000

Add	oozie.services	org.apache.oozie.service.SchedulerService, org.apache.oozie.service.InstrumentationService, org.apache.oozie.service.CallableQueueService, org.apache.oozie.service.UUIDService, org.apache.oozie.service.ELService, org.apache.oozie.service.AuthorizationService, org.apache.oozie.service.UserGroupInformationService, org.apache.oozie.service.HadoopAccessorService, org.apache.oozie.service.URIHandlerService, org.apache.oozie.service.MemoryLocksService, org.apache.oozie.service.DagXLogInfoService, org.apache.oozie.service.SchemaService, org.apache.oozie.service.LiteWorkflowAppService, org.apache.oozie.service.JPAService, org.apache.oozie.service.StoreService, org.apache.oozie.service.CoordinatorStoreService, org.apache.oozie.service.SLAStoreService, org.apache.oozie.service.DBLiteWorkflowStoreService, org.apache.oozie.service.CallbackService, org.apache.oozie.service.ActionService, org.apache.oozie.service.ActionCheckerService, org.apache.oozie.service.RecoveryService, org.apache.oozie.service.PurgeService, org.apache.oozie.service.CoordinatorEngineService, org.apache.oozie.service.BundleEngineService, org.apache.oozie.service.DagEngineService, org.apache.oozie.service.CoordMaterializeTriggerService, org.apache.oozie.service.StatusTransitService, org.apache.oozie.service.PauseTransitService, org.apache.oozie.service.GroupsService, org.apache.oozie.service.ProxyUserService,org.apache.oozie.service.XLogStreamingService,org.apache.oozie.service.JobsConcurrencyService
Add	oozie.services.ext	org.apache.oozie.service.PartitionDependencyManagerService,org.apache.oozie.service.HCatAccessorService
Add	oozie.service.SchemaService.wf.ext.schemas	shell-action-0.1.xsd,shell-action-0.2.xsd,shell-action-0.3.xsd,email-action-0.1.xsd,email-action-0.2.xsd,hive-action-0.2.xsd,hive-action-0.3.xsd,hive-action-0.4.xsd,hive-action-0.5.xsd,sqoop-action-0.2.xsd,sqoop-actio
Add	oozie.service.coord.check.maximum.frequency	false
Add	oozie.service.AuthorizationService.security.enabled	false
Add	oozie.service.HadoopAccessorService.kerberos.enabled	false
Add	oozie.authentication.simple.anonymous.allowed	true
Add	log4j.appender.oozie.layout.ConversionPattern	%d{ISO8601} %5p %c{1}:%L - SERVER[${oozie.instance.id}] %m%n

Table 7. Oozie-site.xml - Properties to Add

Do not delete existing values from each property values list. IF you have customized schema or property values, make sure the customized values appear on Configs. Values in each property value list must be separated by commas, no spaces.

2. After modifying all properties on the Oozie Configs page, choose **Save** to update `oozie.site.xml`, using the modified configurations.
3. Replace the content of `/usr/oozie/share` in HDFS. On the Oozie server host:
1. Extract the Oozie sharelib into a `tmp` folder.

```
mkdir -p /tmp/oozie_tmp
cp /usr/lib/oozie/oozie-sharelib.tar.gz /tmp/oozie_tmp
cd /tmp/oozie_tmp
tar xzvf oozie-sharelib.tar.gz
```

2. Back up the `/user/oozie/share` folder in HDFS and then delete it. If you have any custom files in this folder back them up separately and then add them back after the share folder is updated.

```
mkdir /tmp/oozie_tmp/oozie_share_backup
chmod 777 /tmp/oozie_tmp/oozie_share_backup
```

As the Oozie user,

```
su -l <HDFS_USER> -c "hdfs dfs -copyToLocal /user/oozie/share
/tmp/oozie_tmp/oozie_share_backup"
su -l <HDFS_USER> -c "hdfs dfs -rm -r /user/oozie/share"
```

3. Add the latest share libs that you extracted in step 1. After you have added the files, modify ownership and acl.

```
su -l <HDFS_USER> -c "hdfs dfs -copyFromLocal /tmp/oozie_tmp/share
/user/oozie/."
su -l <HDFS_USER> -c "hdfs dfs -chown -R <OOZIE_USER>:<HADOOP_GROUP>
/user/oozie"
su -l <HDFS_USER> -c "hdfs dfs -chmod -R 755 /user/oozie"
```

4. Use the Services view to start the Oozie service. Make sure that ServiceCheck passes for Oozie.

20. Update WebHCat.

1. Modify the `webhcat-site` config type.

Using the Ambari web UI, navigate to `Services` > `WebHCat` and modify the following configuration:

Action	Property Name	Property Value
Modify	templeton.storage.class	org.apache.hive.hcatalog.templeton.tool.ZooKeeperStorage

Table 8. WebHCat Properties to Modify

2. On each WebHCat host, update the Pig and Hive tar bundles, by updating the following files:

- `/apps/webhcat/pig.tar.gz`
- `/apps/webhcat/hive.tar.gz`

You will find these files only on a host where webhcat is installed.

For example, to update a *.tar.gz file:

1. Move the file to a local directory. As the WebHCat user,

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -copyToLocal
/apps/webhcat/*.tar.gz $<local_backup_dir>"
```

2. Remove the old file.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -rm
/apps/webhcat/*.tar.gz"
```

3. Copy the new file.

```
su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal
/usr/share/HDP-webhcat/*.tar.gz /apps/webhcat/"
```

3. On each WebHCat host, update `/app/webhcat/hadoop-streaming.jar` file.

1. Move the file to a local directory.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -copyToLocal
/apps/webhcat/hadoop-streaming*.jar $<local_backup_dir>"
```

2. Remove the old file.

```
su -l <HCAT_USER> -c "hadoop --config /etc/hadoop/conf fs -rm
/apps/webhcat/hadoop-streaming*.jar"
```

3. Copy the new hadoop-streaming.jar file.

```
su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal
/usr/lib/hadoop-mapreduce/hadoop-streaming*.jar /apps/webhcat"
```

21. Make sure Ganglia no longer attempts to monitor JobTracker.

1. Make sure Ganglia is stopped.
2. Log into the host where JobTracker was installed (and where ResourceManager is installed after the upgrade).
3. Backup the folder `/etc/ganglia/hdp/HDPJobTracker`.
4. Remove the folder `/etc/ganglia/hdp/HDPJobTracker`.
5. Remove the folder `$ganglia_runtime_dir/HDPJobTracker`.

For the value of `$ganglia_runtime_dir`, in the Upgrade Folder, check the saved global configuration file `global_<$TAG>`.

22. Use the Services view to start the remaining services back up.

23 The upgrade is now fully functional but not yet finalized. Using the `finalize` command removes the previous version of the NameNode and DataNode storage directories.

After the upgrade is finalized, the system cannot be rolled back. Usually this step is not taken until a thorough testing of the upgrade has been performed.

The upgrade must be finalized before another upgrade can be performed.

Directories used by Hadoop 1 services set in /etc/hadoop/conf/taskcontroller.cfg are not automatically deleted after upgrade. Administrators can choose to delete these directories after the upgrade.

To finalize the upgrade, execute the following command once, on the primary NamaNode host in your HDP cluster, as the HDFS User:
`sudo su -l <HDFS_USER> -c "hadoop dfsadmin -finalizeUpgrade"` where `<HDFS_USER>` is the HDFS Service user (by default, `hdfs`).

Upgrading host server operating systems in an Ambari-managed Hadoop System

Ambari requires specific versions of the files for components that it uses. There are three steps you should take to make sure that these versions continue to be available:

- Disable automatic OS updates
- Do not update any HDP components such as MySQL, Ganglia, etc.
- If you must perform an OS update, do a manual kernel update only.

Upgrading an older Ambari Server version to 1.2.5

This 12-step, manual procedure upgrades an Ambari Server from an older, 1.x version to version 1.2.5. Upgrading the Ambari Server version does not change the underlying Hadoop Stack version.

You must know the location of the Nagios server for Step 9. Use the **Services View > Summary** panel to locate the host on which the Nagios server is running.

1 Stop the Ambari Server and all Ambari Agents.

- On the Ambari Server host:

```
ambari-server stop
```

- On each Ambari Agent host:

```
ambari-agent stop
```

2 Get the new Ambari bits.

Using wget, fetch the repository file, then replace the old repository file with the new repository file on every host.

Check your current directory before you download the new repository file to make sure that no previous versions of the file exist. If a previous version exists, the new downloaded file will be saved with a numeric extension such as ambari.repo.1. Make sure that the version you copy is the new version.

1 Fetch the new repository file:

- For RHEL/CentOS 5/Oracle Linux 5:

```
wget http://public-repo-
1.hortonworks.com/ambari/centos5/1.x/updates/1.2.5.17/ambari.repo
```

- For RHEL/CentOS 6/Oracle Linux 6:

```
wget http://public-repo-
1.hortonworks.com/ambari/centos6/1.x/updates/1.2.5.17/ambari.repo
```

- For SLES 11:

```
wget http://public-repo-
1.hortonworks.com/ambari/suse11/1.x/updates/1.2.5.17/ambari.repo
```

2 Replace the old repository file with the new repository file.

- For RHEL/CentOS 5/Oracle Linux 5:

```
cp ambari.repo /etc/yum.repos.d/ambari.repo
```

- For RHEL/CentOS 6/Oracle Linux 6:

```
cp ambari.repo /etc/yum.repos.d/ambari.repo
```

- For SLES 11:

```
cp ambari.repo /etc/zypp/repos.d/ambari.repo
```

If your cluster does not have access to the Internet, set up a local repository with this data before you continue. See Configure the Local Repositories for more information.

3 Upgrade Ambari Server.

On the Ambari Server host:

- For RHEL/CentOS/Oracle Linux:

```
yum clean all
yum upgrade ambari-server-1.2.5.17 ambari-log4j-1.2.5.17
```

- For SLES:

```
zypper clean
zypper up ambari-server-1.2.5.17 ambari-log4j-1.2.5.17
```

4 Check for upgrade success.

- As the process runs, the console should produce output similar, although not identical, to the following text:

```
Setting up Upgrade Process
Resolving Dependencies
--> Running transaction check
---> Package ambari-agent.x86_64 0:1.2.2.3-1 will be updated
---> Package ambari-agent.x86_64 0:1.2.2.4-1 will be updated ...
---> Package ambari-agent.x86_64 0:1.2.2.5-1 will be an update ...
```

After the process is complete, check each host to make sure the new 1.2.4 files have been installed.

```
rpm -qa | grep ambari
```

- If the upgrade fails, the console displays output similar to the following text:

```
Setting up Upgrade Process
No Packages marked for Update
```

5 On the Ambari Server host, check for a folder named `/etc/ambari-server/conf.save`. If such a folder exists, rename it, using the following command:

```
mv /etc/ambari-server/conf.save /etc/ambari-server/conf
```

6 Upgrade the Ambari Server schema.

On the Ambari Server host:

```
ambari-server upgrade
```

7 Upgrade the Ambari Agent on all hosts.

On each Ambari Agent host:

- For RHEL/CentOS/Oracle Linux

```
yum upgrade ambari-agent ambari-log4j
```

- For SLES

```
zypper up ambari-agent ambari-log4j
```

If a warning such as the following, "There are some running programs that use files deleted by recent upgrade" appears, ignore it.

8 On each Agent host, check for a folder named `/etc/ambari-agent/conf.save`. If such a folder exists, rename it, using the following command:

```
mv /etc/ambari-agent/conf.save /etc/ambari-agent/conf
```

9 Upgrade the Nagios and Ganglia add-ons package and restart.

On the Nagios/Ganglia hosts:

- For RHEL/CentOS/Oracle Linux:

```
yum upgrade hdp_mon_nagios_addons hdp_mon_ganglia_addons
service httpd restart
```

- For SLES:

```
zypper up hdp_mon_nagios_addons hdp_mon_ganglia_addons
service apache2 restart
```

10 Start the Ambari Server and all Ambari Agents.

- On the Ambari Server host:

```
ambari-server start
```

- On each Ambari Agent host:

```
ambari-agent start
```

11 Open **Ambari Web**. Point your browser to `http://<your.ambari.server>:8080`

Refresh your browser so that it loads the new version of the code. Hold the Shift key down while clicking the refresh button on the browser. If you have problems, clear your browser cache manually, then restart Ambari Server. Use the Ambari Admin name and password you have set up to log in.

12 Re-start the Ganglia, Nagios, and MapReduce services.

In Ambari Web:

1 Go to Services View, then choose each service.

2 Stop, then re-start each service, using the Management Header.

Troubleshooting an Ambari Server 1.x Upgrade

If upgrading Ambari Server 1.x fails, use the instructions in one of the following sections to fix the failed upgrade.

- Upgrade Failure with PostgreSQL
- Upgrade Failure with Oracle
- Upgrade Failure from a Local Repository

Upgrade Failure with PostgreSQL

If you installed Ambari server with a PostgreSQL database and upgrading Ambari Server using a remote or public repository failed, use the following steps to fix the upgrade.

1 Upgrade the database schema.

```
/var/lib/ambari-server/resources/upgrade/ddl/AmbariRCA-DDL-Postgres-
UPGRADE.sql
```

2 Check database consistency.

```
/var/lib/ambari-server/resources/upgrade/ddl/Ambari-DDL-Postgres-
UPGRADE-1.3.0.Check.sql
```
with the following parameter:
```
dbname = ambari
```

3 If you find an inconsistency, fix it using:

```
/var/lib/ambari-server/resources/upgrade/ddl/Ambari-DDL-Postgres-
UPGRADE-1.3.0.Fix.sql
```
with the following parameter:
```
dbname = ambari
```

Upgrade Failure with Oracle

If you installed Ambari server with an Oracle database and upgrading Ambari Server using a remote or public repository failed, run the following script to upgrade the database schema.

```
/var/lib/ambari-server/resources/upgrade/ddl/AmbariRCA-DDL-Oracle-UPGRADE.sql
```

Upgrade Failure from a Local Repository

If you install and upgrade Ambari server using a local repository and upgrading Ambari Server using your local repository failed, find information necessary to fix the upgrade in the following locations:

1 Check for local repository version customized in `/var/lib/ambari-server/resources/stacks/HDPLocal`.

2 Check for the repository version used when creating the cluster, in `repos/repoinfo.xml`.

3 If local repository version is NOT the same as the NON-LOCAL repository version: Note the os, version, repoid and baseurl, found in `repos/repoinfo.xml`.

Then, choose the fix appropriate for your database version.

- PostgreSQL
- Oracle

Ambari Server with PostgreSQL

To fix a local Ambari/PostgreSQL upgrade:

1 Repair metadata information.

```
/var/lib/ambari-server/resources/upgrade/dml/Ambari-DML-Postgres-
INSERT_METAINFO.sql
```
with the following parameters:
```
dbname = ambari
metainfo_key = repo:/HDP/<version>/<os>/<repoid>:baseurl
metainfo_value = <baseurl>
```

2 Run the fix script.

```
/var/lib/ambari-server/resources/upgrade/dml/Ambari-DML-Postgres-
UPGRADE_STACK.sql
```

with the following parameters:

```
dbname = ambari
```

Ambari Server with Oracle

To fix a local Ambari/PostgreSQL upgrade:

1. Repair metadata information.

```
/var/lib/ambari-server/resources/upgrade/dml/Ambari-DML-Oracle-
INSERT_METAINFO.sql
```

with the following parameters:

```
argument #1 = repo:/HDP/<version>/<os>/<repoid>:baseurl
argument #2 = <baseurl>
```

2. Run the fix script.

```
/var/lib/ambari-server/resources/upgrade/dml/Ambari-DML-Oracle-
UPGRADE_STACK.sql
```