

Hortonworks Data Platform

Ambari Reference Topics

(Apr 1, 2014)

Hortonworks Data Platform : Ambari Reference Topics

Copyright © 2012-2014 Hortonworks, Inc. All rights reserved.

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.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Table of Contents

1. Installing Ambari Agents Manually	1
1.1. RHEL/CentOS/Oracle Linux 5.x and 6.x	1
1.2. SLES	1
2. Using Custom Hostnames	2
3. Moving the Ambari Server	3
3.1. Back up Current Data	3
3.2. Update Agents	3
3.3. Install the New Server and Populate the Databases	4
4. Using Non-Default Databases	6
4.1. Hive/HCatalog	6
4.1.1. Troubleshooting Hive/HCatalog	7
4.2. Oozie	8
4.2.1. Troubleshooting Oozie	10
4.3. Ambari	11
4.3.1. Troubleshooting Ambari	12
5. Configuring Ambari Server for Internet Proxy	15
5.1. Setting up an Internet Proxy server for Ambari	15
6. Changing the Ambari Server Port	16
6.1. Optional: Change the Default Ambari Server Port	16
7. Changing the JDK Version on an Existing Cluster	17
8. NameNode High Availability	18
8.1. Setting Up NameNode High Availability	18
8.2. Rolling Back NameNode HA	24
8.2.1. Stop HBase	24
8.2.2. Checkpoint the Active NameNode	24
8.2.3. Stop All Services	25
8.2.4. Prepare the Ambari Server Host for Rollback	25
8.2.5. Restore the HBase Configuration	26
8.2.6. Delete ZK Failover Controllers	27
8.2.7. Modify HDFS Configurations	27
8.2.8. Recreate the Secondary NameNode	29
8.2.9. Re-enable the Secondary NameNode	29
8.2.10. Delete All JournalNodes	30
8.2.11. Delete the Additional NameNode	30
8.2.12. Verify your HDFS Components	31
8.2.13. Start HDFS	31
9. Configuring RHEL HA for Hadoop 1.x	32
9.1. Deploy the scripts	32
9.2. Configure Ambari properties across the HA cluster	32
9.3. Troubleshooting RHEL HA	33

List of Tables

4.1. Hive Security Authorization Settings	8
8.1. Core-site.xml properties and values for NameNode HA on a cluster using Hue	23
8.2. Set Environment Variables	25
9.1. Parameter Options for relocate_host_components.py	33

1. Installing Ambari Agents Manually

In some situations you may decide you do not want to have the Ambari Install Wizard install and configure the Agent software on your cluster hosts automatically. In this case you can install the software manually.

Before you begin: On every host in your cluster download the HDP repository as described in [Set Up the Bits](#).

1.1. RHEL/CentOS/Oracle Linux 5.x and 6.x

1. Install the Ambari Agent

```
yum install ambari-agent
```

2. Using a text editor, configure the Ambari Agent by editing the `ambari-agent.ini` file as shown in the following example:

```
vi /etc/ambari-agent/conf/ambari-agent.ini

[server]
hostname={your.ambari.server.hostname}
url_port=8440
secured_url_port=8441
```

3. Start the agent.

```
ambari-agent start
```

The agent registers with the Server on start.

1.2. SLES

1. Install the Ambari Agent.

```
zypper install ambari-agent
```

2. Configure the Ambari Agent by editing the `ambari-agent.ini` file as shown in the following example:

```
vi /etc/ambari-agent/conf/ambari-agent.ini

[server]
hostname={your.ambari.server.hostname}
url_port=8440
secured_url_port=8441
```

3. Start the agent.

```
ambari-agent start
```

The agent registers with the Server on start.

2. Using Custom Hostnames

You can customize the agent registration hostname and the public hostname used for each host in Ambari. Use this capability when "hostname" does not return the public network hostname for your machines.

To customize the name of each host in your cluster:

1. On the **Install Options** screen, select **Perform Manual Registration** for Ambari Agents.
2. Install the Agents manually, as described in [Installing Ambari Agents Manually](#).
3. To echo the customized name of the host to which the Ambari agent registers, for every host, create a script like the following example, named `/var/lib/ambari-agent/hostname.sh`.

```
#!/bin/sh
echo <ambari_hostname>
```

4. Open `/etc/ambari-agent/conf/ambari-agent.ini` on every host, using a text editor.
5. Add to the agent section the following line:

```
hostname_script=/var/lib/ambari-agent/hostname.sh
```

where `/var/lib/ambari-agent/hostname.sh` is the name of your custom echo script.

6. To generate a public hostname for every host, create a script like the following example, named `/var/lib/ambari-agent/public_hostname.sh` to show the name for that host in the UI.

```
#!/bin/bash
hostname <-f>
```

7. Open `/etc/ambari-agent/conf/ambari-agent.ini` on every host, using a text editor.
8. Add to the agent section the following line:

```
public_hostname_script=/var/lib/ambari-agent/public_hostname.sh
```

9. Add the hostnames to `/etc/hosts` on every host.

3. Moving the Ambari Server

Use the following instructions to transfer the Ambari Server to a new host.



Note

These steps describe moving the Ambari Server when it uses the default PostgreSQL database. If you are using a non-default database such as Oracle for Ambari, adjust the database backup, restore, and stop/start procedures to match that database.

1. [Back up all current data from the original Ambari Server and MapReduce databases.](#)
2. [Update all Agents to point to the new Ambari Server.](#)
3. [Install the Server on a new host and populate databases with information from original Server.](#)

3.1. Back up Current Data

1. Stop the original Ambari Server.

```
ambari-server stop
```

2. Create a directory to hold the database backups.

```
cd /tmp
mkdir dbdumps
cd dbdumps/
```

3. Create the database backups.

```
pg_dump -U $AMBARI-SERVER-USERNAME ambari > ambari.sql Password: $AMBARI-
SERVER-PASSWORD
pg_dump -U $MAPRED-USERNAME ambarirca > ambarirca.sql Password: $MAPRED-
PASSWORD
```

Substitute the values you set up at installation for `$AMBARI-SERVER-USERNAME`, `$MAPRED-USERNAME`, `$AMBARI-SERVER-PASSWORD`, and `$MAPRED-PASSWORD`. Defaults are `ambari-server/bigdata` and `mapred/mapred`.

3.2. Update Agents

1. On each agent host, stop the agent.

```
ambari-agent stop
```

2. Remove old agent certificates.

```
rm /var/lib/ambari-agent/keys/*
```

3. Using a text editor, edit `/etc/ambari-agent/conf/ambari-agent.ini` to point to the new host.

```
[server]
hostname=$NEW FULLY.QUALIFIED.DOMAIN.NAME
url_port=8440
secured_url_port=8441
```

3.3. Install the New Server and Populate the Databases

1. Install the Server on the new host.
2. Stop the Server so that you can copy the old database data to the new Server.

```
ambari-server stop
```

3. Restart the PostgreSQL instance.

```
service postgresql restart
```

4. Open the PostgreSQL interactive terminal.

```
su - postgres
psql
```

5. Using the interactive terminal, drop the databases created by the fresh install.

```
drop database ambari;
drop database ambarirca;
```

6. Check to make sure the databases have been dropped.

```
/list
```

The databases should not be listed.

7. Create new databases to hold the transferred data.

```
create database ambari;
create database ambarirca;
```

8. Exit the interactive terminal

```
^d
```

9. Copy the saved data from [Back up Current Data](#) to the new Server.

```
cd /tmp
scp -i <ssh-key> root@<original-server>/tmp/dbdumps/*.sql/tmp
(Note: compress/transfer/uncompress as needed from source to dest)
psql -d ambari -f /tmp/ambari.sql
psql -d ambarirca -f /tmp/ambarirca.sql
```

10. Start the new Server.

```
<exit to root>
ambari-server start
```

11. On each Agent host, start the Agent.


```
ambari-agent start
```

12. Open Ambari Web. Point your compatible browser to:

```
<new_Ambari_Server>:8080
```

13. Go to **Services** -> **MapReduce** and use the Management Header to **Stop** and **Start** the MapReduce service.

14. Start other services as necessary.

The new Server is ready to use.

4. Using Non-Default Databases

Use the following instructions to prepare a non-default database for Hive/HCatalog, Oozie, or Ambari. You must complete these instructions before you set up the Ambari Server by running `ambari-server setup`.

4.1. Hive/HCatalog

1. On the Hive Metastore machine, install the appropriate JDBC .jar file.

- For **Oracle**:

a. Download the Oracle JDBC (OJDBC) driver from <http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html>.

Select Oracle Database 11g Release 2 - ojdbc6.jar.

b. Copy the .jar file to the Java share directory.

```
cp ojdbc6.jar /usr/share/java
```

c. Make sure the .jar file has the appropriate permissions - 644.

- For **MySQL**:

a. Install the connector.

- RHEL/CentOS/Oracle Linux

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

- SLES

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

b. Confirm that MySQL .jar file is in the Java share directory

```
ls /usr/share/java/mysql-connector-java.jar
```

c. Make sure the .jar file has the appropriate permissions - 644.

2. On the Ambari Server host, install the appropriate JDBC .jar file.

- For **Oracle**:

a. Download the Oracle JDBC (OJDBC) driver from <http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html>.

Select Oracle Database 11g Release 2 - ojdbc6.jar.

b. Copy the .jar file to the Java share directory.

```
cp ojdbc6.jar /var/lib/ambari-server/resources
```

c. Make sure the .jar file has the appropriate permissions - 644.

- For **MySQL**:
 - a. Download the mysql connector driver from the host on which you installed mysql-connector-java.
 - b. Copy the .jar file to the Java share directory.

```
cp mysql-connector-java.jar /var/lib/ambari-server/resources
```

- c. Make sure the .jar file has the appropriate permissions - 644.

3. Create a user for Hive and grant it permissions.

- For **Oracle**, create the Hive user and grant it database permissions.

```
# sqlplus sys/root as sysdba
SQL> CREATE USER $HIVEUSER IDENTIFIED BY $HIVEPASSWORD;
SQL> GRANT SELECT_CATALOG_ROLE TO $HIVEUSER;
SQL> GRANT CONNECT, RESOURCE TO $HIVEUSER;
SQL> QUIT;
```

Where `$HIVEUSER` is the Hive user name and `$HIVEPASSWORD` is the Hive user password.

- For **MySQL**, create the Hive user and grant it database permissions.

```
# mysql -u root -p
mysql> CREATE USER '$HIVEUSER'@'%' IDENTIFIED BY '$HIVEPASSWORD';
mysql> GRANT ALL PRIVILEGES ON *.* TO '$HIVEUSER'@'%';
mysql> flush privileges;
```

Where `$HIVEUSER` is the Hive user name and `$HIVEPASSWORD` is the Hive user password.

4. For **Oracle** only Load the Hive Metastore Schema.

- The Hive Metastore database schema must be pre-loaded into your Oracle database using the schema script.

```
sqlplus $HIVEUSER/$HIVEPASSWORD < hive-schema-0.12.0.oracle.sql
```

The file `hive-schema-0.12.0.oracle.sql` is found in the `/var/lib/ambari-server/resources/` directory of the Ambari Server machine, once you have completed the step in the install process.

4.1.1. Troubleshooting Hive/HCatalog

Use these entries to help you troubleshoot any issues you might have installing Hive/HCatalog with non-default databases.

4.1.1.1. Problem: Hive Metastore Install Fails Using Oracle

Check the install log:

```
cp /usr/share/java/${jdbc_jar_name} ${target}] has failures: true
```

The Oracle JDBC .jar file cannot be found.

4.1.1.1.1. Soution

Make sure the file is in the appropriate directory on the Hive Metastore server and click **Retry**.

4.1.1.2. Problem: Install Warning when "Hive Check Execute" Fails Using Oracle

Check the install log:

```
java.sql.SQLException: ORA-01754:
a table may contain only one column of type LONG
```

The Hive Metastore schema was not properly loaded into the database.

4.1.1.2.1. Soution

Complete the install with the warning. Check your database to confirm the Hive Metastore schema is loaded. Once in the Ambari Web GUI, browse to **Services > Hive/HCat**. Use the Management Header to re-run the smoke test (**Maintenance ->Run Smoke Test**) to check that the schema is correctly in place.

4.1.1.3. Problem: Hive Check Execute may fail after completing an Ambari upgrade to version 1.4.2

For secure and non-secure clusters, with Hive security authorization enabled, the Hive service check may fail. Hive security authorization may not be configured properly.

4.1.1.3.1. Solution

Two workarounds are possible. Using Ambari Web, in **Hive Configs Advanced**:

- Disable `hive.security.authorization`, by setting the `hive.security.authorization.enabled` value to false.
- or
- Properly configure Hive security authorization. For example, set the following properties:

Table 4.1. Hive Security Authorization Settings

Property	Value
<code>hive.security.authorization.manager</code>	<code>org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationP</code>
<code>hive.security.metastore.authorization.manager</code>	<code>org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationP</code>
<code>hive.security.authenticator.manager</code>	<code>org.apache.hadoop.hive.ql.security.ProxyUserAuthenticator</code>

For more information about configuring Hive security, see [Metastore Server Security](#) in [Hive Authorization](#) and the HCatalog document [Storage Based Authorization](#).

4.2. Oozie

1. On the Oozie Server machine, install the appropriate JDBC .jar file.

- For **Oracle**:
 - a. Download the Oracle JDBC (OJDBC driver from <http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html>.

Select Oracle Database 11g Release 2 - ojdbc6.jar.
 - b. Copy the .jar file to the Java share directory.

```
cp ojdbc6.jar /usr/share/java
```
 - c. Make sure the .jar file has the appropriate permissions - 644.
- For **MySQL**:
 - a. Install the connector.
 - RHEL/CentOS/Oracle Linux

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

```
zypper install mysql-connector-java-5.0.8-1
```
 - b. Confirm that the MySQL .jar file is in the Java share directory.

```
ls /usr/share/java/mysql-connector-java.jar
```
 - c. Make sure the .jar file has the appropriate permissions - 644.
- 2. On the Ambari Server host, install the appropriate JDBC .jar file.
 - For **Oracle**:
 - a. Download the Oracle JDBC (OJDBC) driver from <http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html>.

Select Oracle Database 11g Release 2 - ojdbc6.jar .
 - b. Copy the .jar file to the Java share directory.

```
cp ojdbc6.jar /var/lib/ambari-server/resources
```
 - c. Make sure the .jar file has the appropriate permissions - 644.
 - For **MySQL**:
 - a. Download the mysql connector driver from the host on which you installed mysql-connector-java.
 - b. Copy the .jar file to the Java share directory.

```
cp mysql-connector-java.jar /var/lib/ambari-server/resources
```
 - c. Make sure the .jar file has the appropriate permissions - 644.

3. Create a user for Oozie and grant it permissions.

- For **Oracle**, create the Oozie user and grant it database permissions:

```
# sqlplus sys/root as sysdba
SQL> CREATE USER $OOZIEUSER IDENTIFIED BY $OOZIEPASSWORD;
SQL> GRANT ALL PRIVILEGES TO $OOZIEUSER;
SQL> QUIT;
```

Where `$OOZIEUSER` is the Oozie user name and `$OOZIEPASSWORD` is the Oozie user password.

- For **MySQL**:

a. Create the Oozie user and grant it database permissions.

```
# mysql -u root -p
mysql> CREATE USER '$OOZIEUSER'@'%' IDENTIFIED BY '$OOZIEPASSWORD';
mysql> GRANT ALL PRIVILEGES ON *.* TO '$OOZIEUSER'@'%';
mysql> flush privileges;
```

Where `$OOZIEUSER` is the Oozie user name and `$OOZIEPASSWORD` is the Oozie user password.

b. Create the Oozie database.

```
# mysql -u root -p
mysql> CREATE DATABASE oozie;
```

4.2.1. Troubleshooting Oozie

Use these entries to help you troubleshoot any issues you might have installing Oozie with non-default databases.

4.2.1.1. Problem: Oozie Server Install Fails Using MySQL

Check the install log:

```
cp /usr/share/java/mysql-connector-java.jar
/usr/lib/oozie/libext/mysql-connector-java.jar]
has failures: true
```

The MySQL JDBC .jar file cannot be found.

4.2.1.1.1. Soution

Make sure the file is in the appropriate directory on the Oozie server and click **Retry**.

4.2.1.2. Problem: Oozie Server Install Fails Using Oracle or MySQL

Check the install log:

```
Exec[exec cd /var/tmp/oozie &&
/usr/lib/oozie/bin/ooziedb.sh create -sqlfile oozie.sql -run ]
has failures: true
```

Oozie was unable to connect to the database or was unable to successfully setup the schema for Oozie.

4.2.1.2.1. Soution

Check the database connection settings provided during the **Customize Services** step in the install wizard by browsing back to **Customize Services** -> **Oozie**. After confirming and adjusting your database settings, proceed forward with the install wizard.

If the Install Oozie Server wizard continues to fail, get more information by connecting directly to the Oozie server and executing the following command as `$OOZIEUSER`:

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

4.3. Ambari

1. On the Ambari Server machine, install the appropriate .jar file.

- For **Oracle**:

a. Download the Oracle JDBC (OJDBC) driver from <http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html>.

b. Select Oracle Database 11g Release 2 - ojdbc6.jar

c. Copy the .jar file to the Java share directory.

```
cp ojdbc6.jar /usr/share/java
```

- For **MySQL**:

Install the MySQL connector.

```
yum install mysql-connector-java*
```

2. Make sure the .jar file has the appropriate permissions.

- For **Oracle**:

The .jar file should have 644 permissions.

- For **MySQL**:

```
cd /usr/share/java ls mysql*
```

Should return present and 644 permissions.

3. Create the Ambari user, password, and tablespace, and grant the account database permissions.

- For **Oracle**:

```
# sqlplus sys/root as sysdba
SQL> create user $AMBARIUSER identified by $AMBARIPASSWORD default
tablespace "USERS" temporary tablespace "TEMP";
SQL> grant unlimited tablespace to $AMBARIUSER;
SQL> grant create session to $AMBARIUSER;
```

```
SQL> grant create table to $AMBARIUSER;
SQL> quit;
```

Where `$AMBARIUSER` is the Ambari user name and `$AMBARIPASSWORD` is the Ambari user password.

- For **MySQL**, on the MySQL server host:

```
mysql -u root
CREATE USER '$AMBARIUSER'@'%' IDENTIFIED BY '$AMBARIPASSWORD';
GRANT ALL PRIVILEGES ON *.* TO '$AMBARIUSER'@'%';
CREATE USER '$AMBARIUSER'@'localhost' IDENTIFIED BY '$AMBARIPASSWORD';
GRANT ALL PRIVILEGES ON *.* TO '$AMBARIUSER'@'localhost';
CREATE USER '$AMBARIUSER'@'$AMBARISERVERFQDN' IDENTIFIED BY
'$AMBARIPASSWORD';
GRANT ALL PRIVILEGES ON *.* TO '$AMBARIUSER'@'$AMBARISERVERFQDN';
FLUSH PRIVILEGES;
```

Where `$AMBARISERVERFQDN` is the Fully Qualified Domain Name of the Ambari server host.

4. Load the database schema on the Ambari Server host.

- For **Oracle**:

Create the Ambari Server schema by running a script.

```
sqlplus $AMBARIUSER/$AMBARIPASSWORD <
/var/lib/ambari-server/resources/Ambari-DDL-Oracle-CREATE.sql
```

The file `Ambari-DDL-Oracle-CREATE.sql` is found in the `/var/lib/ambari-server/resources/` directory of the Ambari Server machine, once you have completed the step in the install process.

- For **MySQL**:

Create the Ambari Server database and schema.

```
mysql -u $AMBARIUSER -p
CREATE DATABASE $AMBARIDATABASE;
USE $AMBARIDATABASE;
SOURCE Ambari-DDL-MySQL-CREATE.sql;
```

Where `$AMBARIDATABASE` is the Ambari database name.

When setting up the Ambari server, select Advanced Database Configuration > option 3: MySQL.

On the Ambari server host, find the database schema in `/var/lib/ambari-server/resources/` after setting up the bits during the installation process.

4.3.1. Troubleshooting Ambari

Use these entries to help you troubleshoot any issues you might have installing Ambari with an existing Oracle database.

4.3.1.1. Problem: Ambari Server Fails to Start: No Driver

Check `/var/log/ambari-server/ambari-server.log` for the following error:

```
ExceptionDescription:Configurationerror.  
Class[oracle.jdbc.driver.OracleDriver] not found.
```

The Oracle JDBC .jar file cannot be found.

4.3.1.1.1. Soution

Make sure the file is in the appropriate directory on the Ambari server and re-run `ambari-server setup`. See [Step 1](#) above.

4.3.1.2. Problem: Ambari Server Fails to Start: No Connection

Check `/var/log/ambari-server/ambari-server.log` for the following error:

```
The Network Adapter could not establish the connection  
Error Code: 17002
```

Ambari Server cannot connect to the database.

4.3.1.2.1. Soution

Confirm the database host is reachable from the Ambari Server and is correctly configured by reading `/etc/ambari-server/conf/ambari.properties`.

```
server.jdbc.url=jdbc:oracle:thin:  
    @oracle.database.hostname:1521/ambaridb  
server.jdbc.rca.url=jdbc:oracle:thin:  
    @oracle.database.hostname:1521/ambaridb
```

4.3.1.3. Problem: Ambari Server Fails to Start: Bad Username

Check `/var/log/ambari-server/ambari-server.log` for the following error:

```
Internal Exception: java.sql.SQLException: ORA01017:  
invalid username/password; logon denied
```

You are using an invalid username/password.

4.3.1.3.1. Soution

Confirm the user account is set up in the database and has the correct privileges. See [Step 3](#) above.

4.3.1.4. Problem: Ambari Server Fails to Start: No Schema

Check `/var/log/ambari-server/ambari-server.log` for the following error:

```
Internal Exception: java.sql.SQLSyntaxErrorException: ORA00942:  
table or view does not exist
```

The schema has not been loaded.

4.3.1.4.1. Soution

Confirm you have loaded the database schema. See [Step 4](#) above.

5. Configuring Ambari Server for Internet Proxy

5.1. Setting up an Internet Proxy server for Ambari

If you plan to use the public repositories for installing the Stack, Ambari Server must have Internet access to confirm access to the repositories and validate the repositories. If your machine requires use of an Internet Proxy, you must configure Ambari Server to use the proxy.

1. On the Ambari Server, edit `/var/lib/ambari-server/ambari-env.sh`.
2. Add `"-Dhttp.proxyHost=myproxyhost -Dhttp.proxyPort=1234"` to the `AMBARI_JVM_ARGS`.
3. Restart the Ambari Server to pick up this change.



Note

If you plan to use local repositories, see [Optional: Configure Ambari for Local Repositories](#). Configuring Ambari to use a proxy server and have Internet access is not required. The Ambari Server must have access to your local repositories.

6. Changing the Ambari Server Port

6.1. Optional: Change the Default Ambari Server Port

By default Ambari uses port 8080 for access to Ambari Web and the REST API. If you want to change the port number, you need to edit the Ambari properties file.



Important

Ambari Server should not be running when you do this; either make the edits before you start Ambari Server the first time or bring the server down to make the edits.

1. On the Ambari Server host, open `/etc/ambari-server/conf/ambari.properties` with a text editor.
2. Add the client API port property and set it to your desired port value:

```
client.api.port=<port_number>
```

3. Start or re-start the Ambari Server. You can now access Ambari Web via the newly configured port:

```
http://{your.ambari.server}:<port_number>
```

7. Changing the JDK Version on an Existing Cluster

During your initial Ambari Server Setup, you selected the JDK to use or provided a path to a custom JDK already installed on your hosts.

Use the following procedure to change the JDK:

1. Re-run Ambari Server Setup.

```
ambari-server setup
```

2. At the prompt to change the JDK, Enter **y**.

```
Do you want to change Oracle JDK [y/n] (n)? y
```

3. At the prompt to choose a JDK, Enter **1** to change the JDK to v1.7.

```
[1] - Oracle JDK 1.7  
[2] - Oracle JDK 1.6  
[3] - Custom JDK  
Enter choice: 1
```

If you choose Oracle JDK 1.7 or Oracle JDK 1.6, the JDK you choose downloads and installs automatically.

4. If you choose Custom JDK, verify or add the custom JDK path on all hosts in the cluster.
5. After setup completes, you must restart each component for the new JDK to be used by the Hadoop services.

Using the Ambari Web UI, do one of the following:

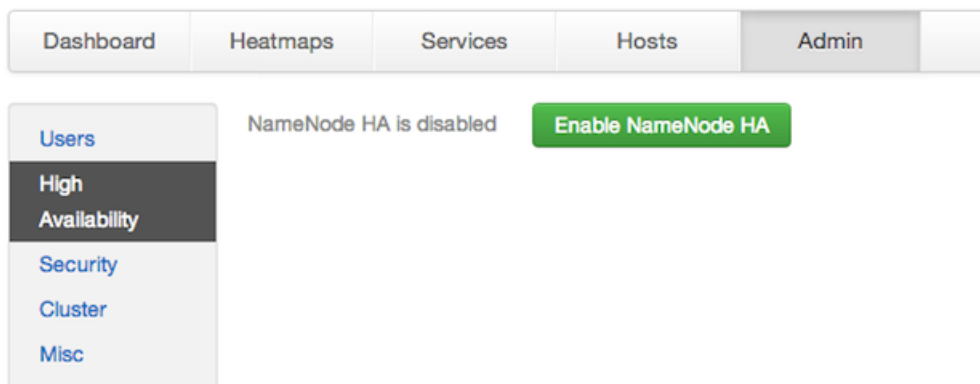
- Restart each component
- Restart each host
- Restart all services

8. NameNode High Availability

Use these instructions to set up NameNode HA using Ambari Web.

8.1. Setting Up NameNode High Availability

On Ambari Web, go to the Admin view. Select **High Availability** in the left navigation bar.



1. Check to make sure you have at least three hosts in your cluster and are running at least three ZooKeeper servers.
2. Click **Enable NameNode HA** and follow the **Enable NameNode HA Wizard**. The wizard describes a set of automated and manual steps you must take to set up NameNode high availability.
3. **Get Started**: This step gives you an overview of the process and allows you to select a Nameservice ID. You use this Nameservice ID instead of the NameNode FQDN once HA has been set up. Click **Next** to proceed.

Enable NameNode HA Wizard

ENABLE NAMEDNODE HA WIZARD

Get Started

Select Hosts

Review

Create Checkpoint

Configure Components

Initialize JournalNodes

Start Components

Initialize Metadata

Finalize HA Setup

Get Started

This wizard will walk you through enabling NameNode HA on your cluster. Once enabled, you will be running a Standby NameNode in addition to your Active NameNode. This allows for an Active-Standby NameNode configuration that automatically performs failover.

The process to enable HA involves a combination of **automated steps** (that will be handled by the wizard) and **manual steps** (that you must perform in sequence as instructed by the wizard).

You should plan a cluster maintenance window and prepare for cluster downtime when enabling NameNode HA.

If you have HBase running, please exit this wizard and stop HBase first.

Nameservice ID:

[Next →](#)

- Select Hosts:** Select a host for the additional NameNode and the JournalNodes. The wizard suggests options that you can adjust using the dropdown lists. Click **Next** to proceed.

Select Hosts

Select a host that will be running the additional NameNode. In addition, select the hosts to run JournalNodes, which store NameNode edit logs in a fault tolerant manner.

Current NameNode:

Additional NameNode:

JournalNode:

JournalNode:

JournalNode:

c6401.ambari.apache.org (1.8 GB, 1 cores)

NameNode HBase Master ZooKeeper

JournalNode

c6402.ambari.apache.org (1.8 GB, 1 cores)

SNameNode History Server

ResourceManager App Timeline Server

Nagios Server Ganglia Server

HiveServer2 Hive Metastore

WebHCat Server Oozie Server

ZooKeeper Falcon Server Nimbus

Storm UI Server Logviewer Server

DRPC Server Storm REST API Server

JournalNode NameNode

c6403.ambari.apache.org (1.8 GB, 1 cores)

ZooKeeper **JournalNode**

[← Back](#)
[Next →](#)

- Review:** Confirm your host selections and click **Next**.

Review

Confirm your host selections.

Current NameNode:	c6401.ambari.apache.org
Secondary NameNode:	c6402.ambari.apache.org - TO BE DELETED
Additional NameNode:	c6402.ambari.apache.org + TO BE INSTALLED
JournalNode:	c6401.ambari.apache.org + TO BE INSTALLED
	c6402.ambari.apache.org + TO BE INSTALLED
	c6403.ambari.apache.org + TO BE INSTALLED

Review Configuration Changes.
The following lists the configuration changes that will be made by the Wizard to enable NameNode HA. This information is for **review only** and is not editable except for the `dfs.journalnode.edits.dir` property

- ▶ **HDFS**
- ▶ **HBase**

← Back Next →

6. **Create Checkpoints:** Follow the instructions in the step. You need to login to your **current** NameNode host to run the commands to put your NameNode into safe mode and create a checkpoint. When Ambari detects success, the message on the bottom of the window changes. Click **Next**.

Manual Steps Required: Create Checkpoint on NameNode

1. Login to the NameNode host `c6401.ambari.apache.org`.
2. Put the NameNode in Safe Mode (read-only mode):

```
sudo su -l hdfs -c 'hdfs dfsadmin -safemode enter'
```
3. Once in Safe Mode, create a Checkpoint:

```
sudo su -l hdfs -c 'hdfs dfsadmin -saveNamespace'
```
4. You will be able to proceed once Ambari detects that the NameNode is in Safe Mode and the Checkpoint has been created successfully.

If the **Next** button is enabled before you run the **"Step 3: Create a Checkpoint"** command, it means there is a recent Checkpoint already and you may proceed without running the **"Step 3: Create a Checkpoint"** command.

Checkpoint created Next →

7. **Configure Components:** The wizard configures your components, displaying progress bars to let you track the steps. Click **Next** to continue.

Configure Components

Please proceed to the next step.

- ✔ Stop All Services
- ✔ Install Additional NameNode
- ✔ Install JournalNodes
- ✔ Reconfigure HDFS
- ✔ Start JournalNodes
- ✔ Disable Secondary NameNode

Next

8. **Initialize JournalNodes:** Follow the instructions in the step. You need to login to your **current** NameNode host to run the command to initialize the JournalNodes. When Ambari detects success, the message on the bottom of the window changes. Click **Next**.

Manual Steps Required: Initialize JournalNodes

1. Login to the NameNode host `c6401.ambari.apache.org`.
2. Initialize the JournalNodes by running:

```
sudo su -l hdfs -c 'hdfs namenode -initializeSharedEdits'
```
3. You will be able to proceed once Ambari detects that the JournalNodes have been initialized successfully.

JournalNodes initialized **Next →**

9. **Start Components:** The wizard starts the ZooKeeper servers and the NameNode, displaying progress bars to let you track the steps. Click **Next** to continue.

Start Components

Please proceed to the next step.

- ✔ Start ZooKeeper Servers
- ✔ Start NameNode

Next

10. **Initialize Metadata:** Follow the instructions in the step. For this step you must login to both the **current** NameNode and the **additional** NameNode. Make sure you are logged into the correct host for each command. Click **Next** when you have completed the two commands. A **Confirmation** popup appears to remind you that you must do both steps. Click **OK** to confirm.

Manual Steps Required: Initialize NameNode HA Metadata

1. Login to the NameNode host `c6401.ambari.apache.org`.
2. Initialize the metadata for NameNode automatic failover by running:


```
sudo su -l hdfs -c 'hdfs zkfc -formatZK'
```
3. Login to the Additional NameNode host `c6402.ambari.apache.org`.

Important! Be sure to login to the Additional NameNode host. This is a different host from the Steps 1 and 2 above.
4. Initialize the metadata for the Additional NameNode by running:


```
sudo su -l hdfs -c 'hdfs namenode -bootstrapStandby'
```

Please proceed once you have completed the steps above.

[Next →](#)

11 Finalize HA Setup: The wizard the setup, displaying progress bars to let you track the steps. Click **Done** to finish the wizard. After the Ambari Web GUI reloads, you may see some alert notifications. Wait a few minutes until the services come back up. If necessary, restart any components using Ambari Web.

Finalize HA Setup

Please wait while the wizard finalizes the HA setup.

- ✓ Start Additional NameNode
- ✓ Install Failover Controllers
- ✓ Start Failover Controllers
- ✓ Reconfigure HBase
- ✓ Delete Secondary NameNode
- ⚙ Start All Services 72%

[Done](#)



Note

Choose Services, then start Nagios, after completing all steps in the HA wizard.

12.If you are using Hive, you must manually change the Hive Metastore FS root to point to the Nameservice URI instead of the NameNode URI. You created the Nameservice ID in the [Get Started](#) step.

a. Check the current FS root. On the Hive host:

```
hive --config /etc/hive/conf.server --service metatool -listFSRoot
```

The output might look like this:

```
Listing FS Roots..
hdfs://<namenode-host>:8020/apps/hive/warehouse
```

- b. Use this command to change the FS root:

```
$ hive --config /etc/hive/conf.server --service metatool -updateLocation
<new-location> <old-location>

For example, where the Nameservice ID is mycluster:
$ hive --config /etc/hive/conf.server --service metatool -updateLocation
hdfs://mycluster:8020/apps/hive/warehouse hdfs://c6401.ambari.apache.
org:8020/apps/hive/warehouse
```

The output might look like this:

```
Successfully updated the following locations..
Updated X records in SDS table
```

- 13.If you are using Oozie, you must use the Nameservice URI instead of the NameNode URI in your workflow files. For example, where the Nameservice ID is mycluster:

```
<workflow-app xmlns="uri:oozie:workflow:0.2" name="map-reduce-wf">
  <start to="mr-node"/>
  <action name="mr-node">
    <map-reduce>
      <job-tracker>${jobTracker}</job-tracker>
      <name-node>hdfs://mycluster</name-node>
```

- 14.If you are using Hue, to enable NameNode High Availability, you must use httpfs instead of webhdfs to communicate with name nodes inside the cluster. After successfully setting up NameNode High Availability:

- a. Install an httpfs server on any node in the cluster:

```
yum install hadoop-httpfs
```

- b. Ensure that Hue hosts and groups use the httpfs server.

For example, on the httpfs server host, add to httpfs-site.xml the following lines:

```
<property> <name>httpfs.proxyuser.hue.hosts</name> <value>*</value> </
property>
<property> <name>httpfs.proxyuser.hue.groups</name> <value>*</value> </
property>
```

- c. Ensure that groups and hosts in the cluster use the httpfs server. For example, Using Ambari, in **Services > HDFS > Configs** add to core-site.xml the following properties and values:

Table 8.1. Core-site.xml properties and values for NameNode HA on a cluster using Hue

Property	Value
hadoop.proxyuser.httpfs.groups	*
hadoop.proxyuser.httpfs.hosts	*

- d. Using Ambari, in **Services > HDFS** restart the HDFS service in your cluster.

- e. On the Hue host, configure Hue to use the https server by editing hue.ini to include the following line:

```
webhdfs_url=http://{fqdn of https server}:14000/webhdfs/v1/
```

- f. Restart the Hue service.

8.2. Rolling Back NameNode HA

To roll back NameNode HA to the previous non-HA state use the following step-by-step manual process. Some of the steps are optional depending on your installation.

1. [Stop HBase](#)
2. [Checkpoint the Active NameNode](#)
3. [Stop All Services](#)
4. [Prepare the Ambari Server Host for Rollback](#)
5. [Restore the HBase Configuration](#)
6. [Delete ZK Failover Controllers](#)
7. [Modify HDFS Configurations](#)
8. [Recreate the Secondary NameNode](#)
9. [Re-enable the Secondary NameNode](#)
10. [Delete All JournalNodes](#)
11. [Delete the Additional NameNode](#)
12. [Verify the HDFS Components](#)
13. [Start HDFS](#)

8.2.1. Stop HBase

1. From Ambari Web, go to the Services view and select HBase.
2. Click **Stop** on the Management Header.
3. Wait until HBase has stopped completely before continuing.

8.2.2. Checkpoint the Active NameNode

If HDFS has been in use **after** you enabled NameNode HA, but you wish to revert back to a non-HA state, you must checkpoint HDFS state before proceeding with the rollback.

If the **Enable NameNode HA** wizard failed and you need to revert back, you can skip this step and move on to [Stop All Services](#).

- If Kerberos security has **not** been enabled on the cluster:

On the Active NameNode host, execute the following commands to save the namespace. You must be the HDFS service user (`$HDFS_USER`) to do this.

```
sudo su -l $HDFS_USER -c 'hdfs dfsadmin -safemode enter'
sudo su -l $HDFS_USER -c 'hdfs dfsadmin -saveNamespace'
```

- If Kerberos security has been enabled on the cluster:

```
sudo su -l $HDFS_USER -c 'kinit -kt /etc/security/keytabs/nn.service.keytab
nn/$HOSTNAME@$REALM;hdfs dfsadmin -safemode enter'
sudo su -l $HDFS_USER -c 'kinit -kt /etc/security/keytabs/nn.service.keytab
nn/$HOSTNAME@$REALM;hdfs dfsadmin -saveNamespace'
```

Where `$HDFS_USER` is the HDFS service user, `$HOSTNAME` is the Active NameNode hostname, and `$REALM` is your Kerberos realm.

8.2.3. Stop All Services

Use the Services view in Ambari Web and click **Stop All** in the Services navigation panel. You must wait until all the services are completely stopped.

8.2.4. Prepare the Ambari Server Host for Rollback

Log into the Ambari server host and set the following environment variables to prepare for the rollback procedure:

Table 8.2. Set Environment Variables

Variable	Value
<code>export AMBARI_USER=AMBARI_USERNAME</code>	Substitute the value of the administrative user for Ambari Web. The default value is <code>admin</code> .
<code>export AMBARI_PW=AMBARI_PASSWORD</code>	Substitute the value of the administrative password for Ambari Web. The default value is <code>admin</code> .
<code>export AMBARI_PORT=AMBARI_PORT</code>	Substitute the Ambari Web port. The default value is 8080.
<code>export AMBARI_PROTO=AMBARI_PROTOCOL</code>	Substitute the value of the protocol for connecting to Ambari Web. Options are <code>http</code> or <code>https</code> . The default value is <code>http</code> .
<code>export CLUSTER_NAME=CLUSTER_NAME</code>	Substitute the name of your cluster, set during the Ambari Install Wizard process. For example: <code>mycluster</code> .
<code>export NAMENODE_HOSTNAME=NN_HOSTNAME</code>	Substitute the FQDN of the host for the non-HA NameNode. For example: <code>nn01.mycompany.com</code> .
<code>export ADDITIONAL_NAMENODE_HOSTNAME=ANN_HOSTNAME</code>	Substitute the FQDN of the host for the additional NameNode in your HA setup.
<code>export SECONDARY_NAMENODE_HOSTNAME=SNN_HOSTNAME</code>	Substitute the FQDN of the host for the Secondary NameNode for the non-HA setup.
<code>export JOURNALNODE1_HOSTNAME=JOUR1_HOSTNAME</code>	Substitute the FQDN of the host for the first Journal Node.

Variable	Value
export JOURNALNODE2_HOSTNAME= <i>JOUR2_HOSTNAME</i>	Substitute the FQDN of the host for the second Journal Node.
export JOURNALNODE3_HOSTNAME= <i>JOUR3_HOSTNAME</i>	Substitute the FQDN of the host for the third Journal Node.



Important

Double check that these environment variables are set correctly.

8.2.5. Restore the HBase Configuration

If you have installed HBase, you may need to restore a configuration to its pre-HA state.

1. To check if your current HBase configuration needs to be restored, on the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME hbase-site
```

Where the environment variables you set up before substitute for the variable names.

Look for the configuration property `hbase.rootdir`. If the value is set to the NameService ID you set up using the **Enable NameNode HA** wizard, you need to revert the `hbase-site` configuration set up back to non-HA values. If it points instead to a specific NameNode host, it does not need to be rolled back and you can go on to [Delete ZK Failover Controllers](#).

For example:

```
"hbase.rootdir": "hdfs://name-service-id:8020/apps/hbase/data"
The hbase.rootdir property points to the NameService ID and the value needs to be rolled back

"hbase.rootdir": "hdfs://nn01.mycompany.com:8020/apps/hbase/data"
The hbase.rootdir property points to a specific NameNode host and not a NameService ID. This does not need to be rolled back.
```

2. If you need to roll back the `hbase.rootdir` value, on the Ambari Server host, use the `config.sh` script to make the necessary change:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p $AMBARI_PW -port $AMBARI_PORT set localhost $CLUSTER_NAME hbase-site hbase.rootdir hdfs://${NAMENODE_HOSTNAME}:8020/apps/hbase/data
```

Where the environment variables you set up before substitute for the variable names.

3. Verify that the `hbase.rootdir` property has been restored properly. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME hbase-site
```

The `hbase.rootdir` property should now be set to the NameNode hostname, not the NameService ID.

8.2.6. Delete ZK Failover Controllers

You may need to delete ZK Failover Controllers.

1. To check if you need to delete ZK Failover Controllers, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=ZKFC
```

If this returns an empty `items` array, you can go on to [Modify HDFS Configuration](#). Otherwise you must use the DELETE commands below.

2. To delete all ZK Failover Controllers, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}://
localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${NAMENODE_HOSTNAME}/host_components/ZKFC
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}://
localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${ADDITIONAL_NAMENODE_HOSTNAME}/host_components/ZKFC
```

3. Verify that the ZK Failover Controllers have been deleted. On the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=ZKFC
```

This command should return an empty `items` array.

8.2.7. Modify HDFS Configurations

You may need to modify your `hdfs-site` configuration and/or your `core-site` configuration.

1. To check if you need to modify your `hdfs-site` configuration, on the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
$AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME hdfs-site
```

If you see **any** of the following properties, you must delete them from your `hdfs-site` configuration.

- `dfs.nameservices`
- `dfs.client.failover.proxy.provider.${NAMESERVICE_ID}`
- `dfs.ha.namenodes.${NAMESERVICE_ID}`
- `dfs.ha.fencing.methods`
- `dfs.ha.automatic-failover.enabled`
- `dfs.namenode.http-address.${NAMESERVICE_ID}.nn1`
- `dfs.namenode.http-address.${NAMESERVICE_ID}.nn2`

- `dfs.namenode.rpc-address.${NAMESERVICE_ID}.nn1`
- `dfs.namenode.rpc-address.${NAMESERVICE_ID}.nn2`
- `dfs.namenode.shared.edits.dir`
- `dfs.journalnode.edits.dir`
- `dfs.journalnode.http-address`
- `dfs.journalnode.kerberos.internal.spnego.principal`
- `dfs.journalnode.kerberos.principal`
- `dfs.journalnode.keytab.file`

Where `${NAMESERVICE_ID}` is the NameService ID you created when you ran the **Enable NameNode HA** wizard.

2. To delete these properties, execute the following for **each property** you found. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p $AMBARI_PW -port $AMBARI_PORT delete localhost $CLUSTER_NAME hdfs-site property_name
```

Where you replace *property_name* with the name of **each** of the properties to be deleted.

3. Verify that all of the properties have been deleted. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME hdfs-site
```

None of the properties listed above should be present.

4. To check if you need to modify your `core-site` configuration, on the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME core-site
```

5. If you see the property `ha.zookeeper.quorum`, it must be deleted. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p $AMBARI_PW -port $AMBARI_PORT delete localhost $CLUSTER_NAME core-site ha.zookeeper.quorum
```

6. If the property `fs.defaultFS` is set to the NameService ID, it must be reverted back to its non-HA value. For example:

```
"fs.defaultFS" : "hdfs://name-service-id"
```

The property `fs.defaultFS` needs to be modified as it points to a NameService ID

```
"fs.defaultFS" : "hdfs://nn01.mycompany.com"
```


The property `fs.defaultFS` does not need to be changed as it points to a specific NameNode and not a NameService ID

- To revert the property `fs.defaultFS` to the NameNode host value, on the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
$AMBARI_PW -port $AMBARI_PORT set localhost $CLUSTER_NAME core-site fs.
defaultFS hdfs://${NAMENODE_HOSTNAME}
```

- Verify that the `core-site` properties are now properly set. On the Ambari Server host:

```
/var/lib/ambari-server/resources/scripts/configs.sh -u $AMBARI_USER -p
$AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME core-site
```

The property `fs.defaultFS` should be set to point to the NameNode host and the property `ha.zookeeper.quorum` should not be there.

8.2.8. Recreate the Secondary NameNode

You may need to recreate your Secondary NameNode.

- To check to see if you need to recreate the Secondary NameNode, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=SECONDARY_NAMENODE
```

If this returns an empty `items` array, you must recreate your Secondary NameNode. Otherwise you can go on to [Re-enable Secondary NameNode](#).

- Recreate your Secondary NameNode. On the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X POST -d '{"host_components" :
 [{"HostRoles": {"component_name": "SECONDARY_NAMENODE"}}] }'
${AMBARI_PROTO}://localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/
hosts?Hosts/host_name=${SECONDARY_NAMENODE_HOSTNAME}
```

- Verify that the Secondary NameNode now exists. On the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=SECONDARY_NAMENODE
```

This should return a non-empty `items` array containing the Secondary NameNode.

8.2.9. Re-enable the Secondary NameNode

To re-enable the Secondary NameNode, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X PUT -d '{"RequestInfo":
 {"context": "Enable Secondary NameNode"}, "Body": {"HostRoles":
 {"state": "INSTALLED"}}}' ${AMBARI_PROTO}://localhost:${AMBARI_PORT}/api/v1/
clusters/${CLUSTER_NAME}/hosts/${SECONDARY_NAMENODE_HOSTNAME}/host_components/
SECONDARY_NAMENODE
```

- If this returns 200, go to [Delete All JournalNodes](#).

- If this returns 202, wait a few minutes and run the following on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET "${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=SECONDARY_NAMENODE&fields=HostRoles/state"
```

When "state" : "INSTALLED" is in the response, go on to the next step.

8.2.10. Delete All JournalNodes

You may need to delete any JournalNodes.

1. To check to see if you need to delete JournalNodes, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=JOURNALNODE
```

If this returns an empty `items` array, you can go on to [Delete Additional NameNode](#). Otherwise you must delete the JournalNodes.

2. To delete the JournalNodes, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}://
localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${JOURNALNODE1_HOSTNAME}/host_components/JOURNALNODE
```

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}://
localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${JOURNALNODE2_HOSTNAME}/host_components/JOURNALNODE
```

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}://
localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/
${JOURNALNODE3_HOSTNAME}/host_components/JOURNALNODE
```

3. Verify that all the JournalNodes have been deleted. On the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=JOURNALNODE
```

This should return an empty `items` array.

8.2.11. Delete the Additional NameNode

You may need to delete your Additional NameNode.

1. To check to see if you need to delete your Additional NameNode, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:
${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/
component_name=NAMENODE
```

If the `items` array contains two NameNodes, the Additional NameNode must be deleted.

2. To delete the Additional NameNode that was set up for HA, on the Ambari Server host:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X DELETE ${AMBARI_PROTO}://localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/hosts/${ADDITIONAL_NAMENODE_HOSTNAME}/host_components/NAMENODE
```

3. Verify that the Additional NameNode has been deleted:

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -i -X GET ${AMBARI_PROTO}://localhost:${AMBARI_PORT}/api/v1/clusters/${CLUSTER_NAME}/host_components?HostRoles/component_name=NAMENODE
```

This should return an `items` array that shows only one NameNode.

8.2.12. Verify your HDFS Components

Make sure you have the correct components showing in HDFS.

1. In Ambari Web, go to the Services view and select HDFS from the Services navigation panel.
2. Check the Summary panel and make sure that the first three lines look like this:
 - NameNode
 - SNameNode
 - DataNodes

You should **not** see any line for JournalNodes.

8.2.13. Start HDFS

1. On the HDFS page of the Services view, click **Start** in the Management Header to start up HDFS. Wait until the service is fully started and has passed the service check.

If HDFS does not start, you may need to go through the previous steps again.
2. Start all the other services by using **Start All** in the Services navigation panel.

9. Configuring RHEL HA for Hadoop 1.x

Ambari supports High Availability of components such as NameNode or JobTracker in a HDP 1.x cluster running RHEL HA. After installing NameNode monitoring components on hosts in an HA cluster, as described in [HDP System Administration](#), configure Ambari to reassign any component on a failover host in the cluster, using the `host_relocate_component.py` script.

For example, if the host for the primary NameNode or JobTracker component fails, Ambari reassigns the primary NameNode or JobTracker component to the configured failover host, when you start or restart Ambari server.



Note

Make sure that NameNode is installed. For successful reassignment, a component must be in one of the following states:

- Maintenance (started or stopped)
- Unknown

9.1. Deploy the scripts

While the Ambari server and ambari agents are running on each host:

1. Download `relocate_host_component.py` from `/var/lib/ambari-server/resources/scripts` on the Ambari server to `/usr/bin/` on each failover host.
2. Download `hadoop.sh` from `/var/lib/ambari-server/resources/scripts` on the Ambari server and replace `hadoop.sh` in `/usr/share/cluster/` on each failover host.

9.2. Configure Ambari properties across the HA cluster

To enable Ambari to run `relocate_host_component.py`, edit the cluster configuration file on each failover host in the HA cluster, using a text editor.

In `/etc/cluster/cluster.conf`, set values for each of the following properties:

- `<server>=<ambari-hostname / ip>`
- `<port>=<8080>`
- `<protocol>=<http / https>`
- `<user>=<admin>`
- `<password>=<admin>`
- `<cluster>=<cluster-name>`

- `<output>=</var/log/ambari_relocate.log>`

For example, the Hadoop daemon section of cluster.conf on the NameNode localhost in an HA cluster will look like:

```
<hadoop
__independent_subtree="1" __max_restarts="10" __restart_expire_time="600"
name="NameNode Process"
daemon="namenode" boottime="10000" probetime="10000" stoptime="10000" url=
"http://10.0.0.30:50070/dfshealth.jsp"
pid="/var/run/hadoop/hdfs/hadoop-hdfs-namenode.pid" path="/"

ambariproperties="server=localhost,port=8080,protocol=http,user=admin,
password=admin,cluster=c1,output=/var/log/ambari_relocate.log"

/>
```

The relocate_host_component.py script reassigns components on failover of any host in the HA cluster, when you start or restart Ambari server.

9.3. Troubleshooting RHEL HA

1. Review errors in `/var/log/messages/`.
2. If the following error message appears:

```
abrtcd: Executable '/usr/bin/relocate_resources.py' doesn't belong to any
package and ProcessUnpackaged is set to 'no'
```

In `/etc/abrt/abrt-action-save-package-data.conf`, set `ProcessUnpackaged=Yes`.

3. If the scripts return Error status=exit code 3, make sure the following are true:
 - The ambari agent on the failover host is running.
 - Failover did not result from STOP HDFS or STOP NN/JT, using Ambari.

The following table lists and describes parameters for `relocate_host_components.py`.

Table 9.1. Parameter Options for relocate_host_components.py

Parameter	Value	Example	Description
-h,	na	-help	Display all parameter options.
-v,	na	-verbose	Increases output verbosity.
-s,	SERVER_HOSTNAME	-s-host=SERVER_HOSTNAME	Ambari server host name (FQDN)
-p,	SERVER_PORT,	-port=SERVER_PORT	Ambari server port. [default: 8080]
-r,	PROTOCOL,	-protocol=PROTOCOL	Protocol for communicating with Ambari server (http/https) [default: http]
-c,	CLUSTER_NAME,	-cluster-name=CLUSTER_NAME	Ambari cluster to operate on.
-e,	SERVICE_NAME,	-service-name=SERVICE_NAME	Ambari Service to which the component belongs.
-m,	COMPONENT_NAME	-component-name=COMPONENT_NAME	Ambari Service Component to operate on.

Parameter	Value	Example	Description
-n,	NEW_HOSTNAME,	-new-host=NEW_HOSTNAME	New host to relocate the component to.
-a,	ACTION,	-action=ACTION	Script action. [default: relocate]
-o,	FILE,	-output-file=FILE	Output file. [default: /temp/ambari_reinstall_probe.out]
-u,	USERNAME,	-username=USERNAME	Ambari server admin user. [default: admin]
-w,	PASSWORD,	-password=PASSWORD	Ambari server admin password.
-d,	COMPONENT_NAME,	start-component	Start the component after reassignment.