Upgrading Ambari

Ambari 2.0

Hortonworks

Hortonworks Data Platform

Mar 29, 2015

Hortonworks Data Platform : Upgrading Ambari

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

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

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

Table of Contents

Ambari and the HDP Stack being managed by Ambari can be upgraded independently. This guide provides information on:

- Upgrading to Ambari 2.0
- Planning for Ambari Alerts and Metrics in Ambari 2.0
- Upgrading Ambari with Kerberos-Enabled Cluster
- Upgrade HDP Stack from HDP 2.1 to 2.2
- Upgrade HDP Stack from HDP 2.0 to 2.2
- Automated HDP Stack Upgrade: HDP 2.2.0 to 2.2.4
- Manual HDP Stack Upgrade: HDP 2.2.0 to 2.2.4

Ambari 2.0 does not include support for managing HDP 1.3 Stack. For more information, see the Stack Compatibility Matrix. If you are using Ambari to manage an HDP 1.3 Stack, prior to upgrading to Ambari 2.0, **you must first upgrade your Stack to HDP 2.0 or later**. For more information about upgrading HDP 1.3 Stack to HDP 2.0 or later, see the Ambari 1.7.0 upgrade instructions. Then, return to this guide and perform your upgrade to Ambari 2.0.

Upgrading to Ambari 2.0

Use this procedure to upgrade Ambari 1.4.1 through 1.7.0 to Ambari 2.0.0. If your current Ambari version is 1.4.1 or below, you must upgrade the Ambari Server version to 1.7.0 before upgrading to version 2.0.0. Upgrading Ambari version does not change the underlying HDP Stack being managed by Ambari.

Before Upgrading Ambari to 2.0.0, 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.
- You ***must*** know the location of the Ganglia server before you begin the upgrade process.
- You ***must*** backup the Ambari Server database.
- You ***must*** make a safe copy of the Ambari Server configuration file found at `/etc/ambari-server/conf/ambari.properties`.
- Plan to ***remove Nagios and Ganglia from your cluster and replace with Ambari Alerts and Metrics***. For more information, see Planning for Ambari Alerts and Metrics in Ambari 2.0.
- If you have a Kerberos-enabled cluster, you ***must*** review Upgrading Ambari with Kerberos-Enabled Cluster and be prepared to perform post-upgrade steps required.
- 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 HDP 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

To begin the upgrade:

1. Stop the Nagios and Ganglia services. In `Ambari Web`:
   - Browse to `Services` and select the Nagios service.
   - Use `Service Actions` to stop the Nagios service.
   - Wait for the Nagios service to stop.
   - Browse to `Services` and select the Ganglia service.
   - Use `Service Actions` to stop the Ganglia service.
   - 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 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/centos6/2.x/updates/2.0.0/ambari.repo -O
/etc/yum.repos.d/ambari.repo
```

For SLES 11:

```
wget -nv http://public-repo-
1.hortonworks.com/ambari/suse11/2.x/updates/2.0.0/ambari.repo -O
/etc/zypp/repos.d/ambari.repo
```

For Ubuntu 12:

```
wget -nv http://public-repo-
1.hortonworks.com/ambari/ubuntu12/2.x/updates/2.0.0/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/centos5/2.x/updates/2.0.0/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 Using a Local Repository for more information.

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.

5. Upgrade Ambari Server. On 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
   ```

When performing upgrade on SLES, you will see a message "There is an update candidate for 'ambari-server', but it is from different vendor. Use 'zypper install ambari-server-2.0.0-101.noarch' to install this candidate". You will need to to use yast to update the package, as follows:

1. From the command line run: > yast.

   ```
   > yast
   ```

   You will see command line UI for YaST program.
2. Choose Software > Software Management, then click enter button.
3. In the Search Phrase field, enter ambari-server, then click the enter button.
4. On the right side you will see the search result ambari-server 2.0.0 . Click Actions, choose Update, then click the enter button.
5. Go to Accept, and click enter.

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-log4j.noarch 0:1.7.0.169-1  will be updated
   ...
    ---> Package ambari-log4j.noarch 0:2.0.0.1129-1 will be an
   update ...
    ---> Package ambari-server.noarch 0:1.7.0-169  will be updated
   ...
    ---> Package ambari-log4j.noarch 0:2.0.0.1129  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 0:2.0.0.111-1 ambari-server.noarch
0:2.0.0-111 Complete!
```

Confirm there is only one ambari-server*.jar file in `/usr/lib/ambari-server`. If there is more than one JAR file with name ambari-server*.jar, move all JARs except ambari-server-2.0.0.*.jar to /tmp before proceeding with upgrade.

7. On the Ambari Server host: If ambari-agent is also installed on this host, first run "yum upgrade ambari-agent" (or equivalent in other OS'es) Now, upgrade the server database schema by running,

   ```
   ambari-server upgrade
   ```

8. Upgrade the Ambari Agent on each host. On 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".

When performing upgrade on SLES, you will see a message "There is an update candidate for 'ambari-server', but it is from different vendor. Use 'zypper install ambari-server-2.0.0-101.noarch' to install this candidate". You will need to to use yast to update the package, as follows:

1. From the command line run: > yast.

   ```
   > yast
   ```

   You will see command line UI for YaST program.
2. Choose Software > Software Management, then click enter button.
3. In the Search Phrase field, enter ambari-server, then click the enter button.
4. On the right side you will see the search result ambari-server 2.0.0 . Click Actions, choose Update, then click the enter button.

Go to Accept, and click enter.

For Ubuntu:

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

9. After the upgrade process completes, check each host to make sure the new 2.0.0 files have been installed:

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

10. Start the Ambari Server. On the Ambari Server host:

    ```
    ambari-server start
    ```

11. Start the Ambari Agents on all hosts. On each Ambari Agent host:

    ```
    ambari-agent start
    ```

12. 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.

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

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

14. If you have customized logging properties, you will see a Restart indicator next to each service name after upgrading to Ambari 2.0.0.

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:

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

15. Review the HDP-UTILS repository Base URL setting in Ambari.

    If you are upgrading from Ambari 1.6.1 or earlier, the HDP-UTILS repository Base URL is no longer set in the `ambari.repo` file.

    If using HDP 2.2 Stack:

    - Browse to Ambari Web > Admin > Stack and Versions.
    - Click on the Versions tab.
    - You will see the current installed HDP Stack version displayed.
    - Click the Edit repositories icon in the upper-right of the version display 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.

    If using HDP 2.0 or 2.1 Stack:

    - Browse to Ambari Web > Admin > Stack and Versions.
    - Under the Services table, the current Base URL settings are displayed.
    - Confirm the value of the HDP-UTILS repository Base URL is correct for your environment or click the Edit button to modify the HDP-UTILS Base URL.
    - 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.

16. If using HDP 2.2 Stack, you must get the cluster hosts to advertise the "current version". This can be done by restarting a master or slave component (such as a DataNode) on each host to have the host advertise its version so Ambari can record the version. For example, in Ambari Web, navigate to the Hosts page and select any Host that has the DataNode component, then restart that DataNode component on that single host.
17. If you have configured Ambari to authenticate against an external LDAP or Active Directory, review your Ambari LDAP authentication settings. You must re-run "ambari-server setup-ldap". For more information, see Set Up LDAP or Active Directory Authentication.
18. If you have configured your cluster for Hive or Oozie with an external database (Oracle, MySQL or PostgreSQL), you must re-run "ambari-server setup --jdbc-db and --jdbc-driver" to get the JDBC driver JAR file in place. For more information, see Using Non-Default Databases - Hive and Using Non-Default Databases - Oozie.
19. Adjust your cluster for Ambari Alerts and Metrics. For more information, see Planning for Ambari Alerts and Metrics in Ambari 2.0.
20. Adjust your cluster for Kerberos (if already enabled). For more information, see Upgrading Ambari with Kerberos-Enabled Cluster.

Planning for Ambari Alerts and Metrics in Ambari 2.0

As part of Ambari 2.0, Ambari includes built-in systems for alerting and metrics collection. Therefore, when upgrading to Ambari 2.0, the legacy Nagios and Ganglia services must be removed and replaced with the new systems.

We strongly recommended that you perform and validate this procedure in a test environment prior to attempting the Ambari upgrade on production systems.

Moving from Nagios to Ambari Alerts

After upgrading to Ambari 2.0, the Nagios service will be removed from the cluster. The Nagios server and packages will remain on the existing installed host but Nagios itself is removed from Ambari management.

Nagios used the operating system sendmail utility to dispatch email alerts on changes. With Ambari Alerts, the email dispatch is handled from the Ambari Server via Javamail. Therefore, you must provide SMTP information to Ambari for sending email alerts. Have this information ready. You will use it after the Ambari 2.0 upgrade to get Ambari Alert email notifications configured in the new Ambari Alerts system.

The Ambari Alerts system is configured automatically to replace Nagios but you must:

1. Configure email notifications in Ambari to handle dispatch of alerts. Browse to `Ambari Web > Alerts`.
2. In the Actions menu, select `Manage Notifications`.
3. Click to `Create a new Notification`. Enter information about the SMTP host, port to and from email addresses and select the Alerts to receive notifications.
4. Click `Save`.

(Optional) Remove the Nagios packages (nagios, nagios-www) from the Nagios host.

For more information Ambari Alerts, see Managing Alerts in the Ambari User's Guide.

Moving from Ganglia to Ambari Metrics

After upgrading to Ambari 2.0, the Ganglia service stays intact in cluster. You must perform the following steps to remove Ganglia from the cluster and to move to the new Ambari Metrics system.

- If you are using HDP 2.2 Stack, Storm metrics will not work with Ambari Metrics until you are upgraded to HDP 2.2.4 or later.
- Do not add the Ambari Metrics service to your cluster until you have removed Ganglia using the steps below.

1. Stop Ganglia service via Ambari Web.
2. Using the Ambari REST API, remove the Ganglia service by executing the following:

```
curl -u <admin_user_name>:<admin_password> -H 'X-Requested-By:ambari' -X DELETE 'http://<ambari_server_host>:8080/api/v1/clusters/<cluster_name>/services/GANGLIA'
```

3. Refresh Ambari Web and make sure that Ganglia service is no longer visible.
4. In the Actions menu on the left beneath the list of Services, use the "Add Service" wizard to add `Ambari Metrics` to the cluster.

   This will install an `Ambari Metrics Collector` into the cluster, and an `Ambari Metrics Monitor` on each host.
5. Pay careful attention to following service configurations:

Section	Property	Description	Default Value
Advanced ams-hbase-site	hbase.rootdir	Ambari Metrics service uses HBase as default storage backend. Set the rootdir for HBase to either local filesystem path if using Ambari Metrics in embedded mode or to a HDFS dir. For example: hdfs://namenode.example.org:8020/amshbase.	file:///var/lib/ambari-metrics-collector/hbas

6. For the cluster services to start sending metrics to Ambari Metrics, restart all services. For example, restart HDFS, YARN, HBase, Flume, Storm and Kafka.

(Optional) Remove the Ganglia packages (ganglia-gmetad and ganglia-gmond) from the hosts.

If you are managing a HDP 2.2 cluster that includes Kafka, you must adjust the Kafka configuration to send metrics to the Ambari Metrics system.

From Ambari Web, browse to `Services > Kafka > Configs` and edit the `kafka-env template` found under `Advanced kafka-env` to include the following:

```
# Add kafka sink to classpath and related dependencies if [ -e "/usr/lib/ambari-metrics-kafka-sink/ambari-metrics-kafka-sink.jar" ]; then export CLASSPATH=$CLASSPATH:/usr/lib/ambari-metrics-kafka-sink/ambari-metrics-kafka-sink.jar export CLASSPATH=$CLASSPATH:/usr/lib/ambari-metrics-kafka-sink/lib/* fi
```

Upgrading Ambari with Kerberos-Enabled Cluster

If you are upgrading to Ambari 2.0 from an Ambari-managed cluster that is already Kerberos enabled, because of the new Ambari 2.0 Kerberos features, you need perform the following steps after Ambari upgrade.

1. Review the procedure for Configuring Ambari and Hadoop for Kerberos in the Ambari Security Guide.
2. Have your Kerberos environment information readily available, including your KDC Admin account credentials.
3. Take note of current Kerberos security settings for your cluster.
   - Browse to Services > HDFS > Configs.
   - Record the core-site auth-to-local property value.
4. Upgrade Ambari according to the steps in Upgrading to Ambari 2.0.
5. Ensure your cluster and the Services are healthy.
6. Browse to `Admin > Kerberos` and you'll notice Ambari thinks that Kerberos is not enabled. Run the `Enable Kerberos Wizard`, following the instructions in the Ambari Security Guide.
7. Ensure your cluster and the Services are healthy.
8. Verify the Kerberos security settings for your cluster are correct.
   - Browse to Services > HDFS > Configs.
   - Check the core-site auth-to-local property value.
   - Adjust as necessary, based on the pre-upgrade value recorded in Step 3.

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 v2.2.0 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 2.2.0 stack. For example, if you want to run Storm or Falcon components on the HDP 2.2.0 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.

Use this procedure for upgrading from HDP 2.1 to any of the HDP 2.2 maintenance releases. For example, to HDP 2.2.4. The instructions in this document refer to HDP 2.2.x.x as a placeholder. To use an HDP 2.2.x.x maintenance release, be sure to replace 2.2.x.x in the following instructions with the appropriate maintenance version, such as 2.2.0.0 for the HDP 2.2 GA release, or 2.2.4.2 for an HDP 2.2 maintenance release.

Refer to the HDP documentation for the information about the latest HDP 2.2 maintenance releases.

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, **disable** Kerberos before performing the Stack upgrade. On `Ambari Web UI > Admin > Security`, click **Disable Kerberos**. You can re-enable Kerberos 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.

Once the above tasks have been performed:

1. Use Ambari Web, browse to `Services`. Go thru each service and in the `Service Actions` menu, select `Stop All`, except for 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.

   - 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
     ```

   - 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
     ```

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

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

   - 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.x. For specific information, see Database Requirements.

- 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.

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>	sudo -u <username> psql <databasename> < <inputfilename.sql>

Database Type	Backup	Restore
	For example: sudo -u postgres pg_dump hive > /tmp/mydir/backup_hive.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

- 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.

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. Stage the upgrade script.
    - Create an "Upgrade Folder". For example, `/work/upgrade_hdp_2`, on a host that can communicate with Ambari Server. The Ambari Server host is a suitable candidate.
    - Copy the upgrade script to the Upgrade Folder. The script is available here: `/var/lib/ambari-server/resources/scripts/upgradeHelper.py` on the Ambari Server host.
    - Copy the upgrade catalog to the Upgrade Folder. The catalog is available here: `/var/lib/ambari-server/resources/upgrade/catalog/UpgradeCatalog_2.1_to_2.2.x.json`.

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.

16. Backup current configuration settings.
    - Go to the Upgrade Folder you just created in step 15.
    - Execute the backup-configs action:

      python upgradeHelper.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.
17. On the Ambari Server host, stop Ambari Server and confirm that it is stopped.

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

18. 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:

Be sure to **replace GA/2.2.x.x** in the following instructions with the appropriate maintenance version, such as GA/2.2.0.0 for the HDP 2.2 GA release, or updates/2.2.4.2 for an HDP 2.2 maintenance release.

For RHEL/CentOS/Oracle Linux 6:

```
wget -nv http://public-repo-1.hortonworks.com/HDP/centos6/2.x/GA/2.2.x.x/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.x.x/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.x.x/hdp.repo -O
/etc/zypp/repos.d/HDP.repo
```

For Ubuntu12:

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/ubuntu12/2.x/GA/2.2.x.x/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.x.x/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 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:

- On all hosts, clean the yum repository.

```
yum clean all
```

- 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"
```

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

```
yum install "hadoop_2_2_x_0_*" "oozie_2_2_x_0_*" "pig_2_2_x_0_*"
"sqoop_2_2_x_0_*" "zookeeper_2_2_x_0_*" "hbase_2_2_x_0_*"
"hive_2_2_x_0_*" "tez_2_2_x_0_*" "storm_2_2_x_0_*"
"falcon_2_2_x_0_*" "flume_2_2_x_0_*" "phoenix_2_2_x_0_*"
"accumulo_2_2_x_0_*" "mahout_2_2_x_0_*" rpm -e --nodeps hue-shell
yum install hue hue-common hue-beeswax hue-hcatalog hue-pig hue-
oozie
```

- 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:

- On all hosts, clean the zypper repository.

```
zypper clean --all
```

- 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"
```

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

```
zypper install "hadoop\_2_2_x_0_*" "oozie\_2_2_x_0_*"
"pig\_2_2_x_0_*" "sqoop\_2_2_x_0_*" "zookeeper\_2_2_x_0_*"
"hbase\_2_2_x_0_*" "hive\_2_2_x_0_*" "tez\_2_2_x_0_*"
"storm\_2_2_x_0_*" "falcon\_2_2_x_0_*" "flume\_2_2_x_0_*"
"phoenix\_2_2_x_0_*" "accumulo\_2_2_x_0_*" "mahout\_2_2_x_0_*"  rpm
-e --nodeps hue-shell zypper install hue hue-common hue-beeswax hue-
hcatalog hue-pig hue-oozie
```

- 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.

- 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.x.x-xxxx.el6.noarch for the HDP 2.2.x release.
```

   If not, then run: yum install hdp-select

   Run `hdp-select` as root, on every node. `hdp-select set all 2.2.x.x-<$version>` where `$version` is the build number.

   For the HDP 2.2.4.2 release <$version> = 2.

7. Verify that all components are on the new version. The output of this statement should be empty,

```
hdp-select status | grep -v 2\.2\.x\.x | grep -v None
```

8. If you are using Hue, you must upgrade Hue manually. For more information, see Confiure and Start Hue.

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 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/2.2.x.x-<$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.x.x-<$version>/hive/bin/schematool -upgradeSchema -dbType <databaseType>

      where <$version> is the 2.2.x 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 Managing Stacks and Versions.

   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 2.1 Stack to 2.2: 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. Update the respective configurations.
   - Go to the Upgrade Folder you created when Preparing the 2.1 Stack for Upgrade.
   - Execute the update-configs action:

     ```
     python upgradeHelper.py --hostname $HOSTNAME --user $USERNAME --
     password $PASSWORD --clustername $CLUSTERNAME --
     fromStack=$FROMSTACK --toStack=$TOSTACK --
     upgradeCatalog=$UPGRADECATALOG update-configs [configuration
     item]
     ```

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
- <FROMSTACK> is the version number of pre-upgraded stack, for example 2.1
- <TOSTACK> it the version number of the upgraded stack, for example 2.2.x
- <UPGRADECATALOG> is the path to the upgrade catalog file, for example UpgradeCatalog_2.1_to_2.2.x.json

For example, to update all configuration items:

```
python upgradeHelper.py --hostname $HOSTNAME --user $USERNAME --
password $PASSWORD --clustername $CLUSTERNAME --fromStack=2.1 --
toStack=2.2.x --upgradeCatalog=UpgradeCatalog_2.1_to_2.2.x.json
update-configs
```

To update configuration item hive-site:

```
python upgradeHelper.py --hostname $HOSTNAME --user $USERNAME --
password $PASSWORD --clustername $CLUSTERNAME --fromStack=2.1 --
toStack=2.2.x --upgradeCatalog=UpgradeCatalog_2.1_to_2.2.x.json
update-configs hive-site
```

5. 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.
6. Using the Ambari Web UI, add any new services that you want to run on the HDP 2.2.x stack. You must add a Service before editing configuration properties necessary to complete the upgrade.
7. Using the `Ambari Web UI` > `Services`, start the ZooKeeper service.
8. 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; 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
   ```

9. 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.x.x-<$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.

10. 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.x.x-<$version>/hadoop/libexec && /usr/hdp/2.2.x.x-<$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`.

11. Start all DataNodes.

    At each DataNode, as the HDFS user,

    ```
    su -l <HDFS_USER> -c "/usr/hdp/2.2.x.x-<$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.

12. Restart HDFS.
    - 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.
    - 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.

- 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. If YARN is installed in your HDP 2.1 stack, and the Application Timeline Server (ATS) component is **NOT**, then you must create and install ATS component 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.

    - 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>/service
      s/YARN/components/APP_TIMELINE_SERVER
      ```

    - 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_component_hostname>/host_components/APP_TIMELINE_SERVER
      ```

    - 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_component_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.x.x-
      <$version>/mapreduce/"
      ```

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

      ```
      su -l <HDFS_USER> -c "hdfs dfs -copyFromLocal /usr/hdp/2.2.x.x-
      <$version>/hadoop/mapreduce.tar.gz /hdp/apps/2.2.x.x-
      <$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.x.x-<$version>/mapreduce"; su -l
      <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.2.x.x-
      <$version>/mapreduce/mapreduce.tar.gz"
      ```

- Update YARN Configuration Properties for HDP 2.2.x

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

  - Add

Name	Value
hadoop.registry.zk.quorum	<!--List of hostname:port pairs defining the zookeeper quorum binding for the registry-->
yarn.resourcemanager.zk-address	localhost:2181

- Update Hive Configuration Properties for HDP 2.2.x
  - Using Ambari Web UI > Services > Hive > Configs > Advanced webhcat-site:

    Find the `templeton.hive.properties` property and remove whitespaces after "," from the value.
  - 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.zookeeper.quorum	<!-- List of zookeeper server to talk to -->

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.

- **Perform the following preparation steps on each Oozie server host:**

You must replace your Oozie configuration after upgrading.

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

- Create /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22 directory.

  mkdir /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22

- Copy the JDBC jar of your Oozie database to both /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22 and /usr/hdp/2.2.x.x-<$version>/oozie/libtools. For example, if you are using MySQL, copy your mysql-connector-java.jar.
- Copy these files to /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22 directory

  cp /usr/lib/hadoop/lib/hadoop-lzo*.jar /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22; cp /usr/share/HDP-oozie/ext-2.2.zip /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22; cp /usr/share/HDP-oozie/ext-2.2.zip /usr/hdp/2.2.x.x-<$version>/oozie/libext
- Grant read/write access to the Oozie user.

  chmod -R 777 /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22

- **Upgrade steps:**
  - On the Services view, make sure that YARN and MapReduce2 services are running.
  - Make sure that the Oozie service is stopped.
  - 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`.
  - Upgrade Oozie. At the Oozie database host, as the Oozie service user:

    `sudo su -l <OOZIE_USER> -c"/usr/hdp/2.2.x.x-<$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>`.
  - 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.x.x-<$version>/oozie/bin/oozie-setup.sh prepare-war -d /usr/hdp/2.2.x.x-<$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".

- 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.

- Replace the content of /usr/oozie/share in HDFS.

  On the Oozie server host:

  - Extract the Oozie sharelib into a tmp folder.

    ```
    mkdir -p /tmp/oozie_tmp; cp /usr/hdp/2.2.x.x-
    <$version>/oozie/oozie-sharelib.tar.gz /tmp/oozie_tmp; cd
    /tmp/oozie_tmp; tar xzvf oozie-sharelib.tar.gz;
    ```

  - 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.

  - 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.

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

    Make sure that ServiceCheck passes for Oozie.

23. Update WebHCat.

    - 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

- 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.

- 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:

  - 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>"
    ```

  - Remove the old file.

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

  - Copy the new file.

    ```
    su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -
    copyFromLocal /usr/hdp/2.2.x.x-<$version>/hive/hive.tar.gz
    /apps/webhcat/"; su -l <HCAT_USER> -c "hdfs --config
    /etc/hadoop/conf dfs -copyFromLocal /usr/hdp/2.2.x.x-
    <$version>/pig/pig.tar.gz /apps/webhcat/";
    ```

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

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

  - 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>"
    ```

  - Remove the old file.

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

- Copy the new hadoop-streaming.jar file.

  ```
  su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -copyFromLocal /usr/hdp/2.2.x.x-<$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 Pid.

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.

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

  ```
  su -l <HDFS_USER> -c "hdfs dfs -mkdir -p /hdp/apps/2.2.x.x-<$version>/tez/"  su -l <HDFS_USER> -c "hdfs dfs -copyFromLocal -f /usr/hdp/2.2.x.x-<$version>/tez/lib/tez.tar.gz /hdp/apps/2.2.x.x-<$version>/tez/." su -l <HDFS_USER> -c "hdfs dfs -chown -R <HDFS_USER>:<HADOOP_GROUP> /hdp" su -l <HDFS_USER> -c "hdfs dfs -chmod -R 555 /hdp/apps/2.2.x.x-<$version>/tez" su -l hdfs -c "hdfs dfs -chmod -R 444 /hdp/apps/2.2.x.x-<$version>/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
  ```

- Stop the Storm Rest_API Component.

  ```
  curl -u admin:admin -X PUT -H 'X-Requested-By:1' -d '{"RequestInfo":{"context":"Stop Component"},"Body":{"HostRoles":{"state":"INSTALLED"}}}' http://server:8080/api/v1/clusters/c1/hosts/host_name/host_components/STORM_REST_API
  ```

In HDP 2.2, STORM_REST_API component was deleted because the service was moved into STORM_UI_SERVER. When upgrading from HDP 2.1 to 2.2, you must delete this component using the API as follows:

- Delete the Storm Rest_API Component.

  ```
  curl -u admin:admin -X DELETE -H 'X-Requested-By:1' http://server:8080/api/v1/clusters/c1/hosts/host_name/host_components/STORM_REST_API
  ```

26. Upgrade Pig.

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

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

27. Using `Ambari Web UI` > `Services` > `Storm`, start the Storm service.
28. Using `Ambari Web` > `Services` > `Service Actions`, re-start all stopped 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"
```

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.x on your current, Ambari-installed-and-managed cluster.

1. Prepare the 2.0 Stack for Upgrade
2. Upgrade the 2.0 Stack to 2.2
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.

Use this procedure for upgrading from HDP 2.0 to any of the HDP 2.2 maintenance releases. For example, to HDP 2.2.4. The instructions in this document refer to HDP 2.2.x.x as a placeholder. To use an HDP 2.2.x.x maintenance release, be sure to replace 2.2.x.x in the following instructions with the appropriate maintenance version, such as 2.2.0.0 for the HDP 2.2 GA release, or 2.2.4.2 for an HDP 2.2 maintenance release.

Refer to the HDP documentation for the information about the latest HDP 2.2 maintenance releases.

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.

Once the above tasks have been completed:

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.

- 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
```

- 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
```

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

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

- 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.x. For specific information, see Database Requirements.

- Optional - Back up the Hive Metastore database.

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

Hive Metastore Database Backup and Restore

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

Database Type	Backup	Restore
Oracle	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

- Optional - Back up the Oozie Metastore database.

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

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. Stage the upgrade script.
    - Create an "Upgrade Folder". For example, `/work/upgrade_hdp_2`, on a host that can communicate with Ambari Server. The Ambari Server host is a suitable candidate.
    - Copy the upgrade script to the Upgrade Folder. The script is available on the Ambari Server host in `/var/lib/ambari-server/resources/scripts/upgradeHelper.py`.
    - Copy the upgrade catalog to the Upgrade Folder. The catalog is available in `/var/lib/ambari-server/resources/upgrade/catalog/UpgradeCatalog_2.0_to_2.2.x.json`.

Make sure that Python is available on the host and that the version is 2.6 or higher: `python -version` For RHEL/Centor/Oracle Linus 5, you must use Python 2.6.

15. Backup current configuration settings:
    - Go to the Upgrade Folder you just created in step 14.
    - Execute the backup-configs action:

      ```
      python upgradeHelper.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.
16. On the Ambari Server host, stop Ambari Server and confirm that it is stopped.

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

17. 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:

Be sure to **replace GA/2.2.x.x** in the following instructions with the appropriate maintenance version, such as GA/2.2.0.0 for the HDP 2.2 GA release, or updates/2.2.4.2 for an HDP 2.2 maintenance release.

For RHEL/CentOS/Oracle Linux 6:

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/centos6/2.x/GA/2.2.x.x/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.x.x/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.x.x/hdp.repo -O
/etc/zypp/repos.d/HDP.repo
```

For UBUNTU 12:

```
wget -nv http://public-repo-
1.hortonworks.com/HDP/ubuntu12/2.x/GA/2.2.x.x/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.x.x/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, 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:

- On all hosts, clean the yum repository.

  ```
  yum clean all
  ```

- 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"
  ```

- Install the following components:

  ```
  yum install "hadoop_2_2_x_0_*" "oozie_2_2_x_0_*" "pig_2_2_x_0_*"
  "sqoop_2_2_x_0_*" "zookeeper_2_2_x_0_*" "hbase_2_2_x_0_*"
  "hive_2_2_x_0_*" "flume_2_2_x_0_*" "phoenix_2_2_x_0_*"
  "accumulo_2_2_x_0_*" "mahout_2_2_x_0_*"  rpm -e --nodeps hue-
  shell  yum install hue hue-common hue-beeswax hue-hcatalog hue-
  pig hue-oozie
  ```

- Verify that the components were upgraded.

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

  Nothing should appear in the returned list.

For SLES:

- On all hosts, clean the zypper repository.

  ```
  zypper clean --all
  ```

- 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"
  ```

- Install the following components:

  ```
  zypper install "hadoop\_2_2_x_0_*" "oozie\_2_2_x_0_*"
  "pig\_2_2_x_0_*" "sqoop\_2_2_x_0_*" "zookeeper\_2_2_x_0_*"
  "hbase\_2_2_x_0_*" "hive\_2_2_x_0_*" "flume\_2_2_x_0_*"
  "phoenix\_2_2_x_0_*" "accumulo\_2_2_x_0_*" "mahout\_2_2_x_0_*"
  rpm -e --nodeps hue-shell  zypper install hue hue-common hue-
  beeswax hue-hcatalog hue-pig hue-oozie
  ```

- 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.

- 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-2.2.4.4-2.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.x.x-<$version>` where <$version> is the build number. **For the HDP 2.2.4.2 release <$version> = 2**.

Check that the `hdp-select` package installed:

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

You should see: `hdp-select-2.2.4.2-2.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.x.x-<$version>`

where <$version> is the build number. **For the HDP 2.2.4.2 release <$version> = 2**.

7. Verify that all components are on the new version. The output of this statement should be empty,

```
hdp-select status | grep -v 2\.2\.x\.x | grep -v None
```

8. If you are using Hue, you must upgrade Hue manually. For more information, see Confiure and Start Hue.
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 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/2.2.x.x-<$version>/hive/lib`, if it is 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.x.x-<$version>/hive/bin/schematool -upgradeSchema -
  dbType <databaseType>
  ```

  where <$version> is the 2.2.x 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, `ambari-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 2.2.0 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 Managing Stacks and Versions..

   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. Update the respective configurations.

   - Go to the Upgrade Folder you created when Preparing the 2.0 Stack for Upgrade.
   - Execute the update-configs action:

     ```
     python upgradeHelper.py --hostname $HOSTNAME --user $USERNAME --
     password $PASSWORD --clustername $CLUSTERNAME --
     fromStack=$FROMSTACK --toStack=$TOSTACK --
     upgradeCatalog=$UPGRADECATALOG update-configs [configuration
     item]
     ```

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

<FROMSTACK> is the version number of pre-upgraded stack, for example 2.0

<TOSTACK> it the version number of the upgraded stack, for example 2.2.x

<UPGRADECATALOG> is the path to the upgrade catalog file, for example UpgradeCatalog_2.0_to_2.2.x.json

For example, To update all configuration items:

```
python upgradeHelper.py --hostname $HOSTNAME --user $USERNAME --
password $PASSWORD --clustername $CLUSTERNAME --fromStack=2.0 --
toStack=2.2.x --upgradeCatalog=UpgradeCatalog_2.0_to_2.2.x.json
update-configs
```

To update configuration item hive-site:

```
python upgradeHelper.py --hostname $HOSTNAME --user $USERNAME --
password $PASSWORD --clustername $CLUSTERNAME --fromStack=2.0 --
toStack=2.2.x --upgradeCatalog=UpgradeCatalog_2.0_to_2.2.x.json
update-configs hive-site
```

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

7. 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.x.x-<$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.

8. 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.x.x-
   <$version>/hadoop/libexec && /usr/hdp/2.2.x.x-
   <$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.

9. Start all DataNodes.

   On each DataNode, as the HDFS user,

   ```
   su -l <HDFS_USER> -c "/usr/hdp/2.2.x.x-<$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.

10. Restart HDFS.

    - 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.
    - 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.

- 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. 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.
authorization.StorageBasedAuthorizationProvider
hive.security.metastore.authorization.manager=org.apache.hadoop.hive.ql
.security.authorization.StorageBasedAuthorizationProvider
hive.security.authenticator.manager=org.apache.hadoop.hive.ql.security.
ProxyUserAuthenticator
```

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.

14. Update Hive Configuration Properties for HDP 2.2.x

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

    - hive-site

Name	Value
hive.cluster.delegation.token.store.zookeeper.connectString	<!-- The ZooKeeper token store connect string. -->
hive.zookeeper.quorum	<!-- List of zookeeper servers to talk to -->

    - webhcat-site

Name	Value
templeton.hive.properties	<!-- Properties to set when running hive -->

Pay attention: values should not contain white space after each comma.

15. 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.

- 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>/service
s/YARN/components/APP_TIMELINE_SERVER
```

- 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_component_hostname>/host_components/APP_TIMELINE_SERVER
```

- 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_component_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. 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.NullAppli
cationHistoryStore 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.

17. 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.x.x-
      <$version>/mapreduce/"
      ```

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

      ```
      su -l <HDFS_USER> -c "hdfs dfs -copyFromLocal /usr/hdp/2.2.x.x-
      <$version>/hadoop/mapreduce.tar.gz /hdp/apps/2.2.x.x-
      <$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.x.x-<$version>/mapreduce"; su -l
      <HDFS_USER> -c "hdfs dfs -chmod -R 444 /hdp/apps/2.2.x.x-
      <$version>/mapreduce/mapreduce.tar.gz"
      ```

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

Name	Value
hadoop.registry.zk.quorum	<!-- List of zookeeper servers to talk to -->
yarn.resourcemanager.zk-address	<!-- Zookeeper server to talk to -->
yarn.timeline-service.address	<!-- Timeline service fqdn address -->
yarn.timeline-service.webapp.address	<!-- Timeline service webapp fqdn address -->
yarn.timeline-service.webapp.https.address	<!-- Timeline service https webapp fqdn address -->

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

20. Using Ambari Web > Services > Service Actions, start HBase and ensure the service check passes.
21. Using Ambari Web > Services > Service Actions, start the Hive service.
22. Upgrade Oozie.
    - Perform the following preparation steps on each Oozie server host:

You must replace your Oozie configuration after upgrading.

      - Copy configurations from `oozie-conf-bak` to the `/etc/oozie/conf` directory on each Oozie server and client.
      - Create /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22 directory.

        ```
        mkdir /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22
        ```

      - Copy the JDBC jar of your Oozie database to both /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22 and /usr/hdp/2.2.x.x-<$version>/oozie/libtools. For example, if you are using MySQL, copy your mysql-connector-java.jar.
      - Copy these files to /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22 directory

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

      - Grant read/write access to the Oozie user.

        ```
        chmod -R 777 /usr/hdp/2.2.x.x-<$version>/oozie/libext-upgrade22
        ```

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

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

`sudo su -l <OOZIE_USER> -c"/usr/hdp/2.2.x.x-<$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>.

- 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.x.x-<$version>/oozie/bin/oozie-setup.sh prepare-war -d /usr/hdp/2.2.x.x-<$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".

- 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.

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

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

    `30000`

  - 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

- 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
  ```

- 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
  ```

- 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.

- 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.

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

  Indicates if Oozie is configured to use Kerberos.

- 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
  ```

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

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

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

  ```
  mkdir -p /tmp/oozie_tmp; cp /usr/hdp/2.2.x.x-
  <$version>/oozie/oozie-sharelib.tar.gz /tmp/oozie_tmp; cd
  /tmp/oozie_tmp; tar xzvf oozie-sharelib.tar.gz;
  ```

- 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.

- 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.

- Add the Falcon Service, using `Ambari Web > Services > Actions > +Add Service`. Without Falcon, Oozie will fail.
- Use the `Ambari Web UI` > `Services` view to start the Oozie service. Make sure that ServiceCheck passes for Oozie.

23. Update WebHCat.

- Expand Advanced > webhcat-site.xml.

  Check if templeton.hive.properties is set correctly.

- 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:

- 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>"
  ```

- Remove the old file.

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

- Copy the new file.

  ```
  su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -
  copyFromLocal /usr/hdp/2.2.x.x-<$version>/hive/hive.tar.gz
  /apps/webhcat/"; su -l <HCAT_USER> -c "hdfs --config
  /etc/hadoop/conf dfs -copyFromLocal /usr/hdp/2.2.x.x-
  <$version>/pig/pig.tar.gz /apps/webhcat/";
  ```
  where <HCAT_USER> is the HCatalog service user. For example, hcat.

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

  - Remove the old file.

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

  - Copy the new hadoop-streaming.jar file.

    ```
    su -l <HCAT_USER> -c "hdfs --config /etc/hadoop/conf dfs -
    copyFromLocal /usr/hdp/2.2.x.x-<$version>/hadoop-
    mapreduce/hadoop-streaming*.jar /apps/webhcat"
    ```

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

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

The Tez client should also be installed on the Pig host.

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.

25. Using the `Ambari Web UI> Services > Hive`, start the Hive service.

26. 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.

27. Using `Ambari Web > Services`, re-start the remaining services.
28. 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"
```

Automated HDP Stack Upgrade: HDP 2.2.0 to 2.2.4

Ambari 2.0 has the capability to perform an automated cluster upgrade for maintenance and patch releases for the Stack. This capability is available for HDP 2.2 Stack only. If you have a cluster running HDP 2.2, you can perform Stack upgrades to later maintenance and patch releases. For example: you can upgrade from the GA release of HDP 2.2 (which is HDP 2.2.0.0) to the first maintenance release of HDP 2.2 (which is HDP 2.2.4.2).

This section describes the steps to perform an upgrade from HDP 2.2.0 to HDP 2.2.4.

Prerequisites

To perform an automated cluster upgrade from Ambari, your cluster must meet the following prerequisites:

Item	Requirement	Description
Cluster	Stack Version	Must be running HDP 2.2 Stack. This capability is not available for HDP 2.0 or 2.1 Stacks.
Version	Target Version	All hosts must have the target version installed. See the Register Version and Install Version sections for more information.
HDFS	NameNode HA	NameNode HA must be enabled and working properly. See the Ambari User's Guide for more information Configuring NameNode High Availability.
HDFS	Decommission	No components should be in decommissioning or decommissioned state.
YARN	YARN WPR	Work Preserving Restart must be configured.
Hosts	Heartbeats	All Ambari Agents must be heartbeating to Ambari Server. Any hosts that are not heartbeating must be in Maintenance Mode.
Hosts	Maintenance Mode	Any hosts in Maintenance Mode must not be hosting any Service master components.
Services	Services Up	All Services must be started.
Services	Maintenance Mode	No Services can be in Maintenance Mode.

If you do not meet the upgrade prerequisite requirements listed above, you can consider a Manual Upgrade of the cluster.

Preparing to Upgrade

It is highly recommended that you perform backups of your Hive Metastore and Oozie Server databases prior to beginning upgrade.

Registering a New Version

Register the HDP 2.2.4.2 Version

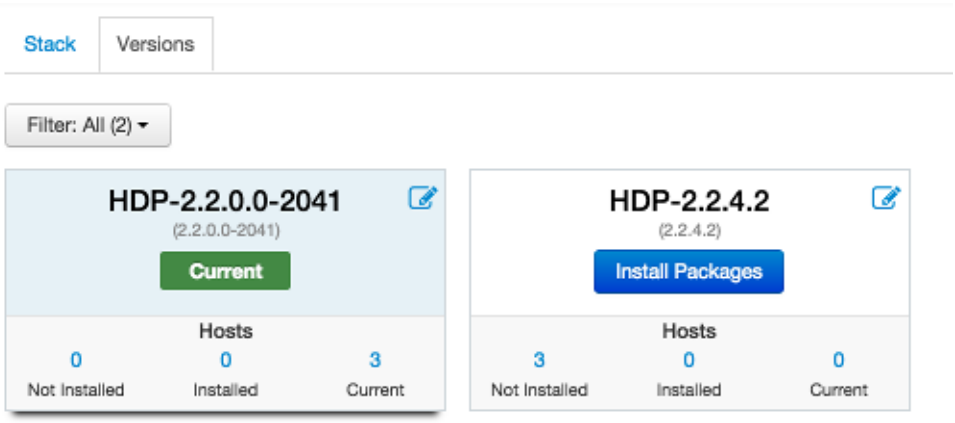
- Log in to Ambari.
- Browse to `Admin > Stack and Versions`.
- Click on the `Versions` tab. Click `Manage Versions`.
- Proceed to register a new version by clicking `+ Register Version`.
- Enter a two-digit version number. For example, enter **4.2** (which makes the version name **HDP-2.2.4.2**).

- Select one or more OS families and enter the respective Base URLs.
- Click `Save`.
- You can click “Install On...MyCluster”, or you can browse back to `Admin > Stack and Versions`. You will see the version current running (HDP 2.2.0.0) and the version you just registered (HDP 2.2.4.2). Proceed to Install a New Version on All Hosts.

Installing a New Version on All Hosts

Install HDP 2.2.4.2 on All Hosts

- Log in to Ambari.
- Browse to `Admin > Stack and Versions`.
- Click on the `Versions` tab.



- Click `Install Packages` and click OK to confirm.
- The Install version operation will start and the new version will be installed on all hosts.
- You can browse to Hosts and to each host > Versions tab to see the new version is installed. Proceed to Perform Upgrade.

Performing an Upgrade

Perform the Upgrade to HDP 2.2.4.2

- Log in to Ambari.
- Browse to `Admin > Stack and Versions`.
- Click on the `Versions` tab.
- Click `Perform Upgrade`.

Manual HDP Stack Upgrade: HDP 2.2.0 to 2.2.4

The following sections describe the steps involved with performing a manual Stack upgrade:

- Registering a New Version
- Installing a New Version on All Hosts
- Performing a Manual Upgrade

This is an alternative to using the Automated Upgrade feature of Ambari when using the HDP 2.2 Stack.

Registering a New Version

Register the HDP 2.2.4.2 Version

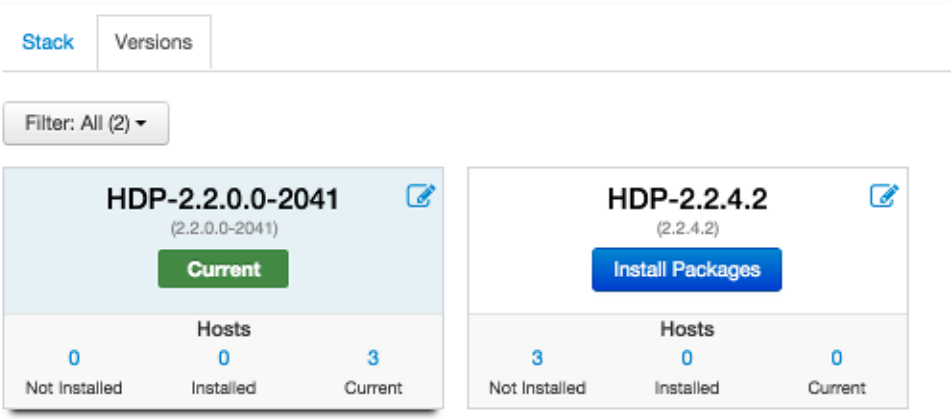
- Log in to Ambari.
- Browse to `Admin > Stack and Versions`.
- Click on the `Versions` tab. You will see the version current running **HDP-2.2.0.0-2041**.
- Click `Manage Versions`.
- Proceed to register a new version by clicking `+ Register Version`.
- Enter a two-digit version number. For example, enter **4.2**(which makes the version name **HDP-2.2.4.2**).

- Select one or more OS families and enter the repository Base URLs for that OS.
- Click `Save`.
- Click `Go to Dashboard` and browse back to `Admin > Stack and Versions > Versions`. You will see the current running version **HDP-2.2.0.0-2041** and the version you just registered **HDP-2.2.4.2**. Proceed to Install a New Version on All Hosts.

Installing a New Version on All Hosts

Install HDP 2.2.4.2 on All Hosts

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab.

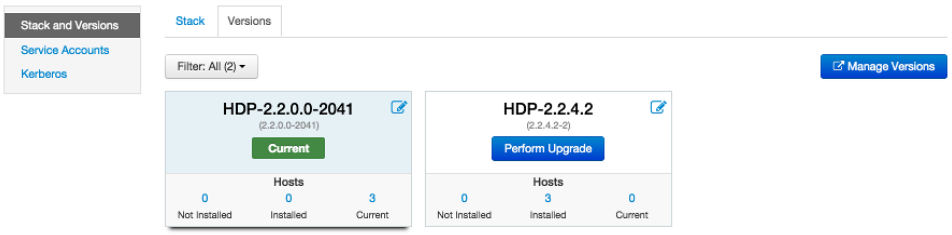


4. Click `Install Packages` and click OK to confirm.
5. The Install version operation will start and the new version will be installed on all hosts.
6. You can browse to Hosts and to each host > Versions tab to see the new version is installed. Proceed to Perform Manual Upgrade.

Performing a Manual Upgrade

Perform the Manual Upgrade to HDP 2.2.4.2

1. Log in to Ambari.
2. Browse to `Admin > Stack and Versions`.
3. Click on the `Versions` tab.
4. Under the newly registered and installed version **HDP-2.2.4.2**, is the actual software repository version in parenthesis (Ambari determined this repository version during the install). For example, in the picture below the display name is **HDP-2.2.4.2** and the repository version **2.2.4.2-2**. Record this repository version. You will use it later in the manual upgrade process.



5. Stop all services from Ambari. On the Services tab, in the Service navigation area Actions button, select `Stop All` to stop all services.

If you are upgrading a NameNode HA configuration, keep your JournalNodes running while performing this upgrade procedure. Upgrade, rollback and finalization operations on HA NameNodes must be performed with all JournalNodes running.

6. Go to the command line on each host and move the current HDP version to the newly installed version using the hdp-select utility and repository version number (obtained in Step 4).

   ```
   hdp-select set all {repository-version}
   ```

   For example:

   ```
   hdp-select set all 2.2.4.2-2
   ```

7. Restart all services from Ambari. One by one, browse to each Service in Ambari Web, and in the Service Actions menu select `Restart All`. Do not select Start All. You must use Restart All. For example, browse to `Ambari Web > Services > HDFS` and select `Restart All`.

8. During a manual upgrade, it is necessary for all components to advertise the version that they are on. This is typically done by Restarting an entire Service. However, ***client-only services*** (e.g., Pig, Tez and Slider) do not have a Restart command. Instead, they need an API call that will trigger the same behavior. For each of services installed that are client-only issue an Ambari REST API call that will cause the hosts running these clients to advertise their version. Perform this REST API call for each client-only service configured in your cluster:

   ```
   curl -X POST -u username:password -H 'X-Requested-By:ambari'
   http://ambari.server:8080/api/v1/clusters/MyCluster/requests '{
   "RequestInfo":   { "command":"RESTART",     "context":"Restart all
   components for TEZ_CLIENT",     "operation_level": {
   "level":"SERVICE",          "cluster_name":"MyCluster",
   "service_name":"TEZ"     }   },   "Requests/resource_filters": [{
   "service_name":"TEZ",     "component_name":"TEZ_CLIENT",
   "hosts":"c6401.ambari.apache.org,c6402.apache.ambari.org"}] }'
   ```

   Replace the Ambari Server username + password, Ambari Server hostname, your cluster name, service name + component name (see the following table), and the list of hosts in your cluster that are running the client.

Service	service_name	component_name
Tez	TEZ	TEZ_CLIENT
Pig	PIG	PIG

Service	service_name	component_name
Slider	SLIDER	SLIDER
Sqoop	SQOOP	SQOOP

9. After all the services are confirmed to be started and healthy, go to the command line on the Ambari Server and run the following to finalize the upgrade, which will move the current version to the new version.

```
ambari-server set-current --cluster-name=MyCluster --version-display-name=HDP-2.2.4.2      Ambari Admin login: admin     Ambari Admin password: *****
```

10. If the `ambari-server set-current` command is not successful, try restarting the Ambari Server and waiting for all agents to re-register before trying again.