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.

Except where otherwise noted, this document is licensed under Creative Commons Attribution ShareAlike 3.0 License. http://creativecommons.org/licenses/by-sa/3.0/legalcode
Table of Contents

I. Setting Up Ambari ........................................................................................................ i
  1. Getting Ready ..................................................................................................... 2
    1.1. Understand the Basics .............................................................................. 2
    1.2. Meet Minimum System Requirements ....................................................... 3
      1.2.1. Hardware Recommendations ......................................................... 4
      1.2.2. Operating Systems Requirements ................................................... 4
      1.2.3. Browser Requirements .................................................................. 4
      1.2.4. Software Requirements .................................................................. 5
      1.2.5. JDK Requirements ......................................................................... 5
      1.2.6. Database Requirements ................................................................. 5
    1.3. Collect Information ................................................................................... 6
    1.4. Prepare the Environment .......................................................................... 6
      1.4.1. Check Existing Installs .................................................................... 6
      1.4.2. Set Up Password-less SSH ............................................................... 7
      1.4.3. Enable NTP on the Cluster and on the Browser Host ....................... 8
      1.4.4. Check DNS ..................................................................................... 8
      1.4.5. Configuring iptables ....................................................................... 9
      1.4.6. Disable SELinux and PackageKit and check the umask Value .......... 10
    1.5. Optional: Configure Ambari for Local Repositories .................................. 10
  2. Running the Ambari Installer ............................................................................. 13
    2.1. Set Up the Bits ....................................................................................... 13
      2.1.1. RHEL/CentOS/Oracle Linux 5.x ..................................................... 13
      2.1.2. RHEL/CentOS/Oracle Linux 6.x ..................................................... 14
      2.1.3. SLES 11 ........................................................................................ 14
    2.2. Set Up the Server ................................................................................... 14
      2.2.1. Setup Options .............................................................................. 15
    2.3. Optional: Set Up LDAP or Active Directory Authentication ....................... 16
    2.4. Optional: Set Up Security for Ambari ...................................................... 18
      2.4.1. Set Up HTTPS for Ambari Server ................................................... 18
      2.4.2. Set Up HTTPS for Ganglia ............................................................ 19
      2.4.3. Set Up HTTPS for Nagios .............................................................. 20
      2.4.4. Optional: Encrypt Database and LDAP Passwords ......................... 22
    2.5. Optional: Set Up Two-Way SSL Between Ambari Server and Ambari
      Agents .......................................................................................................... 24
    2.6. Optional: Change the Ambari Server Port ............................................... 24
    2.7. Start the Ambari Server .......................................................................... 24
  II. Deploying, Configuring, and Upgrading Ambari for Hadoop 2.x ....................... xxvi
  3. Installing, Configuring, and Deploying the Cluster for Hadoop 2.x .................... 27
    3.1. Log into Apache Ambari ......................................................................... 27
    3.2. Welcome ................................................................................................ 27
    3.3. Select Stack ............................................................................................ 27
    3.4. Install Options ........................................................................................ 27
    3.5. Confirm Hosts ....................................................................................... 28
    3.6. Choose Services ..................................................................................... 28
    3.7. Assign Masters ....................................................................................... 29
    3.8. Assign Slaves and Clients ........................................................................ 29
    3.9. Customize Services .................................................................................. 30
      3.9.1. Service Users and Groups ............................................................. 30
7.6. ZooKeeper Ports ................................................................. 60
7.7. WebHCat Port ................................................................. 61
7.8. Ganglia Ports ................................................................. 61
7.9. MySQL Port ................................................................. 62
7.10. Ambari Ports ............................................................... 62
7.11. Nagios Ports ............................................................... 62

8. Appendix: NameNode High Availability .................................................. 63
  8.1. Setting Up NameNode High Availability ........................................... 63
    8.1.1. Rolling Back NameNode HA ............................................. 69

III. Deploying, Configuring, and Upgrading Ambari for Hadoop 1.x ................. lxxvii
  9. Installing, Configuring, and Deploying the Cluster for Hadoop 1.x .......... 78
    9.1. Log into Apache Ambari ...................................................... 78
    9.2. Welcome ............................................................................ 78
    9.3. Select Stack ....................................................................... 78
    9.4. Install Options .................................................................... 78
    9.5. Confirm Hosts .................................................................... 79
    9.6. Choose Services .................................................................. 79
    9.7. Assign Masters ................................................................... 80
    9.8. Assign Slaves and Clients .................................................... 80
    9.9. Customize Services ............................................................. 81
        9.9.1. Service Users and Groups ............................................. 81
        9.9.2. Properties That Depend on Service Usernames/Groups .... 82
        9.9.3. Recommended Memory Configurations for the MapReduce
                Service ........................................................................ 83
    9.10. Review ............................................................................. 83
    9.11. Install, Start and Test .......................................................... 83
    9.12. Summary .......................................................................... 84

10. Troubleshooting Ambari Deployments ..................................................... 85
    10.1. Getting the Logs .................................................................. 85
    10.2. Quick Checks ..................................................................... 85
    10.3. Specific Issues ..................................................................... 85
        10.3.1. Problem: Browser crashed before Install Wizard completed .... 86
        10.3.2. Problem: Install Wizard reports that the cluster install has failed
                .................................................................................. 86
        10.3.3. Problem: “Unable to create new native thread” exceptions in
                HDFS DataNode logs or those of any system daemon .......... 87
        10.3.4. Problem: The “yum install ambari-server” Command Fails .... 87
        10.3.5. Problem: HDFS Smoke Test Fails .................................... 87
        10.3.6. Problem: The HCatalog Daemon Metastore Smoke Test Fails.... 88
        10.3.7. Problem: MySQL and Nagios fail to install on RightScale CentOS
                5 images on EC2 ................................................................ 89
        10.3.8. Problem: Trouble starting Ambari on system reboot .......... 89
        10.3.9. Problem: Metrics and Host information display incorrectly in
                Ambari Web ...................................................................... 89
        10.3.10. Problem: On SUSE 11 Ambari Agent crashes within the first 24
                hours ........................................................................... 90
        10.3.11. Problem: Attempting to Start HBase REST server causes either
                REST server or Ambari Web to fail .................................. 90
        10.3.12. Problem: Multiple Ambari Agent processes are running, causing
                re-register ..................................................................... 90
10.3.13. Problem: Some graphs do not show a complete hour of data until the cluster has been running for an hour ....................................... 90
10.3.14. Problem: After performing a cluster install the Nagios server is not started ............................................................... 91
10.3.15. Problem: A service with a customized service user is not appearing properly in Ambari Web ............................................. 91
10.3.16. Problem: Updated configuration changes are not pushed to client/gateway nodes ....................................................... 91

11. Appendix: Upgrading Ambari Server to 1.2.5 ............................................... 92
12. Appendix: Upgrading the HDP Stack to 1.3.3 ............................................... 95
   12.1. Preparing for the Upgrade ................................................................. 95
   12.2. Setting Up the Ambari Repository .................................................... 96
   12.3. Upgrading Ambari ........................................................................... 97
   12.4. Upgrading the Stack (from 1.2.* to 1.3.3) ....................................... 99
   12.5. Upgrading the Stack (from 1.3.0 to 1.3.3) ...................................... 103

13. Appendix: Configuring Ports ................................................................. 106
   13.1. HDFS Ports ................................................................................... 106
   13.2. MapReduce Ports ......................................................................... 107
   13.3. Hive Ports ..................................................................................... 107
   13.4. HBase Ports .................................................................................. 108
   13.5. ZooKeeper Ports .......................................................................... 108
   13.6. WebHCat Port ............................................................................... 109
   13.7. Ganglia Ports ............................................................................... 109
   13.8. MySQL Port ................................................................................. 109
   13.9. Ambari Ports ............................................................................... 110
   13.10. Nagios Ports ............................................................................... 110

IV. Additional Tasks with Ambari ..................................................................... cxi
   14.1. RHEL/CentOS/Oracle Linux 5.x and 6.x ......................................... 112
   14.2. SLES ............................................................................................. 112

15. Appendix: Using Custom Hostnames ......................................................... 113
16. Appendix: Upgrading Operating Systems on an Ambari-based Hadoop

Installation .............................................................................................. 114

17. Appendix: Moving the Ambari Server ...................................................... 115
   17.1. Back up Current Data ................................................................. 115
   17.2. Update Agents ........................................................................... 115
   17.3. Install the New Server and Populate the Databases ....................... 116

18. Appendix: Using Non-Default Databases ................................................. 118
   18.1. Hive/HCatalog ........................................................................... 118
         18.1.1. Troubleshooting Hive/HCatalog ........................................... 119
   18.2. Oozie ........................................................................................... 120
         18.2.1. Troubleshooting Oozie .......................................................... 122
   18.3. Ambari ......................................................................................... 122
         18.3.1. Troubleshooting Ambari ..................................................... 124

19. Setting Up Kerberos for Use with Ambari ............................................... 126
   19.1. Setting Up Kerberos for Hadoop 2.x ............................................. 126
         19.1.1. Preparing Kerberos ............................................................ 126
         19.1.2. Setting Up Hadoop Users .................................................... 135
         19.1.3. Enabling Kerberos Security ............................................... 137
   19.2. Setting Up Kerberos for Hadoop 1.x ............................................. 138
         19.2.1. Preparing Kerberos ............................................................ 139
19.2.2. Setting Up Hadoop Users ......................................................... 147
19.2.3. Enabling Kerberos Security ....................................................... 149
List of Tables

2.1. Download the repo ................................................................................................ 13
2.2. Ambari Server LDAP Properties ............................................................................... 16
3.1. Service Users ........................................................................................................... 30
3.2. Service Group .......................................................................................................... 31
3.3. HDFS Settings: Advanced ........................................................................................ 31
3.4. MapReduce Settings: Advanced .............................................................................. 31
6.1. Key Properties to Check ......................................................................................... 52
6.2. Properties to Modify .............................................................................................. 56
7.1. HDFS Ports ............................................................................................................. 58
7.2. MapReduce Ports ................................................................................................... 59
7.3. YARN Ports ............................................................................................................ 59
7.4. Hive Ports ............................................................................................................... 60
7.5. HBase Ports ............................................................................................................ 60
7.6. HBase Ports ............................................................................................................ 61
7.7. WebHCat Port ........................................................................................................ 61
7.8. Ganglia Ports .......................................................................................................... 61
7.9. MySQL Port ............................................................................................................ 62
7.10. Ambari Web ......................................................................................................... 62
7.11. Nagios .................................................................................................................. 62
8.1. Set Environment Variables ...................................................................................... 70
9.1. Service Users ........................................................................................................... 81
9.2. Service Group ......................................................................................................... 82
9.3. HDFS Settings: Advanced ........................................................................................ 82
9.4. MapReduce Settings: Advanced .............................................................................. 82
13.1. HDFS Ports .......................................................................................................... 106
13.2. MapReduce Ports .............................................................................................. 107
13.3. Hive Ports ............................................................................................................ 107
13.4. HBase Ports ........................................................................................................ 108
13.5. HBase Ports ........................................................................................................ 108
13.6. WebHCat Port .................................................................................................... 109
13.7. Ganglia Ports ...................................................................................................... 109
13.8. MySQL Port ........................................................................................................ 110
13.9. Ambari Web ....................................................................................................... 110
13.10. Ambari Web ..................................................................................................... 110
15.1. ambari-agent.ini .................................................................................................. 113
19.1. Kerberos terminology ......................................................................................... 127
19.2. Service Principals ............................................................................................... 130
19.3. Ambari Principals .............................................................................................. 131
19.4. Service Keytab File Names ................................................................................... 132
19.5. Kerberos terminology ......................................................................................... 140
19.6. Service Principals .............................................................................................. 142
19.7. Ambari Principals ............................................................................................... 143
19.8. Service Keytab File Names ................................................................................... 144
Part I. Setting Up Ambari

This section describes installing and preparing Ambari so that it can deploy and manage your Hadoop installation. It includes:

- Getting Ready
- Running the Ambari Installer
1. Getting Ready

This section describes the information and materials you need to get ready to install Hadoop using the Apache Ambari Install Wizard. **Apache Ambari** provides an end-to-end management and monitoring application for Apache Hadoop. With Ambari, you can deploy and operate a complete Hadoop stack using a graphical user interface (GUI), manage configuration changes, monitor services, and create alerts for all the nodes in your cluster from a central point.

- **Understand the Basics**
- **Meet Minimum System Requirements**
- **Collect Information**
- **Prepare the Environment**
- **Optional: Configure Ambari for Local Repositories**

1.1. Understand the Basics

The Hortonworks Data Platform consists of three layers of components. A coordinated and tested set of these components is sometimes referred to as the Stack.

- **Core Hadoop 1**: The basic components of Apache Hadoop version 1.x.
  - **Hadoop Distributed File System (HDFS)**: A special purpose file system designed to provide high-throughput access to data in a highly distributed environment.
  - **MapReduce**: A framework for performing high volume distributed data processing using the MapReduce programming paradigm.

- **Core Hadoop 2**: The basic components of Apache Hadoop version 2.x.
  - **Hadoop Distributed File System (HDFS)**: A special purpose file system designed to provide high-throughput access to data in a highly distributed environment.
  - **YARN**: A resource negotiator for managing high volume distributed data processing. Previously part of the first version of MapReduce.
  - **MapReduce 2 (MR2)**: A set of client libraries for computation using the MapReduce programming paradigm and a History Server for logging job and task information. Previously part of the first version of MapReduce.
  - **Essential Hadoop**: A set of Apache components designed to ease working with Core Hadoop.
  - **Apache Pig**: A platform for creating higher level data flow programs that can be compiled into sequences of MapReduce programs, using Pig Latin, the platform’s native language.
• **Apache Hive**: A tool for creating higher level SQL queries using HiveQL, the tool’s native language, that can be compiled into sequences of MapReduce programs. Included with **Apache HCatalog**.

• **Apache HCatalog**: A metadata abstraction layer that insulates users and scripts from how and where data is physically stored. Now part of **Apache Hive**. Includes **WebHCat**, which provides a set of REST APIs for HCatalog and related Hadoop components. Originally named **Templeton**.

• **Apache HBase**: A distributed, column-oriented database that provides the ability to access and manipulate data randomly in the context of the large blocks that make up HDFS.

• **Apache ZooKeeper**: A centralized tool for providing services to highly distributed systems. ZooKeeper is necessary for HBase installations.

• **Hadoop Support**: A set of components that allow you to monitor your Hadoop installation and to connect Hadoop with your larger compute environment.

• **Apache Oozie**: A server based workflow engine optimized for running workflows that execute Hadoop jobs.

  Running the current Oozie examples requires some reconfiguration from the standard Ambari installation. See Using HDP for Workflow and Scheduling (Oozie)

• **Apache Sqoop**: A component that provides a mechanism for moving data between Hadoop and external structured data stores. Can be integrated with Oozie workflows.

• **Apache Flume**: A log aggregator. This component must be installed manually. It is not supported in the context of Ambari at this time.

  See Installing and Configuring Flume in HDP for more information.

• **Ganglia**: An Open Source tool for monitoring high-performance computing systems.

• **Nagios**: An Open Source tool for monitoring systems, services, and networks.

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

1.2. **Meet Minimum System Requirements**

To run Hadoop, your system must meet minimum requirements.

• **Hardware Recommendations**

• **Operating Systems Requirements**

• **Browser Requirements**

• **Software Requirements**

• **JDK Requirements**
• Database Requirements

1.2.1. Hardware Recommendations

There is no single hardware requirement set for installing Hadoop.

For more information on the parameters that may affect your installation, see Hardware Recommendations For Apache Hadoop.

1.2.2. Operating Systems Requirements

The following operating systems are supported:

• Red Hat Enterprise Linux (RHEL) v5.x or 6.x (64-bit)
• CentOS v5.x or 6.x (64-bit)
• Oracle Linux v5.x or 6.x
• SUSE Linux Enterprise Server (SLES) 11, SP1 (64-bit)

Important

The installer pulls many packages from the base OS repos. If you do not have a complete set of base OS repos available to all your machines at the time of installation you may run into issues.

If you encounter problems with base OS repos being unavailable, please contact your system administrator to arrange for these additional repos to be proxied or mirrored. For more information see Optional: Configure the Local Repositories

1.2.3. Browser Requirements

The Ambari Install Wizard runs as a browser-based Web app. You must have a machine capable of running a graphical browser to use this tool. The supported browsers are:

• Windows (Vista, 7)
  • Internet Explorer 9.0 and higher (for Vista + Windows 7)
  • Firefox latest stable release
  • Safari latest stable release
  • Google Chrome latest stable release
• Mac OS X (10.6 or later)
  • Firefox latest stable release
  • Safari latest stable release
• Google Chrome latest stable release
• Linux (RHEL, CentOS, SLES, Oracle Linux)
• Firefox latest stable release
• Google Chrome latest stable release

1.2.4. Software Requirements

On each of your hosts:
• yum
• rpm
• scp
• curl
• wget

Important
The Python version shipped with SUSE 11, 2.6.0-8.12.2, has a critical bug that may cause the Ambari Agent to fail with 24 hours. If you are installing on SUSE 11, please update all your hosts to Python version 2.6.8-0.15.1.

1.2.5. JDK Requirements

The following Java runtimes are supported:
• Oracle JDK 1.6.0_31 64-bit
• Oracle JDK 1.7
• OpenJDK 7

1.2.6. Database Requirements

Hive/HCatalog, Oozie, and Ambari all require their own internal databases.

• Hive/HCatalog: By default uses an Ambari-installed MySQL 5.x instance. With appropriate preparation, you can also use an existing MySQL 5.x or Oracle 11g r2 instance. See Using Non-Default Databases for more information on using existing instances.

• Oozie: By default uses an Ambari-installed Derby instance. With appropriate preparation, you can also use an existing MySQL 5.x or Oracle 11g r2 instance. See Using Non-Default Databases for more information on using existing instances.

• Ambari: By default uses an Ambari-installed PostgreSQL 8.x instance. With appropriate preparation, you can also use an existing Oracle 11g r2 instance. See Using Non-Default Databases for more information on using existing instances.
1.3. Collect Information

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

- The fully qualified domain name (FQDN) for each host in your system, and which components you want to set up on which host. The Ambari install wizard does not support using IP addresses. You can use `hostname -f` to check for the FQDN if you do not know it.

**Note**

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

- The base directories you want to use as mount points for storing:
  - NameNode data
  - DataNodes data
  - Secondary NameNode data
  - Oozie data
  - MapReduce data (Hadoop version 1.x)
  - YARN data (Hadoop version 2.x)
  - ZooKeeper data, if you install ZooKeeper
  - Various log, pid, and db files, depending on your install type

1.4. Prepare the Environment

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

- Check Existing Installs
- Set up Password-less SSH
- Enable NTP on the Cluster
- Check DNS
- Configure iptables
- Disable SELinux, PackageKit and Check umask Value

1.4.1. Check Existing Installs

Ambari automatically installs the correct versions of the files that are necessary for Ambari and Hadoop to run. Versions other than the ones that Ambari uses can cause problems in running the installer, so remove any existing installs that do not match the following lists.
<table>
<thead>
<tr>
<th></th>
<th>RHEL/CentOS/Oracle Linux v5</th>
<th>RHEL/CentOS/Oracle Linux v6</th>
<th>SLES 11</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambari Server</td>
<td>• libffi 3.0.5-1.el5</td>
<td>• postgresql 8.4.13-1.el6_3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• python26 2.6.8-2.el5</td>
<td>• postgresql-libs 8.4.13-1.el6_3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• python26-libs 2.6.8-2.el5</td>
<td>• postgresql-server 8.4.13-1.el6_3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• postgresql 8.4.13-1.el6_3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• postgresql-libs 8.4.13-1.el6_3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• postgresql-server 8.4.13-1.el6_3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• libpq5 9.1.5-0.2.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ambari Agent [a]</td>
<td>None</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>Nagios Server [b]</td>
<td>• nagios 3.5.0-99</td>
<td>• nagios 3.5.0-99</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• nagios-devel 3.5.0-99</td>
<td>• nagios-devel 3.5.0-99</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• nagios-www 3.5.0-99</td>
<td>• nagios-www 3.5.0-99</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• nagios-plugins 1.4.9-1</td>
<td>• nagios-plugins 1.4.9-1</td>
<td></td>
</tr>
<tr>
<td>Ganglia Server [c]</td>
<td>• ganglia-gmetad 3.5.0-99</td>
<td>• ganglia-gmetad 3.5.0-99</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ganglia-devel 3.5.0-99</td>
<td>• ganglia-devel 3.5.0-99</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• libganglia 3.5.0-99</td>
<td>• libganglia 3.5.0-99</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ganglia-web 3.5.7-99</td>
<td>• ganglia-web 3.5.7-99</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• rrdtool 1.4.5-1.el5</td>
<td>• rrdtool 1.4.5-1.el6</td>
<td></td>
</tr>
<tr>
<td>Ganglia Monitor [d]</td>
<td>• ganglia-gmond 3.5.0-99</td>
<td>• ganglia-gmond 3.5.0-99</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• libganglia 3.5.0-99</td>
<td>• libganglia 3.5.0-99</td>
<td></td>
</tr>
</tbody>
</table>

[a] Installed on each host in your cluster. Communicates with the Ambari Server to execute commands.
[b] The host that runs the Nagios server
[c] The host that runs the Ganglia Server
[d] Installed on each host in the cluster. Sends metrics data to the Ganglia Collector.

### 1.4.2. Set Up Password-less SSH

To have Ambari Server automatically install Ambari Agents in all your cluster hosts, you must set up password-less SSH connections between the main installation (Ambari Server) host and all other machines. The Ambari Server host acts as the client and uses the key-pair to access the other hosts in the cluster to install the Ambari Agent.

**Note**

You can choose to install the Agents on each cluster host manually. In this case you do not need to setup SSH. See Appendix: Installing Ambari Agents Manually for more information.

1. Generate public and private SSH keys on the Ambari Server host
   ```bash
   ssh-keygen
   ```
2. Copy the SSH Public Key (id_rsa.pub) to the root account on your target hosts.
3. Add the SSH Public Key to the authorized_keys file on your target hosts.

```
cat id_rsa.pub >> authorized_keys
```

4. Depending on your version of SSH, you may need to set permissions on the .ssh directory (to 700) and the authorized_keys file in that directory (to 600) on the target hosts.

```
chmod 700 ~/.ssh
chmod 600 ~/.ssh /authorized_keys
```

5. From the Ambari Server, make sure you can connect to each host in the cluster using SSH.

```
ssh root@{remote.target.host}
```

You may see this warning. This happens on your first connection and is normal.

**Are you sure you want to continue connecting (yes/no)?**

6. Retain a copy of the SSH Private Key on the machine from which you will run the web-based Ambari Install Wizard.

**Note**

It is possible to use a non-root SSH account, if that account can execute `sudo` without entering a password.

1.4.3. Enable NTP on the Cluster and on the Browser Host

The clocks of all the nodes in your cluster and the machine that runs the browser through which you access Ambari Web must be able to synchronize with each other.

1.4.4. Check DNS

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

If you are unable to configure DNS and Reverse DNS, you must edit the hosts file on every host in your cluster to contain the address of each of your hosts and to set the Fully Qualified Domain Name hostmane of each of those hosts. The following instructions cover basic hostname network setup for generic Linux hosts. Different versions and flavors of Linux might require slightly different commands. Please refer to your specific operating system documentation for the specific details for your system.

1.4.4.1. Edit the Host File

1. Using a text editor, open the hosts file on every host in your cluster. For example:

```
vi /etc/hosts
```

2. Add a line for each host in your cluster. The line should consist of the IP address and the FQDN. For example:
1.2.3.4 fully.qualified.domain.name

**Note**

Do not remove the following two lines from your host file, or various programs that require network functionality may fail.

```
127.0.0.1 localhost.localdomain localhost
::1 localhost6.localdomain6 localhost6
```

### 1.4.4.2. Set the Hostname

1. Use the "hostname" command to set the hostname on each host in your cluster. For example:

```
hostname fully.qualified.domain.name
```

2. Confirm that the hostname is set by running the following command:

```
hostname -f
```

This should return the name you just set.

### 1.4.4.3. Edit the Network Configuration File

1. Using a text editor, open the network configuration file on every host. This file is used to set the desired network configuration for each host. For example:

```
vim /etc/sysconfig/network
```

2. Modify the HOSTNAME property to set the fully.qualified.domain.name.

```
NETWORKING=yes
NETWORKING_IPV6=yes
HOSTNAME=fully.qualified.domain.name
```

### 1.4.5. Configuring iptables

For Ambari to communicate during setup with the hosts it deploys to and manages, certain ports must be open and available. The easiest way to do this is to temporarily disable iptables.

```
chkconfig iptables off
/etc/init.d/iptables stop
```

You can restart `iptables` after setup is complete.

If the security protocols at your installation do not allow you to disable `iptables`, you can proceed with them on, as long as all of the relevant ports are open and available. If you plan to run with them enabled, see Configuring Ports (for the 1.x stack) or Configuring Ports (for the 2.x stack) for more information on the necessary ports per component.

During the Ambari Server setup process, Ambari checks to see if `iptables` is running. If it is, a warning prints to remind you to check that the necessary ports are open and available. The Host Confirm step of the Cluster Install Wizard will also issue a warning for each host that has `iptables` running.
Important

If you leave iptables enabled and do not set up the necessary ports, the cluster installation will fail.

1.4.6. Disable SELinux and PackageKit and check the umask Value

1. SELinux must be temporarily disabled for the Ambari setup to function. Run the following command on each host in your cluster:

   ```bash
   setenforce 0
   ```

2. On the RHEL/CentOS installation host, if PackageKit is installed, open `/etc/yum/pluginconf.d/refresh-packagekit.conf` with a text editor and make this change:

   ```bash
   enabled=0
   ```

   **Note**

   PackageKit is not enabled by default on SLES. Unless you have specifically enabled it, you do not need to do this step.

3. Make sure umask is set to 022.

1.5. Optional: Configure Ambari for Local Repositories

If your cluster does not have access to the Internet, or you are creating a large cluster and you want to conserve bandwidth, you need to provide access to the bits using an alternative method.

1. Set up the local mirror repositories as needed for HDP and HDP Utils.

   For more information on your options, see Deploying HDP In Production Data Centers with Firewalls.

2. From the Ambari Server host, fetch the Ambari repository file or RPM package as described in Set Up the Bits. You need a connection to the Internet for this step.

   If you do not have a connection to the Internet for this machine, you should follow the instructions in Deploying HDP In Production Data Centers with Firewalls and be sure to perform the optional steps for setting up the Ambari local repository.

3. Configure Ambari Server so that it knows to connect to the mirrored repositories during installation.

   a. On Ambari Server, browse to the stacks definitions directory

      ```bash
      cd /var/lib/ambari-server/resources/stacks
      ```
There are two stack definitions in this directory: HDP and HDPLocal. The HDP definition points to the publicly hosted HDP software packages. You must modify the HDPLocal definition to point to the local repositories you have set up.

b. Browse to the stack HDPLocal/$stack-version/repos directory. For example, for the 1.3.3 version of the stack:

```
cd HDPLocal/1.3.3/repos
```

c. Use a text editor to edit the repo info file. For example:

```
vi repoinfo.xml
```

d. You must update the `<baseurl>` value to point to your local repositories for each operating system that your cluster includes. So, for example, if your system includes hosts running CentOS 6, to point to the HDP 1.3.3 repositories, you would update stanzas to look something like this:

```xml
<os type="centos6">
  <repo>
    <baseurl>http://{your.hosted.local.repository}/HDP-1.3.3/repos/centos6</baseurl>
    <repoid>HDP-1.3.3</repoid>
    <reponame>HDP</reponame>
  </repo>
</os>
```

The appropriate relative path depends on how you have set up your local repos.

**Important**

If you have mixed operating systems in your cluster (for example, CentOS 6 and RHEL 6), you must configure the repositories and have properly edited `<os type>` stanzas for both OSes - centos6 and redhat6. If you do not, some hosts in your cluster will not be able to retrieve the software packages for their operating system.

e. Save this file.

f. Configure which JDK to use and how the JDK will be downloaded and installed.

- If you have not already installed the JDK on all hosts, and plan to use Oracle JDK 1.6, download `jdk-6u31-linux-x64.bin` to `/var/lib/ambari-server/resources`.

- If you plan to use a JDK other than Oracle JDK 1.6, you **must** install the JDK on each host in your cluster and use the `-j` flag when running `ambari-server setup`. See [JDK Requirements](#) for more information on supported JDKs.

- If you have already installed the JDK on all your hosts, you **must** use the `-j` flag when running `ambari-server setup`
Note

See Setup Options for more information on the -j flag.
2. Running the Ambari Installer

This section describes the process for installing Apache Ambari. Ambari manages installing and deploying Hadoop.

2.1. Set Up the Bits

1. Log into the machine that is to serve as the Ambari Server as root. You may login and sudo as su if this is what your environment requires. This machine is the main installation host.

2. Download the Ambari repository file and copy it to your repos.d.

<table>
<thead>
<tr>
<th>Table 2.1. Download the repo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform</td>
</tr>
<tr>
<td>RHEL, CentOS, and Oracle Linux 5</td>
</tr>
<tr>
<td>RHEL, CentOS and Oracle Linux 6</td>
</tr>
<tr>
<td>SLES 11</td>
</tr>
</tbody>
</table>

**Note**

If your cluster does not have access to the Internet, or you are creating a large cluster and you want to conserve bandwidth, you need to provide access to the bits using an alternative method. For more information, see Optional: Configure the Local Repositories section.

When you have the software, continue your installation based on your base platform.

2.1.1. RHEL/CentOS/Oracle Linux 5.x

Confirm the repository is configured by checking the repo list.

```
yum repolist
```

You should see the Ambari and HDP utilities repositories in the list. The version values vary depending the installation.

<table>
<thead>
<tr>
<th>repo id</th>
<th>repo name</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMBARI-1.x</td>
<td>Ambari 1.x</td>
</tr>
<tr>
<td>HDP-UTILS-1.1.0.16</td>
<td>Hortonworks Data Platform Utils</td>
</tr>
</tbody>
</table>

1. Install the Ambari bits using yum. This also installs PostgreSQL.

```
yum install ambari-server
```
2.1.2. RHEL/CentOS/Oracle Linux 6.x

1. Confirm the repository is configured by checking the repo list.

```bash
yum repolist
```

You should see the Ambari and HDP utilities repositories in the list. The version values vary depending the installation.

<table>
<thead>
<tr>
<th>repo id</th>
<th>repo name</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMBARI-1.x</td>
<td>Ambari 1.x</td>
</tr>
<tr>
<td>HDP-UTILS-1.1.0.16</td>
<td>Hortonworks Data Platform Utils</td>
</tr>
</tbody>
</table>

2. Install the Ambari bits using yum. This also installs PostgreSQL.

```bash
yum install ambari-server
```

2.1.3. SLES 11

1. Confirm the downloaded repository is configured by checking the repo list.

```bash
zypper repos
```

You should see the Ambari and HDP utilities in the list. The version values vary depending the installation.

<table>
<thead>
<tr>
<th>#</th>
<th>Alias</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AMBARI.dev-1.x</td>
<td>Ambari 1.x</td>
</tr>
<tr>
<td>2</td>
<td>HDP-UTILS-1.1.0.16</td>
<td>Hortonworks Data Platform Utils</td>
</tr>
</tbody>
</table>

2. Install the Ambari bits using zypper. This also installs PostgreSQL.

```bash
zypper install ambari-server
```

2.2. Set Up the Server

The `ambari-server` command manages the setup process. Follow the prompts:

```bash
ambari-server setup
```

1. If you have not temporarily disabled SELinux, you may get a warning. Enter y to continue.

2. By default, Ambari Server runs under root. If you want to create a different user to run the Ambari Server instead, or to assign a previously created user, select y at Customize user account for ambari-server daemon and give the prompt the username you want to use.

3. If you have not temporarily disabled iptables you may get a warning. Enter y to continue. See Configuring Ports for (2.x) or (1.x) for more information on the ports that must be open and accessible.

4. Agree to the Oracle JDK license when asked. You must accept this license to be able to download the necessary JDK from Oracle. The JDK is installed during the deploy phase.
Note

By default, Ambari Server setup will download and install Oracle JDK 1.6. If you plan to download this JDK and install on all your hosts, or plan to use a different version of the JDK, skip this step and see Setup Options for more information.

5. At Enter advanced database configuration:

• To use the default PostgreSQL database, named ambari, with the default username and password (ambari/bigdata), enter n.

• To use an existing Oracle 11g r2 instance or to select your own database name, username and password for either database, enter y.

• Select the database you want to use and provide any information required by the prompts, including hostname, port, Service Name or SID, username, and password.

Important

If you are using an existing Oracle instance, you must have prepared the instance using the steps detailed in Using Non-Default Databases before running the installer.


2.2.1. Setup Options

The following table describes options frequently used for Ambari Server setup.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-j --java-home</td>
<td>Specifies the JAVA_HOME path to use on the Ambari Server and all hosts in the cluster. By default when you do not specify this option, Setup downloads the Oracle JDK 1.6 binary to /var/lib/ambari-server/resources on the Ambari Server and installs the JDK to /usr/jdk64. Use this option when you plan to use a JDK other than the default Oracle JDK 1.6. See JDK Requirements for more information on the supported JDKs. If you are using an alternate JDK, you must manually install the JDK on all hosts and specify the Java Home path during Ambari Server setup. This path must be valid on all hosts. For example. ambari-server setup -j /usr/java/default</td>
</tr>
<tr>
<td>-s --silent</td>
<td>Setup runs silently. Accepts all default prompt values.</td>
</tr>
<tr>
<td>-v --verbose</td>
<td>Prints verbose info and warning messages to the console during Setup.</td>
</tr>
<tr>
<td>-c --jce-policy</td>
<td>Use specified jce_policy. Must be valid on the Ambari Server host.</td>
</tr>
</tbody>
</table>
2.3. Optional: Set Up LDAP or Active Directory Authentication

By default Ambari uses an internal database as the user store for authentication and authorization. If you want to add LDAP or Active Directory (AD) external authentication in addition for Ambari Web, you need to collect the following information and run a special setup command. Ambari Server must not be running when you execute this command. An LDAP client must be installed on the Ambari Server host.

**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. The following table details the properties and values you need to know to set up LDAP authentication.

**Note**

If you are going to set `bindAnonymously` to false (the default), you need to make sure you have an LDAP Manager name and password set up. If you are going to use SSL, you need to make sure you have already set up your certificate and keys.

**Table 2.2. Ambari Server LDAP Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>authentication.ldap.primaryUrl</code></td>
<td>server:port</td>
<td>The hostname and port for the LDAP or AD server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example: my.ldap.server:389</td>
</tr>
<tr>
<td><code>authentication.ldap.secondaryUrl</code></td>
<td>server:port</td>
<td>The hostname and port for the secondary LDAP or AD server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example: my.secondary.ldap.server:389</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This is an optional value.</td>
</tr>
<tr>
<td><code>authentication.ldap.useSSL</code></td>
<td>true or false</td>
<td>If true, use SSL when connecting to the LDAP or AD server.</td>
</tr>
<tr>
<td><code>authentication.ldap.usernameAttribute</code></td>
<td>[LDAP attribute]</td>
<td>The attribute for username</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example: uid</td>
</tr>
<tr>
<td><code>authentication.ldap.baseDn</code></td>
<td>[Distinguished Name]</td>
<td>The root Distinguished Name to search in the directory for users.</td>
</tr>
<tr>
<td>Property</td>
<td>Values</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>authentication.ldap.bindAnonymously</td>
<td>true or false</td>
<td>If true, bind to the LDAP or AD server anonymously</td>
</tr>
<tr>
<td>authentication.ldap.managerDn</td>
<td>[Full Distinguished Name]</td>
<td>If Bind anonymous is set to false, the Distinguished Name (&quot;DN&quot;) for the manager. Example: uid=hdfs,ou=people,dc=hadoop, dc=apache, dc=org</td>
</tr>
<tr>
<td>authentication.ldap.managerPassword</td>
<td>[password]</td>
<td>If Bind anonymous is set to false, the password for the manager</td>
</tr>
</tbody>
</table>

2. Run the LDAP setup command and answer the prompts with the information you collected above:

    ambari-setup setup-ldap

   a. At the Primary URL* prompt, enter the server URL and port you collected above. Prompts marked with an asterisk are required values.

   b. At the Secondary URL prompt, enter the secondary server URL and port. This is optional value.

   c. At the Use SSL* prompt, enter your selection.

   d. At the User name attribute* prompt, enter your selection. The default value is uid.

   e. At the Base DN* prompt, enter your selection.

   f. At the Bind anonymously* prompt, enter your selection.

   g. At the Manager DN* prompt, enter your selection if you have have set bind.Anonymously to false.

   h. At the Enter the Manager Password*, enter the password for your LDAP manager.

   i. Review your settings and if they are correct, select y.

   j. Start or restart the Server

    ambari-server restart

Initially the users you have enabled all have Ambari User privileges. Ambari Users can read metrics, view service status and configuration, and browse job information. For these new users to be able to start or stop services, modify configurations, and run smoke tests, they need to be Admins. To make this change, use Ambari Web Admin -> Users -> Edit.
2.4. Optional: Set Up Security for Ambari

There are four ways you can increase the security settings for your Ambari server installation.

**Note**

If you plan to configure your cluster for Kerberos, you may use the option, **Setup Ambari kerberos JAAS configuration** option, which is described in Setting up Kerberos for Use with Ambari.

- Set Up HTTPS for Ambari Web
- Set Up HTTPS for Ganglia
- Set UP HTTPS for Nagios
- Encrypt Database and LDAP Passwords

2.4.1. Set Up HTTPS for Ambari Server

If you want to limit access to the Ambari Server to HTTPS connections, you need to provide a certificate. While it is possible to use a self-signed certificate for initial trials, they are not suitable for production environments. Once your certificate is in place, you must run a special setup command.

**Important**

Ambari Server should not be running when you do this. Either make these changes before you start Ambari the first time, or bring the server down before running the setup command.

1. Log into the Ambari Server host.
2. Locate your certificate. If you want to create a temporary self-signed certificate, use this as an example:

   ```
   openssl genrsa -out $wserver.key 2048
   openssl req -new -key $wserver.key -out $wserver.csr
   openssl x509 -req -days 365 -in $wserver.csr -signkey $wserver.key -out $wserver.crt
   ```

   Where $wserver is the Ambari Server hostname.

**Important**

The certificate you use must be PEM-encoded, not DER-encoded. If you attempt to use a DER-encoded certificate, you see this error:

```
unable to load certificate
14010976494024:error:0906D06C:PEM routines:PEM_read_bio:no start line: pem_lib.c
:698:Expecting: TRUSTED CERTIFICATE
```
You can convert a DER-encoded certificate to a PEM-encoded certificate using the following command:

```bash
openssl x509 -in cert.crt -inform der -outform pem -out cert.pem
```

where `cert.crt` is the DER-encoded certificate and `cert.pem` is the resulting PEM-encoded certificate.

3. Run the special setup command and answer the prompts

```bash
ambardi-server setup-security
```

a. Select 1 for **Enable HTTPS for Ambari server**.

b. Respond `y` to **Do you want to configure HTTPS?**

c. Select the port you want to use for SSL. Default is 8443.

d. Provide the path to your certificate and your private key.

e. Provide the password for the private key.

f. Start or restart the Server

```bash
ambardi-server restart
```

### 2.4.2. Set Up HTTPS for Ganglia

If you want Ganglia to use HTTPS instead of HTTP (the default) to communicate with Ambari Server, use the following instructions.

**Important**

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

1. Set up the Ganglia server.

a. Log into the Ganglia server host.

b. Create a self-signed certificate on the Ganglia server host. For example:

```bash
openssl genrsa -out $gserver.key 2048
openssl req -new -key $gserver.key -out $gserver.csr
openssl x509 -req -days 365 -in $gserver.csr -signkey $gserver.key -out $gserver.crt
```

Where `$gserver` is the Ganglia server hostname.

c. Install SSL on the Ganglia server host.

```bash
yum install mod_ssl
```

d. Edit the SSL configuration file on the Ganglia server host.
i. Using a text editor, open:

```
/etc/httpd/conf.d/ssl.conf
```

ii. Add lines setting the certificate and key file names to the files you created above [19]. For example:

```
SSLCertificateFile    = $gserver.crt
SSLCertificateKeyFile = $gserver.key
```

e. Disable HTTP access (optional)

i. Using a text editor, open:

```
/etc/httpd/conf/httpd.conf
```

ii. Comment out the port 80 listener:

```
# Listen 80
```

f. Restart the `httpd` service on the Ganglia server host.

```
service httpd restart
```

2. Set up and restart the Ambari Server.

a. Log into the Ambari Server.

b. Run the special setup command and answer the prompts.

```
ambari-server setup-security
```

i. Select 2 for **Enable HTTPS for Ganglia service**.

ii. Respond `y` to **Do you want to configure HTTPS for Ganglia service**.

iii. Enter your TrustStore type. Your options are jks, jceks, or pks12.

iv. Enter the path to your TrustStore file.

v. Enter the password for your TrustStore and then re-enter to confirm. The password must be at least 6 characters long.

vi. Enter the path to the Ganglia server certificate file.

c. Start or restart the Server

```
ambari-server restart
```

2.4.3. Set Up HTTPS for Nagios

If you want Nagios to use HTTPS instead of HTTP (the default), use the following instructions.
Important

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

1. Set up the Nagios server.

a. Log into the Nagios server host.

b. Create a self-signed certificate on the Nagios server host. For example:

```bash
openssl genrsa -out $nserver.key 2048
openssl req -new -key $nserver.key -out $nserver.csr
openssl x509 -req -days 365 -in $nserver.csr -signkey $nserver.key -out $nserver.crt
```

Where $nserver is the Nagios server hostname.

c. Install SSL on the Nagios server host.

```bash
yum install mod_ssl
```

d. Edit the SSL configuration file on the Nagios server host.

i. Using a text editor, open:

```
/etc/httpd/conf.d/ssl.conf
```

ii. Add lines setting the certificate and key file names to the files you created previously:[21]. For example:

```bash
SSLCertificateFile    = $nserver.crt
SSLCertificateKeyFile = $nserver.key
```

e. Disable HTTP access (optional)

i. Using a text editor, open:

```
/etc/httpd/conf/httpd.conf
```

ii. Comment out the port 80 listener:

```
# Listen 80
```

f. Restart the httpd service on the Nagios server host.

```bash
service httpd restart
```

2. Set up and restart the Ambari Server.

a. Log into the Ambari Server.

b. Run the special setup command and answer the prompts.

```bash
ambari-server setup-security
```

i. Select 2 for Enable HTTPS for Nagios service.
ii. Respond \texttt{y} to \textbf{Do you want to configure HTTPS for Nagios service}. \\

iii. Enter your TrustStore type. Your options are \texttt{jks}, \texttt{jceks}, or \texttt{pks12}. \\

iv. Enter the path to your TrustStore file. \\

v. Enter the password for your TrustStore and then re-enter to confirm. The password must be at least 6 characters long. \\

vi. Enter the path to the Nagios server certificate file. \\

c. Start or restart the Server

\texttt{ambari-server restart} \\

\textbf{2.4.4. Optional: Encrypt Database and LDAP Passwords} \\

By default the passwords for Ambari's database and for access to the LDAP server are stored in a plain text configuration file. To have those passwords encrypted, you need to run a special setup command. \\

\begin{itemize}
  \item \textbf{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, run the special setup command and answer the prompts:

\texttt{ambari-server setup-security} \\

a. Select 4 for \textbf{Encrypt passwords stored in ambari.properties file}. \\

b. Provide a master key for encrypting the passwords. You are prompted to enter the key twice for accuracy. \\

\begin{itemize}
  \item \textbf{Important}
    
    If your passwords are encrypted, you need access to the master key to start Ambari Server.
  
  c. You have three options for maintaining the master key:

  \begin{itemize}
    \item At the \texttt{Persist} prompt, select \texttt{y}. This stores the key in a file on the server.
    \item Create an environment variable \texttt{AMBARI_SECURITY_MASTER_KEY} and set it to the key.
    \item Provide the key manually at the prompt on server startup.
  \end{itemize}

\end{itemize}

d. Start or restart the Server

\texttt{ambari-server restart}
2.4.4.1. Reset Encryption

There may be situations in which you want to:

• **Remove encryption entirely**

• **Change the current master key**, either because the key has been forgotten or because you want to change the current key as a part of a security routine.

![Important]

**Important**

Ambari Server should not be running when you do this.

2.4.4.1.1. Remove Encryption Entirely

To reset Ambari database and LDAP passwords to a completely unencrypted state:

1. On the Ambari host, open `/etc/ambari-server/conf/ambari.properties` with a text editor and set this property

```
security.passwords.encryption.enabled=false
```

2. Delete `/var/lib/ambari-server/keys/credentials.jceks`

3. Delete `/var/lib/ambari-server/keys/master`

4. You must now reset the database password and, if necessary, the LDAP password. Run `ambari-server setup` and `ambari-server setup-ldap` again.

2.4.4.1.2. Change the Current Master Key

To change the master key:

• **If you know the current master key or if the current master key has been persisted:**

  1. Re-run the encryption setup command and follow the prompts.

```
ambari-server setup-security
```

  a. Select 4 for *Encrypt passwords stored in ambari.properties file.*

  b. Enter the current master key when prompted if necessary (if it is not persisted or set as an environment variable).

  c. At the *Do you want to reset Master Key* prompt, enter yes.

  d. At the prompt, enter the new master key and confirm.

• **If you do not know the current master key:**

  1. Remove encryption entirely, as described [here](#).

  2. Re-run `ambari-server setup-security` as described [here](#).

  3. Start or restart the Ambari Server.

```
ambari-server restart
```
2.5. Optional: Set Up Two-Way SSL Between Ambari Server and Ambari Agents

Two-way SSL provides a way to encrypt communication between Ambari Server and Ambari Agents. By default Ambari ships with Two-way SSL disabled. To enable Two-way SSL:

![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 following property:

```
security.server.two_way_ssl = true
```

3. Start or restart the Ambari Server.

```
ambari-server restart
```

The Agent certificates are downloaded automatically during Agent Registration.

2.6. Optional: Change the 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>
```

2.7. Start the Ambari Server

- To start the Ambari Server:
ambari-server start

• To check the Ambari Server processes:
  
  ps -ef | grep Ambari

• To stop the Ambari Server:
  
  ambari-server stop
Part II. Deploying, Configuring, and Upgrading Ambari for Hadoop 2.x

This section describes setting up Hadoop 2.x. It includes:

- Installing, Configuring, and Deploying the Cluster for Hadoop 2.x
- Troubleshooting Ambari Deployments for Hadoop 2.x
- Appendix: Upgrading Ambari Server to 1.4.1
- Appendix: Upgrading the HDP Stack from 1.3.2. or later to 2.0.6
- Appendix: Configuring Ports for Hadoop 2.x
3. Installing, Configuring, and Deploying the Cluster for Hadoop 2.x

This section describes using the Ambari install wizard in your browser to complete your installation, configuration and deployment of Hadoop.

3.1. Log into Apache Ambari

Once you have started the Ambari service, you can access the Ambari Install Wizard through your browser.

1. Point your browser to http://{main.install.hostname}:8080.

2. Log in to the Ambari Server using the default username/password: admin/admin. You can change this later to whatever you want.

3.2. Welcome

The first step creates the cluster name.

1. At the Welcome page, type a name for the cluster you want to create in the text box. No whitespaces or special characters can be used in the name.

2. Click the Next button.

3.3. Select Stack

The Service Stack (or simply the Stack) is a coordinated and tested set of Hadoop components. Use the radio button to select the Stack version you want to install. To install Hadoop 2.x, select 2.0.6.

3.4. Install Options

In order to build up the cluster, the install wizard needs to know general information about how you want to set it up. You need to supply the FQDN of each of your hosts. The wizard also needs to access the private key file you created in Set Up Password-less SSH. It uses these to locate all the hosts in the system and to access and interact with them securely.

1. Use the Target Hosts text box to enter your list of host names, one per line. You can use ranges inside brackets to indicate larger sets of hosts. For example, for host01.domain through host10.domain use host[01-10].domain.

   **Note**

   If you are deploying on EC2, use the internal Private DNS hostnames.

2. If you want to let Ambari automatically install the Ambari Agent on all your hosts using SSH, select Provide your SSH Private Key and either use the Choose File button in the
Host Registration Information section to find the private key file that matches the public key you installed earlier on all your hosts or cut and paste the key into the text box manually.

**Note**

If you are using IE 9, the Choose File button may not appear. Use the text box to cut and paste your private key manually.

Fill in the username for the SSH key you have selected. If you do not want to use root, you must provide the username for an account that can execute sudo without entering a password.

3. If you do not want Ambari to automatically install the Ambari Agents, select **Perform manual registration.** See Appendix: Installing Ambari Agents Manually for more information.

4. Advanced Options
   - If you want to use a local software repository (for example, if your installation does not have access to the Internet), check **Use a Local Software Repository.** For more information on using a local repository see Optional: Configure the Local Repositories

5. Click the Register and Confirm button to continue.

### 3.5. Confirm Hosts

This screen lets you confirm that Ambari has located the correct hosts for your cluster and to check those hosts to make sure they have the correct directories, packages, and processes to continue the install.

If any hosts were selected in error, you can remove them by selecting the appropriate checkboxes and clicking the grey Remove Selected button. To remove a single host, click the small white Remove button in the Action column.

At the bottom of the screen, you may notice a yellow box that indicates some warnings were encountered during the check process. For example, your host may have already had a copy of wget or curl. Click Click here to see the warnings to see a list of what was checked and what caused the warning. On the same page you can get access to a python script that can help you clear any issues you may encounter and let you run Rerun Checks.

When you are satisfied with the list of hosts, click Next.

### 3.6. Choose Services

Hortonworks Data Platform is made up of a number of services. You must at a minimum install HDFS, but you can decide which of the other services you want to install. See Understand the Basics for more information on your options.

1. Select all to preselect all items or minimum to preselect only HDFS.

2. Use the checkboxes to unselect (if you have used all) or select (if you have used minimum) to arrive at your desired list of components.
Note

If you want to use Ambari for monitoring your cluster, make sure you select Nagios and Ganglia. If you do not select them, you get a warning popup when you finish this section. If you are using other monitoring tools, you can ignore the warning.

3. When you have made your selections, click Next.

3.7. Assign Masters

The Ambari install wizard attempts to assign the master nodes for various services you have selected to appropriate hosts in your cluster. The right column shows the current service assignments by host, with the hostname and its number of CPU cores and amount of RAM indicated.

1. To change locations, click the dropdown list next to the service in the left column and select the appropriate host.

2. To remove a ZooKeeper instance, click the green minus icon next to the host address you want to remove.

3. When you are satisfied with the assignments, click the Next button.

3.8. Assign Slaves and Clients

The Ambari install wizard attempts to assign the slave components (DataNodes, NodeManagers, and RegionServers) to appropriate hosts in your cluster. It also attempts to select hosts for installing the appropriate set of clients.

1. Use all or none to select all of the hosts in the column or none of the hosts, respectively.

   If a host has a red asterisk next to it, that host is also running one or more master components. Hover your mouse over the asterisk to see which master components are on that host.

2. Fine-tune your selections by using the checkboxes next to specific hosts.

Note

As an option you can start the HBase REST server manually after the install process is complete. It can be started on any host that has the HBase Master or the Region Server installed. If you attempt to start it on the same host as the Ambari server, however, you need to start it with the -p option, as its default port is 8080 and that conflicts with the Ambari Web default port.

```
/usr/lib/hbase/bin/hbase-daemon.sh start rest -p <custom_port_number>
```

3. When you are satisfied with your assignments, click the Next button.
3.9. Customize Services

The Customize Services screen presents you with a set of tabs that let you manage configuration settings for Hadoop components. The wizard attempts to set reasonable defaults for each of the options here, but you can use this set of tabs to tweak those settings, and you are strongly encouraged to do so, as your requirements may be slightly different. Pay particular attention to the directories suggested by the installer.

Hover over each of the properties to see a brief description of what it does. The number of tabs you see is based on the type of installation you have decided to do. In a complete installation there are ten groups of configuration properties and other related options, such as database settings for Hive/HCat and Oozie, and admin name/password and alert email for Nagios.

The install wizard sets reasonable defaults for all properties except for those related to databases in the Hive and the Oozie tabs, and two related properties in the Nagios tab. These four are marked in red and are the only ones you must set yourself.

Note

If you decide to use an existing database instance for Hive/HCatalog or for Oozie, you must have completed the preparations described in Using Non-Default Databases prior to running the install wizard.

Click the name of the group in each tab to expand and collapse the display.

3.9.1. Service Users and Groups

The individual services in Hadoop are each run under the ownership of a corresponding Unix account. These accounts are known as service users. These service users belong to a special Unix group. In addition there is a special service user for running smoke tests on components during installation and on-demand using the Management Header in the Services View of the Ambari Web GUI. Any of these users and groups can be customized using the Misc tab of the Customize Services step.

If you choose to customize names, Ambari checks to see if these custom accounts already exist. If they do not exist, Ambari creates them. The default accounts are always created during installation whether or not custom accounts are specified. These default accounts are not used and can be removed post-install.

Note

All new service user accounts, and any existing user accounts used as service users, must have a UID >= 1000.

Table 3.1. Service Users

<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Default User Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS</td>
<td>NameNode</td>
<td>hdfs</td>
</tr>
<tr>
<td></td>
<td>SecondaryNameNode</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DataNode</td>
<td></td>
</tr>
</tbody>
</table>
### Service Group

<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Default User Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>MapReduce2</td>
<td>HistoryServer</td>
<td>mapred</td>
</tr>
<tr>
<td>YARN</td>
<td>NodeManager</td>
<td>yarn</td>
</tr>
<tr>
<td></td>
<td>ResourceManager</td>
<td></td>
</tr>
<tr>
<td>Hive</td>
<td>Hive Metastore</td>
<td>hive</td>
</tr>
<tr>
<td></td>
<td>HiveServer2</td>
<td></td>
</tr>
<tr>
<td>HCat</td>
<td>HCatalog Server</td>
<td>hcat</td>
</tr>
<tr>
<td>WebHCat</td>
<td>WebHCat Server</td>
<td>hcat</td>
</tr>
<tr>
<td>Oozie</td>
<td>Oozie Server</td>
<td>oozie</td>
</tr>
<tr>
<td>HBase</td>
<td>MasterServer</td>
<td>hbase</td>
</tr>
<tr>
<td></td>
<td>RegionServer</td>
<td></td>
</tr>
<tr>
<td>ZooKeeper</td>
<td>ZooKeeper</td>
<td>zookeeper</td>
</tr>
<tr>
<td>Ganglia</td>
<td>Ganglia Server</td>
<td>nobody</td>
</tr>
<tr>
<td></td>
<td>Ganglia Collectors</td>
<td></td>
</tr>
<tr>
<td>Nagios</td>
<td>Nagios Server</td>
<td>nagios²</td>
</tr>
<tr>
<td>Smoke Test³</td>
<td>All</td>
<td>ambari-qa</td>
</tr>
</tbody>
</table>

³If you plan to use an existing user account named “nagios”, that “nagios” account must either be in a group named “nagios” or you must customize the Nagios Group.

²The Smoke Test user performs smoke tests against cluster services as part of the install process. It also can perform these on-demand from the Ambari Web GUI.

### 3.9.2. Properties That Depend on Service Usernames/Groups

Some properties must be set to match specific service usernames or service groups. If you have set up non-default, customized service usernames for the HDFS or HBase service or the Hadoop group name, you must edit the following properties:

#### Table 3.3. HDFS Settings: Advanced

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>dfs.permissions.superusergroup</td>
<td>The same as the HDFS username. The default is &quot;hdfs&quot;</td>
</tr>
<tr>
<td>dfs.cluster.administrators</td>
<td>A single space followed by the HDFS username.</td>
</tr>
<tr>
<td>dfs.block.local-path-access.user</td>
<td>The HBase username. The default is &quot;hbase&quot;.</td>
</tr>
</tbody>
</table>

#### Table 3.4. MapReduce Settings: Advanced

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapreduce.cluster.administrators</td>
<td>A single space followed by the Hadoop group name.</td>
</tr>
</tbody>
</table>
3.10. Review

The assignments you have made are displayed. Check to make sure everything is correct. If you need to make changes, use the left navigation bar to return to the appropriate screen.

To print your information for later reference, click Print.

When you are satisfied with your choices, click the Deploy button.

3.11. Install, Start and Test

The progress of the install is shown on the screen. Each component is installed and started and a simple test is run on the component. You are given an overall status on the process in the progress bar at the top of the screen and a host by host status in the main section.

To see specific information on what tasks have been completed per host, click the link in the Message column for the appropriate host. In the Tasks pop-up, click the individual task to see the related log files. You can select filter conditions by using the Show dropdown list. To see a larger version of the log contents, click the Open icon or to copy the contents to the clipboard, use the Copy icon.

Depending on which components you are installing, the entire process may take 40 or more minutes. Please be patient.

When Successfully installed and started the services appears, click Next.

3.12. Summary

The Summary page gives you a summary of the accomplished tasks. Click Complete. You are taken to the Ambari Web GUI.
4. Troubleshooting Ambari Deployments for Hadoop 2.x

The following information can help you troubleshoot issues you may run into with your Ambari-based installation.

4.1. Getting the Logs

The first thing to do if you run into trouble is to find the logs. Ambari Server logs are found at `/var/log/ambari-server/ambari-server.log` Ambari Agent logs are found at `/var/log/ambari-agent/ambari-agent.log`.

4.2. Quick Checks

- Make sure all the appropriate services are running. If you have access to Ambari Web, use the Services View to check the status of each component. If you do not have access to Manage Services, you must start and stop the services manually.

- If the first HDFS put command fails to replicate the block, the clocks in the nodes may not be synchronized. Make sure that Network Time Protocol (NTP) is enabled for your cluster.

- If HBase does not start, check if its slaves are running on 64-bit JVMs. Ambari requires that all hosts must run on 64-bit machines.

- Make sure umask is set to 0022.

- Make sure the HCatalog host can access the MySQL server. From a shell try:

  ```
  mysql -h $FQDN_for_MySQL_server -u $FQDN_for_HCatalog_Server -p
  ```

  You will need to provide the password you set up for Hive/HCatalog during the installation process.

- Make sure MySQL is running. By default, MySQL server does not start automatically on reboot.

  To set auto-start on boot, from a shell, type:

  ```
  chkconfig --level 35 mysql on
  ```

  To then start the service manually from a shell, type:

  ```
  service mysqld start
  ```

4.3. Specific Issues

The following are common issues you might encounter.
4.3.1. Problem: Browser crashed before Install Wizard completed

Your browser crashes or you accidently close your browser before the Install Wizard completes.

4.3.1.1. Solution

The response to a browser closure depends on where you are in the process:

- The browser closes prior to hitting the **Deploy** button.
  
  Re-launch the same browser and continue the install process. Using a different browser forces you to re-start the entire process.

- The browser closes after the **Deploy** button has launched the **Install, Start, and Test** screen.
  
  Re-launch the same browser and continue the process or use a different browser and re-login. You are returned to the **Install, Start, and Test** screen.

4.3.2. Problem: Install Wizard reports that the cluster install has failed

The Install, Start, and Test screen reports that the cluster install has failed.

4.3.2.1. Solution

The response to a report of install failure depends on the cause of the failure:

- The failure is due to intermittent network connection errors during software package installs.
  
  Use the **Retry** button on the **Install, Start, and Test** screen.

- The failure is due to misconfiguration or other setup errors.
  
  1. Use the left nav bar to go back to the appropriate screen; for example, **Customize Services**.
  2. Make your changes.
  3. Continue in the normal way.

- The failure occurs during the start/test sequence.
  
  1. Click **Next** and **Complete** and proceed to the Monitoring **Dashboard**.
  2. Use the **Services View** to make your changes.
  3. Re-start the service using the **Management Header**.
• The failure is due to something else.
1. Open an SSH connection to the Ambari Server host.
2. Clear the database. At the command line, type:
   
   ```
   ambari-server reset
   ```
3. Clear the browser's cache.
4. Re-run the entire Install Wizard.

### 4.3.3. Problem: “Unable to create new native thread” exceptions in HDFS DataNode logs or those of any system daemon

If your nproc limit is incorrectly configured, the smoke tests fail and you see an error similar to this in the DataNode logs:

```
INFO org.apache.hadoop.hdfs.DFSClient: Exception
increateBlockOutputStream java.io.EOFException
INFO org.apache.hadoop.hdfs.DFSClient: Abandoning block
blk_-6935524980745310745_139190
```

#### 4.3.3.1. Solution:
In certain recent Linux distributions (like RHEL/Centos/Oracle Linux 6.x), the default value of `nproc` is lower than the value required if you are deploying the HBase service. To change this value:

1. Using a text editor, open `/etc/security/limits.d/90-nproc.conf` and change the `nproc` limit to approximately 32000. For more information, see `ulimit` and `nproc` recommendations for HBase servers.
2. Restart the HBase server.

### 4.3.4. Problem: The “yum install ambari-server” Command Fails

You are unable to get the initial install command to run.

#### 4.3.4.1. Solution:
You may have incompatible versions of some software components in your environment. Check the list in Check Existing Installs and make any necessary changes. Also make sure you are running a Supported Operating System.

### 4.3.5. Problem: HDFS Smoke Test Fails

If your DataNodes are incorrectly configured, the smoke tests fail and you get this error message in the DataNode logs:
4.3.5.1. Solution:

- Make sure that reverse DNS look-up is properly configured for all nodes in your cluster.
- Make sure you have the correct FQDNs when specifying the hosts for your cluster. Do not use IP addresses - they are not supported.

Restart the installation process.

4.3.6. Problem: The HCatalog Daemon Metastore Smoke Test Fails

If the HCatalog smoke test fails, this is displayed in your console:

Metastore startup failed, see /var/log/hcatalog/hcat.err

4.3.6.1. Solution:

1. Log into the HCatalog node in your cluster

2. Open /var/log/hcatalog/hcat.err or /var/log/hive/hive.log (one of the two will exist depending on the installation) with a text editor

3. In the file, see if there is a MySQL Unknown Host Exception like this:

   at java.lang.reflect.Method.invoke (Method.java:597)
   at org.apache.hadoop.util.Runjar.main (runjar.java:156)
   Caused by: java.net.UnknownHostException:mysql.host.com
   at java.net.InetAddress.getAllByName(INetAddress.java:1157)

   This exception can be thrown if you are using a previously existing MySQL instance and you have incorrectly identified the hostname during the installation process. When you do the reinstall, make sure this name is correct.

4. In the file, see if there is an ERROR Failed initializing database entry like this:

   11/12/29 20:52:04 ERROR DataNucleus.Plugin: Bundle
   org.eclipse.jdt.core required

   This exception can be thrown if you are using a previously existing MySQL instance and you have incorrectly identified the username/password during the installation process. It can also occur when the user you specify does not have adequate privileges on the database. When you do the reinstall, make sure this username/password is correct and that the user has adequate privilege.

5. Restart the installation process.
4.3.7. Problem: MySQL and Nagios fail to install on RightScale CentOS 5 images on EC2

When using a RightScale CentOS 5 AMI on Amazon EC2, in certain cases MySQL and Nagios will fail to install. The MySQL failure is due to a conflict with the pre-installed MySQL and the use of the RightScale EPEL repository (error "Could not find package mysql-server"). Nagios fails to install due to conflicts with the RightScale php-common library.

4.3.7.1. Solution:

On the machines that will host MySQL and Nagios as part of your Hadoop cluster, perform the following:

1. Remove the existing MySQL server
   
   ```
   yum erase MySQL-server-community
   ```

2. Install MySQL server with a disabled RightScale EPEL repository

   ```
   yum install mysql-server --disable-repo=rightscale-epel
   ```

3. Remove the php-common library

   ```
   yum erase php-common-5.2.4-RightScale.x86
   ```

4.3.8. Problem: Trouble starting Ambari on system reboot

If you reboot your cluster, you must restart the Ambari Server and all the Ambari Agents manually.

4.3.8.1. Solution:

Log in to each machine in your cluster separately

1. On the Ambari Server host machine:

   ```
   ambari-server start
   ```

2. On each host in your cluster:

   ```
   ambari-agent start
   ```

4.3.9. Problem: Metrics and Host information display incorrectly in Ambari Web

Charts appear incorrectly or not at all despite being available in the native Ganglia interface or Host health status is displayed incorrectly.

4.3.9.1. Solution:

All the hosts in your cluster and the machine from which you browse to Ambari Web must be in sync with each other. The easiest way to assure this is to enable NTP.
4.3.10. Problem: On SUSE 11 Ambari Agent crashes within the first 24 hours

SUSE 11 ships with Python version 2.6.0-8.12.2 which contains a known bug that causes this crash.

4.3.10.1. Solution:

Upgrade to Python version 2.6.8-0.15.1

4.3.11. Problem: Attempting to Start HBase REST server causes either REST server or Ambari Web to fail

As an option you can start the HBase REST server manually after the install process is complete. It can be started on any host that has the HBase Master or the Region Server installed. If you install the REST server on the same host as the Ambari server, the http ports will conflict.

4.3.11.1. Solution

In starting the REST server, use the -p option to set a custom port. Use the following command to start the REST server.

```
/usr/lib/hbase/bin/hbase-daemon.sh start rest -p <custom_port_number>
```

4.3.12. Problem: Multiple Ambari Agent processes are running, causing re-register

On a cluster host `ps aux | grep ambari-agent` shows more than one agent process running. This causes Ambari Server to get incorrect ids from the host and forces Agent to restart and re-register.

4.3.12.1. Solution

On the affected host, kill the processes and restart.

1. Kill the Agent processes and remove the Agent PID files found here: `/var/run/ambari-agent/ambari-agent.pid`
2. Restart the Agent process:

```
ambari-agent start
```

4.3.13. Problem: Some graphs do not show a complete hour of data until the cluster has been running for an hour

When a cluster is first started, some graphs, like Services View -> HDFS and Services View -> MapReduce, do not plot a complete hour of data, instead showing data only for the
length of time the service has been running. Other graphs display the run of a complete hour.

4.3.13.1. Solution

Let the cluster run. After an hour all graphs will show a complete hour of data.

4.3.14. Problem: After performing a cluster install the Nagios server is not started

The Nagios server is not started after a cluster install and you are unable to manage it from Ambari Web.

4.3.14.1. Solution

1. Log into the Nagios server host.
2. Confirm that the Nagios server is not running. From a shell:
   
   ```
   ps -ef | grep nagios
   ```
   
   You should not see a Nagios process running.
3. Start the Nagios process manually. From a shell:
   
   ```
   service nagios start
   ```
4. The server starts. You should be able to see that started state reflected in Ambari Web. You can now manage (start/stop) Nagios from Ambari Web.

4.3.15. Problem: A service with a customized service user is not appearing properly in Ambari Web

You are unable to monitor or manage a service in Ambari Web when you have created a customized service user name with a hyphen, for example, `hdfs-user`.

4.3.15.1. Solution

Hyphenated service user names are not supported. You must re-run the Ambari Install Wizard and create a different name.

4.3.16. Problem: Updated configuration changes are not pushed to client/gateway nodes

Currently configuration changes are only pushed to daemon running nodes, so any changes are not automatically pushed to client only nodes such as gateway nodes.

4.3.16.1. Solution

Copy the files to the client nodes manually.
5. Appendix: Upgrading Ambari Server to 1.4.1

This process upgrades Ambari Server from version 1.2.5 to 1.4.1. To upgrade Ambari Server from previous versions, you must first upgrade to 1.2.5. Please see Upgrading Ambari Server to 1.2.5 for information on that process. This upgrade does not change the underlying Hadoop Stack. It is a twelve step manual process.

Note

You must know the location of the Nagios server for Step 9. Use the Services View -> Summary panel to locate the host on which it is running.

1. Stop the Ambari Server and all Ambari Agents. From the Ambari Server host:

   ```
   ambari-server stop
   ```

   From each Ambari Agent host:

   ```
   ambari-agent stop
   ```

2. Get the new Ambari bits. Use `wget` to fetch the repository file and replace the old repo file with the new repo file on every host.

   • Fetch the new repo file:

   - For RHEL/CentOS 5/Oracle Linux 5
     ```
     wget http://public-repo-1.hortonworks.com/ambari/centos5/1.x/updates/1.4.1.61/ambari.repo
     ```
   - For RHEL/CentOS 6/Oracle Linux 6
     ```
     wget http://public-repo-1.hortonworks.com/ambari/centos6/1.x/updates/1.4.1.61/ambari.repo
     ```
   - For SLES 11
     ```
     wget http://public-repo-1.hortonworks.com/ambari/suse11/1.x/updates/1.4.1.61/ambari.repo
     ```

   • Replace the old repo file with the new repo file.

   - For RHEL/CentOS 5/Oracle Linux 5
     ```
     cp ambari.repo /etc/yum.repos.d/ambari.repo
     ```

   Important

   Check your current directory before you download the new repo file to make sure that there are no previous versions of the 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.

   • Replace the old repo file with the new repo file.

   - For RHEL/CentOS 5/Oracle Linux 5
     ```
     cp ambari.repo /etc/yum.repos.d/ambari.repo
     ```
For RHEL/CentOS 6/Oracle Linux 6

```bash
cp ambari.repo /etc/yum.repos.d/ambari.repo
```

For SLES 11

```bash
cp ambari.repo /etc/zypp/repos.d/ambari.repo
```

**Note**

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

3. Upgrade Ambari Server.

**Note**

Ambari Server no longer automatically turns `iptables` off. Check your installation setup to make sure that you are not relying on this function. After you have upgraded the server you must either disable iptables manually or make sure that you have all the appropriate ports available. For more information on the ports that must be open and available, see Configuring Ports for Hadoop 2.x

From the Ambari Server host:

- **RHEL/CentOS**
  ```bash
  yum clean all
  yum upgrade ambari-server ambari-log4j
  ```

- **SLES**
  ```bash
  zypper clean
  zypper up ambari-server ambari-log4j
  ```

4. Check for upgrade success:

- As the process runs, the console should produce output similar, although not identical, to this:

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

  After the process is complete, check each host to make sure the new 1.4.1 files have been installed:

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

- If the upgrade fails, the console should produce output similar to this:
Setting up Upgrade Process
No Packages marked for Update

5. Check to see if you have a folder named `/etc/ambari-server/conf.save`. If you do, rename it back:

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

6. Upgrade the Ambari Server schema. From the Ambari Server host:

   `ambari-server upgrade`

7. Upgrade the Ambari Agent on all hosts. From each Ambari Agent host:

   • RHEL/CentOS
     
     `yum upgrade ambari-agent ambari-log4j`

   • SLES
     
     `zypper up ambari-agent ambari-log4j`

   **Note**

   If you get a warning that begins "There are some running programs that use files deleted by recent upgrade" you can ignore it.

8. Check to see if you have a file named `/etc/ambari-agent/conf.save` on each Agent host. If you do, rename it back. On each Agent host:

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

9. Upgrade the Nagios addons package. On the Nagios host:

   • RHEL/CentOS
     
     `yum upgrade hdp_mon_nagios_addons`

   • SLES
     
     `zypper up hdp_mon_nagios_addons`

10. Start the Server and the Agents on all hosts. From the Server host:

    `ambari-server start`

    From each Agent host:

    `ambari-agent start`

11. Open Ambari Web. Point your browser to `http://{your.ambari.server}:8080`

    **Important**

    You need to refresh your browser so that it loads the new version of the code. Hold the Shift key down while clicking the refresh button on the
browser. If you have problems, clear your browser cache manually and restart Ambari Server.

Use the Admin name and password you have set up to log in.

12. Re-start Nagios service. In **Ambari Web**.
   a. Go to the **Services View** and select each service.
   b. Use the **Management Header** to stop and re-start.
6. Appendix: Upgrading the HDP Stack from 1.3.2 or later to 2.0.6

The stack is the coordinated set of Hadoop components that you have installed. If you have a current instance of the 1.3.2 or later stack that was installed and managed by Ambari that you want to upgrade to the new 2.0.6 version of the stack and to also upgrade to the 1.4.1 version of Ambari Server and Agents, use the following instructions. This insures that the upgraded stack can still be managed by Ambari.

Note

If you have turned on Security (Kerberos) for your installation, you should turn it off before the upgrade. On Ambari Web->Admin view->Security-> click Disable Security. You can re-enable after the upgrade.

If you are upgrading from any other 1.x version of the stack, you must upgrade to 1.3.2 or later before you can upgrade to 2.0.6. See Upgrading the HDP Stack to 1.3.3 for more information. Upgrades from previous versions of 2.x are not supported.

Note

If you have already upgraded to Ambari Server 1.4.1 and just want to upgrade the HDP stack, you can skip Section 2 and Section 3.

6.1. Preparing for the Upgrade

Use the following steps to prepare your system for the upgrade.

1. If you are upgrading Ambari as well as the stack, you must know the location of the Nagios servers for that process. Use the Services->Nagios-> Summary panel to locate the hosts on which they are running.

2. Use the Services view on the Ambari Web UI to stop all services, including all clients, running on HDFS. Do not stop HDFS yet.

3. Finalize any prior upgrade if you have not done so already.

```
su $HDFSUSER
hadoop namenode -finalize
```

4. Create the following logs and other files.

Because the upgrade to 2.0.6 includes a version upgrade of HDFS, creating these logs allows you to check the integrity of the file system post upgrade.

a. 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.

```
su $HDFS_USER
```
hadoop fsck / -files -blocks -locations > /tmp/dfs-old-fsck-1.log

where $HDFS_USER is the HDFS Service user (by default, hdfs).

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

   su $HDFS_USER
   hadoop dfs -lsr / > /tmp/dfs-old-lsr-1.log

where $HDFS_USER is the HDFS Service user (by default, hdfs).

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

   su $HDFS_USER
   hadoop dfsadmin -report > /tmp/dfs-old-report-1.log

where $HDFS_USER is the HDFS Service user (by default, hdfs).

d. Optional: copy all or unrecoverable only data stored in HDFS to a local file system or to a backup instance of HDFS.

e. Optional: create the logs again and check to make sure the results are identical.

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

   Important

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

   su $HDFS_USER
   hadoop dfsadmin -safemode enter
   hadoop dfsadmin -saveNamespace

6. Copy the following checkpoint files into a backup directory. You can find the directory by using the Services View in the UI. Select the HDFS service, the Configs tab, in the Namenode section, look up the property NameNode Directories. It will be on your NameNode host.

   • dfs.name.dir/edits
   • dfs.name.dir/image/fsimage
   • dfs.name.dir/current/fsimage

7. On the JobTracker host, copy /etc/hadoop/conf to a backup directory.

   Note

   If you have deployed a custom version of capacity-scheduler.xml and mapred-queue-acls.xml, after the upgrade you will need to use Ambari Web to edit the default Capacity Scheduler. Select Services view ->YARN-Configs-Scheduler-Capacity Scheduler.
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. Make sure all services in the cluster are completely stopped.

10. If you are upgrading Hive, back up the Hive database.

11. Stop Ambari Server. On the Server host:

```
ambari-server stop
```

12. Stop Ambari Agents. On each host:

```
ambari-agent stop
```

### 6.2. Setting Up the Ambari Repository

This process prepares the updated repository.

1. Check to see if you have a conf.save directory for Ambari server and agents. If you do, move them to a back-up location:

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

2. Get the new Ambari bits. Use wget to fetch the repository file and replace the old repo file with the new repo file on every host.

**Important**

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

- For RHEL/CentOS/Oracle Linux 5

```
wget http://public-repo-1.hortonworks.com/ambari/centos5/1.x/updates/1.4.1.61/ambari.repo
cp ambari.repo /etc/yum.repos.d/ambari.repo
```

- For RHEL/CentOS/Oracle Linux 6

```
wget http://public-repo-1.hortonworks.com/ambari/centos6/1.x/updates/1.4.1.61/ambari.repo
cp ambari.repo /etc/yum.repos.d/ambari.repo
```

- For SLES 11

```
wget http://public-repo-1.hortonworks.com/ambari/suse11/1.x/updates/1.4.1.61/ambari.repo
cp ambari.repo /etc/zypp/repos.d/ambari.repo
```
Note

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

6.3. Upgrading to Ambari 1.4.1

This process upgrades Ambari Server, Ambari Agents, Ganglia, and Nagios.

Note

Ambari Server no longer automatically turns `iptables` off. Check your installation setup to make sure that you are not relying on this function. After you have upgraded the server you must either disable iptables manually or make sure that you have all the appropriate ports available. For more information on the ports that must be open and available, see Configuring Ports for Hadoop 2.x

1. Upgrade Ambari Server. From the Ambari Server host:

   • RHEL/CentOS/Oracle Linux
     ```
     yum clean all
     yum upgrade ambari-server ambari-log4j
     ```

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

2. Check for upgrade success:

   • As the process runs, the console should produce output similar, although not identical, to this:

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

     After the process is complete, check each host to make sure the new 1.4.1 files have been installed:

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

   • If the upgrade fails, the console should produce output similar to this:

     Setting up Upgrade Process
     No Packages marked for Update

3. Check to see if you have a folder named `/etc/ambari-server/conf.save`. If you do, rename it back:
mv /etc/ambari-server/conf.save /etc/ambari-server/conf

4. Upgrade the Ambari Server schema. From the Ambari Server host:

   `ambari-server upgrade`

5. Upgrade the Ambari Agent on all hosts. From each Ambari Agent host:

   - RHEL/CentOS/Oracle Linux
     
     `yum upgrade ambari-agent ambari-log4j`
   
   - SLES
     
     `zypper up ambari-agent ambari-log4j`

   **Note**
   
   If you get a warning that begins "There are some running programs that use files deleted by recent upgrade" you can ignore it.

6. Check to see if you have a folder named `/etc/ambari-agent/conf.save` on each Agent host. If you do, rename it back. On each Agent host:

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

7. Upgrade the Nagios addons:

   - RHEL/CentOS/Oracle Linux
     
     `yum upgrade hdp_mon_nagios_addons`
   
   - SLES
     
     `zypper up hdp_mon_nagios_addons`

**6.4. Upgrading the Stack**

This stack upgrade involves removing the HDP 1.x version of MapReduce and replacing it with the HDP 2.x YARN and MapReduce2 components. This process is somewhat long and complex. To help you, a Python script is provided to automate some of the upgrade steps. It is available at `/var/lib/ambari-server/resources/scripts/UpgradeHelper_HDP2.py` on the Ambari Server host. The script can be executed on any host that can communicate with Ambari Server. It requires Python 2.6 or higher.

**6.4.1. Prepare for the Stack Upgrade**

1. Make sure that you have saved the namespace. You should have done this here. The upgrade will fail if you do not save the namespace. If you have not saved the namespace yet:

   a. Restart Ambari Server and Ambari Agents.
   
   b. Restart HDFS only.
c. On the NameNode host:

```
su $HDFS_USER
hadoop dfsadmin -safemode enter
hadoop dfsadmin -saveNamespace
```

d. Stop the HDFS service and wait for it to be fully stopped.

e. Stop the Ambari Server and Ambari Agents.

2. Prepare for the upgrade:

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

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

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

```
python --version
```

**Note**

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

3. Start Ambari Server only. On the Ambari Server host:

```
ambari-server start
```

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

a. Go to the Upgrade Folder.

b. Execute the `backup-configs` action:

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

Where

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

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

c. Execute the `save-mr-mapping` action:

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

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

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

a. If you are not already there, go to the Upgrade Folder.

b. Execute the `delete-mr` action.

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

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

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

6.4.2. Upgrade the Stack

1. Stop Ambari Server. On the Ambari Server host:

```bash
ambari-server stop
```

2. Update the stack version in the Server database, depending on if you are using a local repository:

```bash
Important

Make sure you delete the old MapReduce version before you run upgradestack.

ambari-server upgradestack HDP-2.0.6
```

OR

```bash
ambari-server upgradestack HDPLocal-2.0.6
```

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

```bash
Important

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

- For RHEL/CentOS/Oracle Linux 5
wget http://public-repo-1.hortonworks.com/HDP/centos5/2.x/updates/2.0.6.0/hdp.repo
mv hdp.repo /etc/yum.repos.d/HDP.repo

• For RHEL/CentOS/Oracle Linux 6

wget http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.0.6.0/hdp.repo
mv hdp.repo /etc/yum.repos.d/HDP.repo

• For SLES 11

wget http://public-repo-1.hortonworks.com/HDP/suse11/2.x/updates/2.0.6.0/hdp.repo
mv hdp.repo /etc/zypp/repos.d/HDP.repo

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

• For RHEL/CentOS/Oracle Linux

  a. Remove remaining MapReduce components on all hosts:

     yum erase hadoop-pipes hadoop-sbin hadoop-native

  b. Upgrade the following components:

     yum upgrade "collectd*" "epel-release*" "gccxml*" "pig*" "hadoop*"
     "sqoop*" "zookeeper*" "hbase*" "hive*" "hcatalog*" "webhcat-tar*"
     "oozie*" hdp_mon_nagios_addons

  c. Check to see that the components in that list are upgraded.

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

     None of the components from that list should appear in the returned list.

• For SLES

  a. Remove remaining MapReduce components on all hosts:

     zypper remove hadoop-pipes hadoop-sbin hadoop-native

  b. Upgrade the following components:

     zypper up "collectd*" "epel-release*" "gccxml*" "pig*" "hadoop*"
     "sqoop*" "zookeeper*" "hbase*" "hive*" "hcatalog*" "webhcat-tar*"
     "oozie*" hdp_mon_nagios_addons

6.4.3. Add YARN/MR2 and Update Configurations

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

   ambari-server start

2. Start each Ambari Agent. On all Agent hosts:

   ambari-agent start
3. After the Server and all Agents are running, log into Ambari Web. Do a hard refresh on your browser to make sure you are displaying the updated GUI. Make sure all hosts are healthy and all services are in Stopped state.

4. Add YARN and MapReduce2 services:
   a. If you are not already there, go to the Upgrade Folder.
   b. Execute the `add-yarn-mr2` action:

   ```
   python UpgradeHelper_HDP2.py --hostname $HOSTNAME --user $USERNAME --password $PASSWORD --clustername $CLUSTERNAME add-yarn-mr2
   
   If desired, you can use the -n option to see the API calls as they are being made so that you can verify them.
   
   5. Update the respective configurations:
      a. If you are not already there, go to the Upgrade Folder.
      b. Execute the `update-configs` action:

      ```
      python UpgradeHelper_HDP2.py --hostname $HOSTNAME --user $USERNAME --password $PASSWORD --clustername $CLUSTERNAME update-configs
      
   6. Update individual configuration settings as needed. On the Ambari Server, use /var/lib/ambari-server/resources/scripts/configs.sh to inspect and update the configuration properties.
      a. Get configuration details:

      ```
      configs.sh get $HOSTNAME $CLUSTERNAME $CONFIGURATION-TYPE
      
      For example:
      configs.sh get localhost myclusternename global
      
      b. Evaluate each property value returned and modify as needed:

      ```
      configs.sh set $HOSTNAME $CLUSTERNAME $CONFIGURATION-TYPE "property name" "new value"
      
      For example:
      configs.sh set localhost myclusternename global yarn_log_dir_prefix "/apps/logs/yarn"
      
      c. Remove properties that are not needed:

      ```
      configs.sh delete $HOSTNAME $CLUSTERNAME $CONFIGURATION-TYPE "property name"
      
      For example:
      configs.sh delete localhost myclusternename global dfs.client-write-packet-size

      Table 6.1. Key Properties to Check
<pre><code>  | Configuration Type | Property            | Description                          |
  |--------------------|--------------------|--------------------------------------|
  | global             | yarn_log_dir_prefix| The location for the YARN logs       |
</code></pre>
<table>
<thead>
<tr>
<th>Configuration Type</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>global</td>
<td>yarn_pid_dir_prefix</td>
<td>The location for the YARN pid files</td>
</tr>
<tr>
<td>global</td>
<td>yarn_user</td>
<td>The YARN user</td>
</tr>
<tr>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>yarn-site</td>
<td>yarn.nodemanager.local-dirs</td>
<td>The location for container logs</td>
</tr>
<tr>
<td>yarn-site</td>
<td>yarn.nodemanager.log-dirs</td>
<td>The directories for localized files</td>
</tr>
</tbody>
</table>

**Note**

Make any necessary modifications **before** starting the services.

d. Install the YARN and MapReduce2 services:
   
i. If you are not already there, go to the Upgrade Folder.
   
ii. **Execute the** `install-yarn-mr` **action:**

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

   **Note**

   This is a two step process. You can use the Ambari Web GUI to monitor the progress. Both steps must be **complete** before you continue to the next step.

### 6.4.4. Complete the Stack Upgrade

1. Because the file system version has now changed you must start the NameNode manually. On the NameNode host:

   ```
   su -l $HDFS_USER -c "export HADOOP_LIBEXEC_DIR=/usr/lib/hadoop/libexec && /usr/lib/hadoop/sbin/hadoop-daemon.sh start namenode -upgrade"
   ```

   Depending on the size of your system, this step may take up to 10 minutes.

2. This upgrade can take a long time depending on the number of files you have. You can use `tail` to monitor the log file so that you can track your progress:

   ```
   tail -f /var/log/$HDFS_LOG_FOLDER/hadoop-hdfs-namenode-$HOSTNAME.log
   ```

   Look for lines that confirm the upgrade is complete, one line per name directory, such as Upgrade of /hadoop/hdfs/namenode is complete. You can also look for Registered FSNamesystem State MBean which follows the upgrade of all name directories.

3. Prepare the NameNode to work with Ambari:
   
a. Open the Ambari Web GUI. If it has been open throughout the process, do a hard reset on your browser to force a reload.
   
b. On the Services view, click HDFS to open the HDFS service.
   
c. Click **View Host** to open the NameNode host details page.
d. Use the dropdown menu to stop the NameNode.

e. On the Services view, restart the HDFS service. Make sure it passes the ServiceCheck. It is now under Ambari’s control.

4. After the DataNodes are started, HDFS exits safemode. To monitor the status:

```bash
sudo su -l $HDFS_USER -c "hdfs dfsadmin -safemode get"
```

Depending on the size of your system, this may take up to 10 minutes or so. When HDFS exits safemode, this is displayed as a response to the command:

**Safe mode is OFF**

5. Make sure that the HDFS upgrade was successful. Go through steps 2 and 3 in Section 9.1 to create new versions of the logs and reports. Substitute "new" for "old" in the file names as necessary

6. Compare the old and new versions of the following:

   • `dfs-old-fsck-1.log` versus `dfs-new-fsck-1.log`.
     
     The files should be identical unless the hadoop fsck reporting format has changed in the new version.

   • `dfs-old-lsr-1.log` versus `dfs-new-lsr-1.log`.
     
     The files should be identical unless the format of hadoop fs -lsr reporting or the data structures have changed in the new version.

   • `dfs-old-report-1.log` versus `fs-new-report-1.log`.
     
     Make sure all DataNodes previously belonging to the cluster are up and running.

7. Use the Ambari Web Services view to start YARN.

8. Use the Ambari Web Services view to start MapReduce2.

9. Upgrade HBase:

   a. Make sure that all HBase components - RegionServers and HBase Master - are stopped.

   b. Use the Ambari Web Services view, start the ZooKeeper service. Wait until the ZK service is up and running.

   c. On the HBase Master host, make these configuration changes:

      i. In HBASE_CONFDIR/hbase-site.xml, set the property dfs.client.read.shortcircuit to false.

      ii. In the configuration file, find the value of the hbase.tmp.dir property and make sure that the directory exists and is readable and writeable for the HBase service user and group.
iii. Go to the Upgrade Folder and check in the saved global configuration file named `global_<$TAG>` for the value of the property `hbase_pid_dir` and `hbase_log_dir`. Make sure that the directories are readable and writeable for the HBase service user and group.

```
chown -R $HBASE_USER:$HADOOP_GROUP $HBASE_TMP_DIR
```

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

iv. Upgrade HBase. You must be the HBase service user.

```
su $HBASE_USER
/usr/lib/hbase/bin/hbase upgrade -execute
```

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

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

10. Upgrade Oozie:

a. On the Services view, make sure YARN and MapReduce2 are running.

b. Make sure that the Oozie service is stopped.

c. Upgrade Oozie. You must be the Oozie service user. On the Oozie host:

```
su $OOZIE_USER
/usr/lib/oozie/bin/ooziedb.sh upgrade -run
```

Make sure that the output contains the string "Oozie DB has been upgrade to Oozie version 'OOZIE Build Version'".

d. Prepare the WAR file:

```
/usr/lib/oozie/bin/oozie-setup.sh prepare-war
```

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

e. Modify the following configuration properties in `oozie-site.xml`. On the Ambari Server, use `/var/lib/ambari-server/resources/scripts/configs.sh` to inspect and update the configuration properties as described here [52].
### Table 6.2. Properties to Modify

<table>
<thead>
<tr>
<th>Action</th>
<th>Property Name</th>
<th>Property Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>oozie.service.URIHandleService.uri.handlers</td>
<td>org.apache.oozie.dependency.FSURIHandler,org.apache.oozie.coordinatorFSURIHandler,org.apache.oozie.dependency.HCatURIHandler</td>
</tr>
<tr>
<td>Add</td>
<td>oozie.service.coord.push.check.requeue.interval</td>
<td>30000</td>
</tr>
<tr>
<td>Add</td>
<td>oozie.services.ext</td>
<td>org.apache.oozie.service.PartitionDependencyManagerService</td>
</tr>
<tr>
<td>Add/Modify</td>
<td>oozie.service.SchemaService.wf.ext.schemas</td>
<td>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</td>
</tr>
</tbody>
</table>

*Use this list if you have not modified the default Ambari values. If you have added custom schemas, make sure they exist after the modification. The schemas being added here are shell-action-0.2.xsd, oozie-sla-0.1.xsd, and oozie-sla-0.2.xsd. You can add these to your existing list.

---

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

i. Extract the Oozie sharelib into a tmp folder.

```bash
mkdir -p /tmp/oozie_tmp
cp /usr/lib/oozie/oozie-sharelib.tar.gz /tmp/oozie_tmp
cd /tmp/oozie_tmp
chown -R oozie:hadoop /tmp/oozie_tmp
tar xzvf oozie-sharelib.tar.gz
```

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

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

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

```bash
su -l hdfs -c "hdfs dfs -copyFromLocal /tmp/oozie_tmp/share /user/oozie/" 
su -l hdfs -c "hdfs dfs -chown -R oozie:hadoop /user/oozie" 
su -l hdfs -c "hdfs dfs -chmod -R 755 /user/oozie"
```

---

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

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

a. Make sure Ganglia is stopped.

b. Log into the host where JobTracker was installed (and where ResourceManager is installed after the upgrade).

c. Backup the folder `/etc/ganglia/hdp/HDPJobTracker`.

d. Remove the folder `/etc/ganglia/hdp/HDPJobTracker`. 

---

56
e. Remove the folder $ganglia_runtime_dir/HDPJobTracker.

Note

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

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

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

Important

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.

Note

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:

```
sudo su -l $HDFS_USER -c "hadoop dfsadmin -finalizeUpgrade"
```

where $HDFS_USER is the HDFS Service user (by default, hdfs).
7. Appendix: Configuring Ports for Hadoop 2.x

The tables below specify which ports must be opened for which ecosystem components to communicate with each other. Make sure the appropriate ports are opened before you install Hadoop.

- HDFS Ports
- MapReduce Ports
- YARN Ports
- Hive Ports
- HBase Ports
- ZooKeeper Ports
- WebHCat Port
- Ganglia Ports
- MySQL Port
- Ambari Ports
- Nagios Port

7.1. HDFS Ports

The following table lists the default ports used by the various HDFS services.

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode WebUI</td>
<td>Master Node hosts (NameNode and any back-up NameNodes)</td>
<td>50070</td>
<td>http</td>
<td>Web UI to look at current status of HDFS, explore file system</td>
<td>Yes (Typically admins, Dev/Support teams)</td>
<td>dfs.namnode.http-address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50470</td>
<td>https</td>
<td>Secure http service</td>
<td></td>
<td>dfs.namenode.https-address</td>
</tr>
<tr>
<td>NameNode metadata service</td>
<td>Master Node hosts (NameNode and any back-up NameNodes)</td>
<td>8020/9000</td>
<td>IPC</td>
<td>File system metadata operations</td>
<td>Yes (All clients who directly need to interact with the HDFS)</td>
<td>Embedded in URI specified by fs.defaultFS</td>
</tr>
<tr>
<td>DataNode</td>
<td>All Slave Node hosts</td>
<td>50075</td>
<td>http</td>
<td>DataNode WebUI to</td>
<td>Yes (Typically admins, Dev/Support teams)</td>
<td>dfs.datanode.http-address</td>
</tr>
</tbody>
</table>
### 7.2. MapReduce Ports

The following table lists the default port used by the History Server WebUI.

**Table 7.2. MapReduce Ports**

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>History Server WebUI</td>
<td></td>
<td>19888</td>
<td>http</td>
<td>Web UI for Job History</td>
<td>Yes</td>
<td>mapreduce.jobhistory.webapp</td>
</tr>
</tbody>
</table>

### 7.3. YARN Ports

The following table lists the default ports used by YARN.

**Table 7.3. YARN Ports**

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>YARN Resource Manager</td>
<td></td>
<td>8025</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>YARN RM Admin</td>
<td></td>
<td>8141</td>
<td></td>
<td>The address of interface</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Container Manager</td>
<td></td>
<td>0.0.0.0:45454</td>
<td></td>
<td>The address of manager in the interface</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applications Manager</td>
<td></td>
<td>8050</td>
<td></td>
<td>The address of applications manager in the interface</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 7.4. Hive Ports

The following table lists the default ports used by the various Hive services.

**Note**

Neither of these services is used in a standard HDP installation.
Table 7.4. Hive Ports

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hive Server2</td>
<td>Hive Server machine (Usually a utility machine)</td>
<td>10000</td>
<td>thrift</td>
<td>Service for programatically (Thrift/JDBC) connecting to Hive</td>
<td>Yes (Clients who need to connect to Hive either programatically or through UI SQL tools that use JDBC)</td>
<td>ENV Variable HIVE_PORT</td>
</tr>
<tr>
<td>Hive Metastore</td>
<td></td>
<td>9083</td>
<td>thrift</td>
<td>Service for accessing metadata about Hive tables and partitions.</td>
<td>Yes (Clients that run Hive, Pig and potentially M/R jobs that use HCatalog)</td>
<td>hive.metastore.uris</td>
</tr>
</tbody>
</table>

* To change the metastore port, use this `hive` command: `hive --service metastore -p port_number`

7.5. HBase Ports

The following table lists the default ports used by the various HBase services.

Table 7.5. HBase Ports

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMaster</td>
<td>Master Node hosts (HBase Master Node and any back-up HBase Master node)</td>
<td>60000</td>
<td></td>
</tr>
<tr>
<td>HMaster Info Web UI</td>
<td>Master Node hosts (HBase master Node and back up HBase Master node if any)</td>
<td>60010</td>
<td>http</td>
</tr>
<tr>
<td>Region Server</td>
<td>All Slave Node hosts</td>
<td>60020</td>
<td></td>
</tr>
<tr>
<td>Region Server</td>
<td>All Slave Node hosts</td>
<td>60030</td>
<td>http</td>
</tr>
<tr>
<td>HBase REST Server (optional)</td>
<td>All REST Servers</td>
<td>8080</td>
<td>http</td>
</tr>
<tr>
<td>HBase REST Server Web UI (optional)</td>
<td>All REST Servers</td>
<td>8085</td>
<td>http</td>
</tr>
<tr>
<td>HBase Thrift Server (optional)</td>
<td>All Thrift Servers</td>
<td>9090</td>
<td></td>
</tr>
<tr>
<td>HBase Thrift Server Web UI (optional)</td>
<td>All Thrift Servers</td>
<td>9095</td>
<td></td>
</tr>
</tbody>
</table>

7.6. ZooKeeper Ports

The following table lists the default ports used by the various ZooKeeper services.
Table 7.6. HBase Ports

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZooKeeper Server</td>
<td>All ZooKeeper Node hosts</td>
<td>2888</td>
<td></td>
<td>Port used by ZooKeeper peers to talk to each other. See here for more information.</td>
<td>No</td>
<td>hbase.zookeeper.peerport</td>
</tr>
<tr>
<td>ZooKeeper Server</td>
<td>All ZooKeeper Node hosts</td>
<td>3888</td>
<td></td>
<td>Port used by ZooKeeper peers to talk to each other. See here for more information.</td>
<td>No</td>
<td>hbase.zookeeper.leaderport</td>
</tr>
<tr>
<td>ZooKeeper Server</td>
<td>All ZooKeeper Hosts</td>
<td>2181</td>
<td></td>
<td>Property from ZooKeeper's config zoo.cfg. The port at which the clients will connect.</td>
<td>Yes</td>
<td>hbase.zookeeper.property.clientPort</td>
</tr>
</tbody>
</table>

7.7. WebHCat Port

The following table lists the default port used by the WebHCat service.

Table 7.7. WebHCat Port

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebHCat Server</td>
<td>Any utility machine</td>
<td>50111</td>
<td>http</td>
<td>Web API on top of HCatalog and other Hadoop services</td>
<td>Yes</td>
<td>templeton.port</td>
</tr>
</tbody>
</table>

7.8. Ganglia Ports

The following table lists the default ports used by the various Ganglia services.

Table 7.8. Ganglia Ports

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ganglia Server</td>
<td>Ganglia server host</td>
<td>8660/61/62/63</td>
<td></td>
<td>For metric (gmond) collectors</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Ganglia Monitor</td>
<td>All Slave Node hosts</td>
<td>8660</td>
<td></td>
<td>For monitoring (gmond) agents</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Ganglia Server</td>
<td>Ganglia server host</td>
<td>8651</td>
<td></td>
<td>For ganglia gmetad</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ganglia Web</td>
<td>Ganglia server host</td>
<td></td>
<td>http⁴</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

⁴See Optional: Set Up HTTPS for Ganglia for instructions on enabling HTTPS.
7.9. MySQL Port

The following table lists the default port used by the MySQL service.

Table 7.9. MySQL Port

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>MySQL</td>
<td>MySQL database server host</td>
<td>3306</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7.10. Ambari Ports

The following table lists the default ports used by Ambari.

Table 7.10. Ambari Web

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambari Server</td>
<td>Ambari Server host</td>
<td>8080</td>
<td>http</td>
<td>Interface to Ambari Web and Ambari REST API</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Ambari Server</td>
<td>Ambari Server host</td>
<td>8440</td>
<td>https</td>
<td>Handshake Port for Ambari Agents to Ambari Server</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Ambari Server</td>
<td>Ambari Server host</td>
<td>8441</td>
<td>https</td>
<td>Registration and Heartbeat Port for Ambari Agents to Ambari Server</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

*See Optional: Change the Ambari Server Port for instructions on changing the default port.

*See Optional: Set Up HTTPS for Ambari Web for instructions on enabling HTTPS.

7.11. Nagios Ports

The following table lists the default port used by Nagios.

Table 7.11. Nagios

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nagios Server</td>
<td>Nagios server host</td>
<td>80</td>
<td>http</td>
<td>Nagios Web UI</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

*See Optional: Set Up HTTPS for Nagios for instructions on enabling HTTPS.
8. Appendix: 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 nav 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.
4. **Select Hosts**: Select a host for the additional NameNode and the JournalNodes. The wizard suggests options, but you can adjust using the dropdown lists. Click **Next** to proceed.
5. **Review**: Confirm your host selections and click **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**.
7. **Configure Components:** The wizard configures your components, displaying progress bars to let you track the steps. Click **Next** to continue.

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**.
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.

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.
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.

12. If you are using Hive, you need to 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:

   ```
   /usr/lib/hive/bin/metatool -listFSRoot
   ```

   The output might look like this:

   ```
   Listing FS Roots...
   ```
b. Use this command to change the FS root:

```
$ /usr/lib/hive/bin/metatool -updateLocation <new-location> <old-location>
```

For example, where the Nameservice ID is mycluster:

```
```

The output might look like this:

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

13. If you are using Oozie, you need to 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>
    </map-reduce>
  </action>
</workflow-app>
```

### 8.1.1. 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**
8.1.1.1. 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.

8.1.1.2. 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.

  ```bash
  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:

  ```bash
  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.

8.1.1.3. 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.

8.1.1.4. 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 8.1. Set Environment Variables**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>export AMBARI_USER=AMBARI_USERNAME</td>
<td>Substitute the value of the administrative user for Ambari Web. The default value is admin.</td>
</tr>
<tr>
<td>export AMBARI_PW=AMBARI_PASSWORD</td>
<td>Substitute the value of the administrative password for Ambari Web. The default value is admin.</td>
</tr>
<tr>
<td>export AMBARI_PORT=AMBARI_PORT</td>
<td>Substitute the Ambari Web port. The default value is 8080.</td>
</tr>
</tbody>
</table>
Variable | Value
--- | ---
export AMBARI_PROTO=AMBARI_PROTOCOL | Substitute the value of the protocol for connecting to Ambari Web. Options are http or https. The default value is http.
export CLUSTER_NAME=CLUSTER_NAME | Substitute the name of your cluster, set during the Ambari Install Wizard process. For example: mycluster.
export NAMENODE_HOSTNAME=NN_HOSTNAME | Substitute the FQDN of the host for the non-HA NameNode. For example: nn01.mycompany.com.
export ADDITIONAL_NAMENODE_HOSTNAME=ANN_HOSTNAME | Substitute the FQDN of the host for the additional NameNode in your HA setup.
export SECONDARY_NAMENODE_HOSTNAME=SNN_HOSTNAME | Substitute the FQDN of the host for the Secondary NameNode for the non-HA setup.
export JOURNALNODE1_HOSTNAME=JOUR1_HOSTNAME | Substitute the FQDN of the host for the first Journal Node.
export JOURNALNODE2_HOSTNAME=JOUR2_HOSTNAME | Substitute the FQDN of the host for the second Journal Node.
export JOURNALNODE3_HOSTNAME=JOUR3_HOSTNAME | Substitute the FQDN of the host for the third Journal Node.

**Important**

Double check that these environment variables are set correctly.

### 8.1.1.5. 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:

   ```bash
   /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:

   ```bash
   /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.

### 8.1.1.6. 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} -H "X-Requested-By: ambari" -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} -H "X-Requested-By: ambari" -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} -H "X-Requested-By: ambari" -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} -H "X-Requested-By: ambari" -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.

### 8.1.1.7. 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:
6. If the property \texttt{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 \texttt{fs.defaultFS} needs to be modified as it points to a NameService ID.

```
"fs.defaultFS" : "hdfs://nn01.mycompany.com"
```
The property \texttt{fs.defaultFS} does not need to be changed as it points to a specific NameNode and not a NameService ID.

7. To revert the property \texttt{fs.defaultFS} to the NameNode host value, on the Ambari Server host:

```
/var/lib/ambarni-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}
```

8. Verify that the \texttt{core-site} properties are now properly set. On the Ambari Server host:

```
/var/lib/ambarni-server/resources/scripts/configs.sh -u $AMBARI_USER -p $AMBARI_PW -port $AMBARI_PORT get localhost $CLUSTER_NAME core-site
```
The property \texttt{fs.defaultFS} should be set to point to the NameNode host and the property \texttt{ha.zookeeper.quorum} should not be there.

8.1.1.8. Recreate the Secondary NameNode

You may need to recreate your Secondary NameNode.

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

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -H "X-Requested-By: ambari" -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 \texttt{items} array, you must recreate your Secondary NameNode. Otherwise you can go on to \textbf{Re-enable Secondary NameNode}.

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

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

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

```
curl -u ${AMBARI_USER}:${AMBARI_PW} -H "X-Requested-By: ambari" -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.

### 8.1.1.9. Re-enable the Secondary NameNode

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

```bash
curl -u ${AMBARI_USER}:${AMBARI_PW} -H "X-Requested-By: ambari" -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:

```bash
curl -u ${AMBARI_USER}:${AMBARI_PW} -H "X-Requested-By: ambari" -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.

### 8.1.1.10. 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:

```bash
curl -u ${AMBARI_USER}:${AMBARI_PW} -H "X-Requested-By: ambari" -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:

```bash
curl -u ${AMBARI_USER}:${AMBARI_PW} -H "X-Requested-By: ambari" -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} -H "X-Requested-By: ambari" -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} -H "X-Requested-By: ambari" -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:

```bash
curl -u ${AMBARI_USER}:${AMBARI_PW} -H "X-Requested-By: ambari" -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.
8.1.1.11. 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} -H "X-Requested-By: ambari" -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} -H "X-Requested-By: ambari" -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} -H "X-Requested-By: ambari" -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.

8.1.1.12. Verify your HDFS Components

Make sure you have the correct componets 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.

8.1.1.13. 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.
Part III. Deploying, Configuring, and Upgrading Ambari for Hadoop 1.x

This section describes setting up Hadoop 1.x. It includes:

- Installing, Configuring, and Deploying the Cluster for Hadoop 1.x
- Troubleshooting Ambari Deployments for Hadoop 1.x
- Appendix: Upgrading Ambari Server to 1.2.5
- Appendix: Upgrading the HDP Stack to 1.3.3
- Appendix: Configuring Ports for Hadoop 1.x
9. Installing, Configuring, and Deploying the Cluster for Hadoop 1.x

This section describes using the Ambari install wizard in your browser to complete your installation, configuration and deployment of Hadoop.

9.1. Log into Apache Ambari

Once you have started the Ambari service, you can access the Ambari Install Wizard through your browser.

1. Point your browser to http://{main.install.hostname}:8080.

2. Log in to the Ambari Server using the default username/password: admin/admin. You can change this later to whatever you want.

9.2. Welcome

The first step creates the cluster name.

1. At the Welcome page, type a name for the cluster you want to create in the text box. No whitespaces or special characters can be used in the name.

2. Click the Next button.

9.3. Select Stack

The Service Stack (or simply the Stack) is a coordinated and tested set of Hadoop components. Use the radio button to select the Stack version you want to install.

9.4. Install Options

In order to build up the cluster, the install wizard needs to know general information about how you want to set up your cluster. You need to supply the FQDN of each of your hosts. The wizard also needs to access the private key file you created in Set Up Password-less SSH. It uses these to locate all the hosts in the system and to access and interact with them securely.

1. Use the Target Hosts text box to enter your list of host names, one per line. You can use ranges inside brackets to indicate larger sets of hosts. For example, for host01.domain through host10.domain use host[01-10].domain

2. If you want to let Ambari automatically install the Ambari Agent on all your hosts using SSH, select Provide your SSH Private Key and either use the Choose File button in the

Note

If you are deploying on EC2, use the internal Private DNS hostnames.
Host Registration Information section to find the private key file that matches the public key you installed earlier on all your hosts or cut and paste the key into the text box manually.

Note

If you are using IE 9, the Choose File button may not appear. Use the text box to cut and paste your private key manually.

Fill in the username for the SSH key you have selected. If you do not want to use root, you must provide the username for an account that can execute sudo without entering a password.

3. If you do not want Ambari to automatically install the Ambari Agents, select Perform manual registration. See Appendix: Installing Ambari Agents Manually for more information.

4. Advanced Options

• If you want to use a local software repository (for example, if your installation does not have access to the Internet), check Use a Local Software Repository. For more information on using a local repository see Optional: Configure the Local Repositories

5. Click the Register and Confirm button to continue.

9.5. Confirm Hosts

This screen lets you confirm that Ambari has located the correct hosts for your cluster and to check those hosts to make sure they have the correct directories, packages, and processes to continue the install.

If any hosts were selected in error, you can remove them by selecting the appropriate checkboxes and clicking the grey Remove Selected button. To remove a single host, click the small white Remove button in the Action column.

At the bottom of the screen, you may notice a yellow box that indicates some warnings were encountered during the check process. For example, your host may have already had a copy of wget or curl. Click Click here to see the warnings to see a list of what was checked and what caused the warning. On the same page you can get access to a python script that can help you clear any issues you may encounter and let you run Rerun Checks.

When you are satisfied with the list of hosts, click Next.

9.6. Choose Services

Hortonworks Data Platform is made up of a number of services. You must at a minimum install HDFS, but you can decide which of the other services you want to install. See Understand the Basics for more information on your options.

1. Select all to preselect all items or minimum to preselect only HDFS.
2. Use the checkboxes to unselect (if you have used all) or select (if you have used minimum) to arrive at your desired list of components.

**Note**

If you want to use Ambari for monitoring your cluster, make sure you select Nagios and Ganglia. If you do not select them, you get a warning popup when you finish this section. If you are using other monitoring tools, you can ignore the warning.

3. When you have made your selections, click Next.

### 9.7. Assign Masters

The Ambari install wizard attempts to assign the master nodes for various services you have selected to appropriate hosts in your cluster. The right column shows the current service assignments by host, with the hostname and its number of CPU cores and amount of RAM indicated.

1. To change locations, click the dropdown list next to the service in the left column and select the appropriate host.

2. To remove a ZooKeeper instance, click the green minus icon next to the host address you want to remove.

3. When you are satisfied with the assignments, click the Next button.

### 9.8. Assign Slaves and Clients

The Ambari install wizard attempts to assign the slave components (DataNodes, NodeManagers, and RegionServers) to appropriate hosts in your cluster. It also attempts to select hosts for installing the appropriate set of clients.

1. Use all or none to select all of the hosts in the column or none of the hosts, respectively.

   If a host has a red asterisk next to it, that host is also running one or more master components. Hover your mouse over the asterisk to see which master components are on that host.

2. Fine-tune your selections by using the checkboxes next to specific hosts.

**Note**

As an option you can start the HBase REST server manually after the install process is complete. It can be started on any host that has the HBase Master or the Region Server installed. If you attempt to start it on the same host as the Ambari server, however, you need to start it with the -p option, as its default port is 8080 and that conflicts with the Ambari Web default port.

```
/usr/lib/hbase/bin/hbase-daemon.sh start rest -p <custom_port_number>
```
3. When you are satisfied with your assignments, click the Next button.

9.9. Customize Services

The Customize Services screen presents you with a set of tabs that let you manage configuration settings for Hadoop components. The wizard attempts to set reasonable defaults for each of the options here, but you can use this set of tabs to tweak those settings. and you are strongly encouraged to do so, as your requirements may be slightly different. Pay particular attention to the directories suggested by the installer.

Hover your mouse over each of the properties to see a brief description of what it does. The number of tabs you see is based on the type of installation you have decided to do. In a complete installation there are nine groups of configuration properties and other related options, such as database settings for Hive and Oozie and admin name/password and alert email for Nagios.

The install wizard sets reasonable defaults for all properties except for those related to databases in the Hive tab and the Oozie tab, and two related properties in the Nagios tab. These four are marked in red and are the only ones you must set yourself.

**Note**

If you decide to use an existing database instance for Hive/HCatalog or for Oozie, you must have completed the preparations described in Using Non-Default Databases prior to running the install wizard.

Click the name of the group in each tab to expand and collapse the display.

9.9.1. Service Users and Groups

The individual services in Hadoop are each run under the ownership of a corresponding Unix account. These accounts are known as service users. These service users belong to a special Unix group. In addition there is a special service user for running smoke tests on components during installation and on-demand using the Management Header in the Services View of the Ambari Web GUI. Any of these users and groups can be customized using the Misc tab of the Customize Services step.

If you choose to customize names, Ambari checks to see if these custom accounts already exist. If they do not exist, Ambari creates them. The default accounts are always created during installation whether or not custom accounts are specified. These default accounts are not used and can be removed post-install.

**Note**

All new service user accounts, and any existing user accounts used as service users, must have a UID >= 1000.

**Table 9.1. Service Users**

<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Default User Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS</td>
<td>NameNode</td>
<td>hdfs</td>
</tr>
</tbody>
</table>
### 9.9.2. Properties That Depend on Service Usernames/Groups

Some properties must be set to match specific service usernames or service groups. If you have set up non-default, customized service usernames for the HDFS or HBase service or the Hadoop group name, you must edit the following properties:

#### Table 9.3. HDFS Settings: Advanced

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>dfs.permissions.supergroup</td>
<td>The same as the HDFS username. The default is &quot;hdfs&quot;</td>
</tr>
<tr>
<td>dfs.cluster.administrators</td>
<td>A single space followed by the HDFS username.</td>
</tr>
<tr>
<td>dfs.block.local-path-access.user</td>
<td>The HBase username. The default is &quot;hbase&quot;.</td>
</tr>
</tbody>
</table>

#### Table 9.4. MapReduce Settings: Advanced

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapreduce.tasktracker.group</td>
<td>The Hadoop group name. The default is &quot;hadoop&quot;.</td>
</tr>
<tr>
<td>mapreduce.cluster.administrators</td>
<td>A single space followed by the Hadoop group name.</td>
</tr>
</tbody>
</table>
9.9.3. Recommended Memory Configurations for the MapReduce Service

The following recommendations can help you determine appropriate memory configurations based on your usage scenario:

- Make sure that there is enough memory for all the processes. Remember that system processes take around 10% of the available memory.

- For co-deploying an HBase RegionServer and MapReduce service on the same node, reduce the RegionServer’s heap size (use the `HBase Settings: RegionServer: HBase Region Servers maximum Java heap size` property to modify the RegionServer heap size).

- For co-deploying an HBase RegionServer and the MapReduce service on the same node, or for memory intensive MapReduce applications, modify the map and reduce slots as suggested in the following example:

  **EXAMPLE:** For co-deploying an HBase RegionServer and the MapReduce service on a machine with 16GB of available memory, the following would be a recommended configuration:

  2 GB: system processes
  8 GB: MapReduce slots. 6 Map + 2 Reduce slots per 1 GB task
  4 GB: HBase RegionServer
  1 GB: TaskTracker
  1 GB: DataNode

  To change the number of Map and Reduce slots based on the memory requirements of your application, use the following properties:

  - **MapReduce Settings: TaskTracker:** Number of Map slots per node
  - **MapReduce Settings: TaskTracker:** Number of Reduce slots per node

9.10. Review

The assignments you have made are displayed. Check to make sure everything is correct. If you need to make changes, use the left navigation bar to return to the appropriate screen.

To print your information for later reference, click **Print**.

When you are satisfied with your choices, click the **Deploy** button.

9.11. Install, Start and Test

The progress of the install is shown on the screen. Each component is installed and started and a simple test is run on the component. You are given an overall status on the process in the progress bar at the top of the screen and a host by host status in the main section.
To see specific information on what tasks have been completed per host, click the link in the Message column for the appropriate host. In the Tasks pop-up, click the individual task to see the related log files. You can select filter conditions by using the Show dropdown list. To see a larger version of the log contents, click the Open icon or to copy the contents to the clipboard, use the Copy icon.

Depending on which components you are installing, the entire process may take 40 or more minutes. Please be patient.

When Successfully installed and started the services appears, click Next.

**9.12. Summary**

The Summary page gives you a summary of the accomplished tasks. Click Complete. You are taken to the Ambari Web GUI.
10. Troubleshooting Ambari Deployments

The following information can help you troubleshoot issues you may run into with your Ambari-based installation.

10.1. Getting the Logs

The first thing to do if you run into trouble is to find the logs. Ambari Server logs are found at /var/log/ambari-server/ambari-server.log Ambari Agent logs are found at /var/log/ambari-agent/ambari-agent.log.

10.2. Quick Checks

- Make sure all the appropriate services are running. If you have access to Ambari Web, use the Services View to check the status of each component. If you do not have access to Manage Services, you must start and stop the services manually.

- If the first HDFS put command fails to replicate the block, the clocks in the nodes may not be synchronized. Make sure that Network Time Protocol (NTP) is enabled for your cluster.

- If HBase does not start, check if its slaves are running on 64-bit JVMs. Ambari requires that all hosts must run on 64-bit machines.

- Make sure umask is set to 0022.

- Make sure the HCatalog host can access the MySQL server. From a shell try:

  ```bash
  mysql -h $FQDN_for_MySQL_server -u $FQDN_for_HCatalog_Server -p
  ```

  You will need to provide the password you set up for Hive/HCatalog during the installation process.

- Make sure MySQL is running. By default, MySQL server does not start automatically on reboot.

  To set auto-start on boot, from a shell, type:

  ```bash
  chkconfig --level 35 mysql on
  ```

  To then start the service manually from a shell, type:

  ```bash
  service mysqld start
  ```

10.3. Specific Issues

The following are common issues you might encounter.
10.3.1. Problem: Browser crashed before Install Wizard completed

Your browser crashes or you accidently close your browser before the Install Wizard completes.

10.3.1.1. Solution

The response to a browser closure depends on where you are in the process:

• The browser closes prior to hitting the **Deploy** button.

  Re-launch the **same** browser and continue the install process. Using a different browser forces you to re-start the entire process

• The browser closes after the **Deploy** button has launched the **Install, Start, and Test** screen

  Re-launch the same browser and continue the process or use a different browser and re-login. You are returned to the **Install, Start, and Test** screen.

10.3.2. Problem: Install Wizard reports that the cluster install has failed

The Install, Start, and Test screen reports that the cluster install has failed.

10.3.2.1. Solution

The response to a report of install failure depends on the cause of the failure:

• The failure is due to intermittent network connection errors during software package installs.

  Use the **Retry** button on the **Install, Start, and Test** screen.

• The failure is due to misconfiguration or other setup errors.

  1. Use the left nav bar to go back to the appropriate screen; for example, **Customize Services**.

  2. Make your changes.

  3. Continue in the normal way.

• The failure occurs during the start/test sequence.

  1. Click **Next** and **Complete** and proceed to the Monitoring **Dashboard**.

  2. Use the **Services View** to make your changes.

  3. Re-start the service using the **Management Header**.
• The failure is due to something else.

1. Open an SSH connection to the Ambari Server host.

2. Clear the database. At the command line, type:

```
    ambari-server reset
```

3. Clear the browser's cache.

4. Re-run the entire Install Wizard.

10.3.3. Problem: “Unable to create new native thread” exceptions in HDFS DataNode logs or those of any system daemon

If your nproc limit is incorrectly configured, the smoke tests fail and you see an error similar to this in the DataNode logs:

```
INFO org.apache.hadoop.hdfs.DFSClient: Exception
    incrateBlockOutputStream java.io.EOFException
INFO org.apache.hadoop.hdfs.DFSClient: Abandoning block
blk_-6935524980745310745_139190
```

10.3.3.1. Solution:

In certain recent Linux distributions (like RHEL/Centos/Oracle Linux 6.x), the default value of nproc is lower than the value required if you are deploying the HBase service. To change this value:

1. Using a text editor, open `/etc/security/limits.d/90-nproc.conf` and change the nproc limit to approximately 32000. For more information, see ulimit and nproc recommendations for HBase servers.

2. Restart the HBase server.

10.3.4. Problem: The “yum install ambari-server” Command Fails

You are unable to get the initial install command to run.

10.3.4.1. Solution:

You may have incompatible versions of some software components in your environment. Check the list in Check Existing Installs and make any necessary changes. Also make sure you are running a Supported Operating System.

10.3.5. Problem: HDFS Smoke Test Fails

If your DataNodes are incorrectly configured, the smoke tests fail and you get this error message in the DataNode logs:
DisallowedDataNodeException
org.apache.hadoop.hdfs.server.protocol.
DisallowedDatanodeException

10.3.5.1. Solution:

- Make sure that reverse DNS look-up is properly configured for all nodes in your cluster.
- Make sure you have the correct FQDNs when specifying the hosts for your cluster. Do not use IP addresses - they are not supported.

Restart the installation process.

10.3.6. Problem: The HCatalog Daemon Metastore Smoke Test Fails

If the HCatalog smoke test fails, this is displayed in your console:

Metastore startup failed, see /var/log/hcatalog/hcat.err

10.3.6.1. Solution:

1. Log into the HCatalog node in your cluster
2. Open /var/log/hcatalog/hcat.err or /var/log/hive/hive.log (one of the two will exist depending on the installation) with a text editor
3. In the file, see if there is a MySQL Unknown Host Exception like this:

   at java.lang.reflect.Method.invoke (Method.java:597)
   at org.apache.hadoop.util.Runjar.main (runjar.java:156)
   Caused by: java.net.UnknownHostException:mysql.host.com
   at java.net.InetAddress.getAllByName(INetAddress.java:1157)

   This exception can be thrown if you are using a previously existing MySQL instance and you have incorrectly identified the hostname during the installation process. When you do the reinstall, make sure this name is correct.

4. In the file, see if there is an ERROR Failed initializing database entry like this:

   11/12/29 20:52:04 ERROR DataNucleus.Plugin: Bundle
   org.eclipse.jdt.core required

   This exception can be thrown if you are using a previously existing MySQL instance and you have incorrectly identified the username/password during the installation process. It can also occur when the user you specify does not have adequate privileges on the database. When you do the reinstall, make sure this username/password is correct and that the user has adequate privilege.

5. Restart the installation process.
10.3.7. Problem: MySQL and Nagios fail to install on RightScale CentOS 5 images on EC2

When using a RightScale CentOS 5 AMI on Amazon EC2, in certain cases MySQL and Nagios will fail to install. The MySQL failure is due to a conflict with the pre-installed MySQL and the use of the RightScale EPEL repository (error "Could not find package mysql-server"). Nagios fails to install due to conflicts with the RightScale php-common library.

10.3.7.1. Solution:

On the machines that will host MySQL and Nagios as part of your Hadoop cluster, perform the following:

1. Remove the existing MySQL server
   
   ```bash
   yum erase MySQL-server-community
   ```

2. Install MySQL server with a disabled RightScale EPEL repository
   
   ```bash
   yum install mysql-server --disable-repo=rightscale-epel
   ```

3. Remove the php-common library
   
   ```bash
   yum erase php-common-5.2.4-RightScale.x86
   ```

10.3.8. Problem: Trouble starting Ambari on system reboot

If you reboot your cluster, you must restart the Ambari Server and all the Ambari Agents manually.

10.3.8.1. Solution:

Log in to each machine in your cluster separately

1. On the Ambari Server host machine:
   
   ```bash
   ambari-server start
   ```

2. On each host in your cluster:
   
   ```bash
   ambari-agent start
   ```

10.3.9. Problem: Metrics and Host information display incorrectly in Ambari Web

Charts appear incorrectly or not at all despite being available in the native Ganglia interface or Host health status is displayed incorrectly.

10.3.9.1. Solution:

All the hosts in your cluster and the machine from which you browse to Ambari Web must be in sync with each other. The easiest way to assure this is to enable NTP.
10.3.10. Problem: On SUSE 11 Ambari Agent crashes within the first 24 hours

SUSE 11 ships with Python version 2.6.0-8.12.2 which contains a known bug that causes this crash.

10.3.10.1. Solution:

Upgrade to Python version 2.6.8-0.15.1

10.3.11. Problem: Attempting to Start HBase REST server causes either REST server or Ambari Web to fail

As an option you can start the HBase REST server manually after the install process is complete. It can be started on any host that has the HBase Master or the Region Server installed. If you install the REST server on the same host as the Ambari server, the http ports will conflict.

10.3.11.1. Solution

In starting the REST server, use the -p option to set a custom port. Use the following command to start the REST server.

```bash
/usr/lib/hbase/bin/hbase-daemon.sh start rest -p <custom_port_number>
```

10.3.12. Problem: Multiple Ambari Agent processes are running, causing re-register

On a cluster host `ps aux | grep ambari-agent` shows more than one agent process running. This causes Ambari Server to get incorrect ids from the host and forces Agent to restart and re-register.

10.3.12.1. Solution

On the affected host, kill the processes and restart.

1. Kill the Agent processes and remove the Agent PID files found here: `/var/run/ambari-agent/ambari-agent.pid`

2. Restart the Agent process:

   ```bash
   ambari-agent start
   ```

10.3.13. Problem: Some graphs do not show a complete hour of data until the cluster has been running for an hour

When a cluster is first started, some graphs, like Services View -> HDFS and Services View -> MapReduce, do not plot a complete hour of data, instead showing data only for the
length of time the service has been running. Other graphs display the run of a complete hour.

10.3.13.1. Solution

Let the cluster run. After an hour all graphs will show a complete hour of data.

10.3.14. Problem: After performing a cluster install the Nagios server is not started

The Nagios server is not started after a cluster install and you are unable to manage it from Ambari Web.

10.3.14.1. Solution

1. Log into the Nagios server host.

2. Confirm that the Nagios server is not running. From a shell:
   
   ```
   ps -ef | grep nagios
   ```
   
   You should not see a Nagios process running.

3. Start the Nagios process manually. From a shell:
   
   ```
   service nagios start
   ```
   
   4. The server starts. You should be able to see that started state reflected in Ambari Web. You can now manage (start/stop) Nagios from Ambari Web.

10.3.15. Problem: A service with a customized service user is not appearing properly in Ambari Web

You are unable to monitor or manage a service in Ambari Web when you have created a customized service user name with a hyphen, for example, 'hdfs-user'.

10.3.15.1. Solution

Hyphenated service user names are not supported. You must re-run the Ambari Install Wizard and create a different name.

10.3.16. Problem: Updated configuration changes are not pushed to client/gateway nodes

Currently configuration changes are only pushed to daemon running nodes, so any changes are not automatically pushed to client only nodes such as gateway nodes.

10.3.16.1. Solution

Copy the files to the client nodes manually.
11. Appendix: Upgrading Ambari Server to 1.2.5

This process upgrades Ambari Server. It does not change the underlying Hadoop Stack. This is a twelve step manual process.

**Note**
You must know the location of the Nagios server for Step 9. Use the Services View-> Summary panel to locate the host on which it is running.

1. Stop the Ambari Server and all Ambari Agents. From the Ambari Server host:
   ```
   ambari-server stop
   ```
   From each Ambari Agent host:
   ```
   ambari-agent stop
   ```

2. Get the new Ambari bits. Use `wget` to fetch the repository file and replace the old repo file with the new repo file on every host.
   - Fetch the new repo file:
     ```
     For RHEL/CentOS 5/Oracle Linux 5
     wget http://public-repo-1.hortonworks.com/ambari/centos5/1.x/updates/1.2.5.17/ambari.repo
     ```
     ```
     For RHEL/CentOS 6/Oracle Linux 6
     wget http://public-repo-1.hortonworks.com/ambari/centos6/1.x/updates/1.2.5.17/ambari.repo
     ```
     ```
     For SLES 11
     wget http://public-repo-1.hortonworks.com/ambari/suse11/1.x/updates/1.2.5.17/ambari.repo
     ```

   - Replace the old repo file with the new repo file.
     ```
     For RHEL/CentOS 5/Oracle Linux 5
     cp ambari.repo /etc/yum.repos.d/ambari.repo
     ```
     ```
     For RHEL/CentOS 6/Oracle Linux 6
     ```

**Important**
Check your current directory before you download the new repo file to make sure that there are no previous versions of the 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.

- Replace the old repo file with the new repo file.
3. Upgrade Ambari Server. From the Ambari Server host:

- RHEL/CentOS
  
  ```bash
  yum clean all
  yum upgrade ambari-server ambari-log4j
  ```

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

4. Check for upgrade success:

- As the process runs, the console should produce output similar, although not identical, to this:

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

  After the process is complete, check each host to make sure the new 1.2.4 files have been installed:

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

- If the upgrade fails, the console should produce output similar to this:

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

5. Check to see if you have a folder named `/etc/ambari-server/conf.save`. If you do, rename it back:

  ```bash
  mv /etc/ambari-server/conf.save /etc/ambari-server/conf
  ```

6. Upgrade the Ambari Server schema. From the Ambari Server host:

  ```bash
  ambari-server upgrade
  ```

7. Upgrade the Ambari Agent on all hosts. From each Ambari Agent host:

- RHEL/CentOS
yum upgrade ambari-agent ambari-log4j

- SLES

zypper up ambari-agent ambari-log4j

**Note**

If you get a warning that begins "There are some running programs that use files deleted by recent upgrade" you can ignore it.

8. Check to see if you have a file named `/etc/ambari-agent/conf.save` on each Agent host. If you do, rename it back. On each Agent host:

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

9. Upgrade the Nagios and Ganglia addons package and restart. On the Nagios/Ganglia host:

- RHEL/CentOS

```
yum upgrade hdp_mon_nagios_addons hdp_mon_ganglia_addons
service httpd restart
```

- SLES

```
zypper up hdp_mon_nagios_addons hdp_mon_ganglia_addons
service apache2 restart
```

10. Start the Server and the Agents on all hosts. From the Server host:

```
ambari-server start
```

From each Agent host:

```
ambari-agent start
```

11. Open **Ambari Web**. Point your browser to `http://{your.ambari.server}:8080`

**Important**

You need to refresh your browser so that it loads the new version of the code. Hold the Shift key down while clicking the refresh button on the browser. If you have problems, clear your browser cache manually and restart Ambari Server.

Use the Admin name and password you have set up to log in.

12. Re-start the Ganglia, Nagios, and MapReduce services. In **Ambari Web**.

a. Go to the **Services View** and select each service.

b. Use the **Management Header** to stop and re-start each service.
12. Appendix: Upgrading the HDP Stack to 1.3.3

The stack is the coordinated set of Hadoop components that you have installed. If you have a current instance of the 1.2.0/1.2.1 stack that was installed and managed by Ambari that you want to upgrade to the current 1.3.3 version of the stack and to also upgrade to the 1.2.5 version of Ambari Server and Agents, use the following instructions. This insures that the upgraded stack can still be managed by Ambari.

If you are upgrading from the 1.3.0 stack to the 1.3.3 stack, use Section 5: Upgrading the Stack (from 1.3.0 to 1.3.3), not Section 4: Upgrading the Stack (from 1.2.* to 1.3.3).

Note

If you have already upgraded to Ambari Server 1.2.5 and just want to upgrade the HDP stack, you can skip Sections 9.2 and 9.3.

12.1. Preparing for the Upgrade

Use the following steps to prepare your system for the upgrade.

1. If you are upgrading Ambari as well as the stack, you must know the location of the Nagios and Ganglia servers for that process. Use the Services->Nagios/Ganglia->Summary panel to locate the hosts on which they are running.

2. Use the Services view on the Ambari Web UI to stop all services, including MapReduce and all clients, running on HDFS. Do not stop HDFS yet.

3. Create the following logs and other files.

   Because the upgrade to 1.3.3 includes a version upgrade of HDFS, creating these logs allows you to check the integrity of the file system post upgrade. While this is not absolutely necessary, doing so is strongly encouraged.

   a. 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.

      ```bash
      su $HDFS_USER
      hadoop fsck / -files -blocks -locations > /tmp/dfs-old-fsck-1.log
      
      where $HDFS_USER is the HDFS Service user (by default, hdfs).
      ```

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

      ```bash
      su $HDFS_USER
      hadoop dfs -lsr / > /tmp/dfs-old-lsr-1.log
      
      where $HDFS_USER is the HDFS Service user (by default, hdfs).
      ```

   c. Create a list of all the DataNodes in the cluster.
su $HDFS_USER
hadoop dfsadmin -report > /tmp/dfs-old-report-1.log

where $HDFS_USER is the HDFS Service user (by default, hdfs).

d. Optional: copy all or unrecoverable only data stored in HDFS to a local file system or to a backup instance of HDFS.

e. Optional: create the logs again and check to make sure the results are identical.

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

    hadoop dfsadmin -safemode enter
    hadoop dfsadmin -saveNamespace

5. Copy the following checkpoint files into a backup directory. You can find the directory by using the Services View in the UI. Select the HDFS service, theConfigs tab, in the Namenode section, look up the property NameNode Directories. It will be on your NameNode host.

    • dfs.name.dir/edits// depending on your system, may not exist
    • dfs.name.dir/image/fsimage

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

7. If you are upgrading Hive, back up the Hive database.

8. Stop Ambari Server. On the Server host:

    ambari-server stop

9. Stop Ambari Agents. On each host:

    ambari-agent stop

### 12.2. Setting Up the Ambari Repository

This process prepares the updated repository.

1. Check to see if you have a conf.save directory for Ambari server and agents. If you do, move them to a back-up location:

    mv /etc/ambari-server/conf.save/ /etc/ambari-server/conf.save.bak
    mv /etc/ambari-agent/conf.save/ /etc/ambari-agent/conf.save.bak

2. Get the new Ambari bits. Use wget to fetch the repository file and replace the old repo file with the new repo file on every host.

    Important

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

- For RHEL/CentOS/Oracle Linux 5
  
  wget http://public-repo-1.hortonworks.com/ambari/centos5/1.x/updates/1.4.1.61/ambari.repo
  cp ambari.repo /etc/yum.repos.d/ambari.repo

- For RHEL/CentOS/Oracle Linux 6
  
  wget http://public-repo-1.hortonworks.com/ambari/centos6/1.x/updates/1.4.1.61/ambari.repo
  cp ambari.repo /etc/yum.repos.d/ambari.repo

- For SLES 11
  
  wget http://public-repo-1.hortonworks.com/ambari/suse11/1.x/updates/1.4.1.61/ambari.repo
  cp ambari.repo /etc/zypp/repos.d/ambari.repo

Note

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

### 12.3. Upgrading Ambari

This process upgrades Ambari Server, Ambari Agents, Ganglia, and Nagios.

1. Upgrade Ambari Server. From the Ambari Server host:

   - RHEL/CentOS/Oracle Linux
     
     ```
     yum clean all
     yum upgrade ambari-server ambari-log4j
     ```

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

2. Check for upgrade success:

   - As the process runs, the console should produce output similar, although not identical, to this:

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

     After the process is complete, check each host to make sure the new 1.2.5 files have been installed:
rpm -qa | grep ambari

- If the upgrade fails, the console should produce output similar to this:

  Setting up Upgrade Process
  No Packages marked for Update

3. Check to see if you have a folder named /etc/ambari-server/conf.save. If you do, rename it back:

    mv /etc/ambari-server/conf.save /etc/ambari-server/conf

4. Upgrade the Ambari Server schema. From the Ambari Server host:

    ambari-server upgrade

5. Upgrade the Ambari Agent on all hosts. From each Ambari Agent host:

- RHEL/CentOS/Oracle Linux

    yum upgrade ambari-agent ambari-log4j

- SLES

    zypper up ambari-agent ambari-log4j

  **Note**
  
  If you get a warning that begins "There are some running programs that use files deleted by recent upgrade" you can ignore it.

6. Check to see if you have a folder named /etc/ambari-agent/conf.save on each Agent host. If you do, rename it back. On each Agent host:

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

7. Upgrade Ganglia and Nagios:

- Upgrade Ganglia Server. From the Ganglia Server host:

  - RHEL/CentOS/Oracle Linux

    yum upgrade ganglia-gmond ganglia-gmetad libganglia
yum erase gweb hdp_mon_ganglia_addons
yum install ganglia-web

  - SLES

    zypper up ganglia-gmond ganglia-gmetad libganglia
zypper remove gweb hdp_mon_ganglia_addons
zypper install ganglia-web

- Upgrade Ganglia Monitor. From every host that has Ganglia Monitor installed:

  - RHEL/CentOS/Oracle Linux

    yum upgrade ganglia-gmond libganglia
• SLES

```bash
zypper up ganglia-gmond libganglia
```

• Upgrade Nagios. From the Nagios Server host:

• RHEL/CentOS/Oracle Linux

```bash
yum upgrade nagios
yum upgrade hdp_mon_nagios_addons
yum erase nagios-plugins-1.4.15
yum install nagios-plugins-1.4.9
```

The 1.4.9 version of the plugin may already be installed. In this case, the second step is a no-op.

• SLES

```bash
zypper up nagios
zypper up hdp_mon_nagios_addons
zypper remove nagios-plugins-1.4.15
zypper install nagios-plugins-1.4.9
```

The 1.4.9 version of the plugin may already be installed. In this case, the second step is a no-op.

### 12.4. Upgrading the Stack (from 1.2.* to 1.3.3)

1. Update the stack version in the Server database, depending on if you are using a local repository:

```bash
ambari-server upgradestack HDP-1.3.3
```

OR

```bash
ambari-server upgradestack HDPLocal-1.3.3
```

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

**Important**

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

• For RHEL/CentOS/Oracle Linux 5

```bash
wget http://public-repo-1.hortonworks.com/HDP/centos5/1.x/updates/1.3.3.0/hdp.repo
mv hdp.repo /etc/yum.repos.d/HDP.repo
```

• For RHEL/CentOS/Oracle Linux 6
wget http://public-repo-1.hortonworks.com/HDP/centos6/1.x/updates/1.3.3.0/hdp.repo
mv hdp.repo /etc/yum.repos.d/HDP.repo

- For SLES 11

wget http://public-repo-1.hortonworks.com/HDP/suse11/1.x/updates/1.3.3.0/hdp.repo
mv hdp.repo /etc/zypp/repos.d/HDP.repo

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

- For RHEL/CentOS/Oracle Linux
  a. Upgrade the following components:

  yum upgrade "collectd*" "epel-release*" "gccxml*" "pig*" "hadoop*" "sqoop*" "zookeeper*" "hbase*" "hive*" "hcatalog*" "webhcat-tar*" hdp_mon_nagios_addons

  b. Check to see that the components in that list are upgraded.

  yum list installed | grep HDP-$old-stack-version-number
  None of the components from that list should appear in the returned list.

  c. Upgrade Oozie, if you are using Oozie:

  rpm -e --nopostun oozie-$old_version_number
  yum install oozie
  You can get the value of $old_version_number from the output of the previous step.

  d. Upgrade Oozie Client:

  yum upgrade oozie-client

  e. Upgrade ZooKeeper:

  i. Check to see if ZooKeeper needs upgrading.

  yum list installed | grep zookeeper
  If the displayed version number is not 3.4.5.1.3.3.0, you need to upgrade.

  ii. Because HBase depends on ZooKeeper, deleting the current version of ZooKeeper automatically deletes the current version of HBase. It must be re-installed. Check to see if HBase is currently installed.

  yum list installed | grep hbase

  iii. Delete the current version of ZooKeeper.

  yum erase zookeeper

  iv. Install ZooKeeper
yum install zookeeper

v. If you need to, re-install HBase.

yum install hbase

vi. Check to see if all components have been upgraded:

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

The only non-upgraded component you may see in this list is extjs, which does not need to be upgraded.

• For SLES

a. Upgrade the following components:

zypper up collectd epel-release* gccxml* pig* hadoop* sqoop* hive*
hcatalog* webhcat-tar* hdp_mon_nagios_addons*
yast --update hadoop hcatalog hive

b. Upgrade ZooKeeper and HBase:

zypper update zookeeper-3.4.5.1.3.3.0
zypper remove zookeeper
zypper se -s zookeeper

You should see ZooKeeper v3.4.5.1.3.3.0 in the output.

Install ZooKeeper v3.4.5.1.3.3.0:

zypper install zookeeper-3.4.5.1.3.3.0

This command also uninstalls HBase. Now use the following commands to install HBase:

zypper install hbase-0.94.6.1.3.3.0
zypper update hbase

c. Upgrade Oozie:

rpm -e --noprereq oozie-$old_version_number
zypper update oozie-3.3.2.1.3.3.0
zypper remove oozie
zypper se -s oozie

You should see Oozie v3.3.2.1.3.3.0 in the output.

Install Oozie v3.3.2.1.3.3.0:

zypper install oozie-3.3.2.1.3.3.0

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

ambari-server start

5. Start each Ambari Agent. On all Agent hosts:

ambari-agent start
6. Because the file system version has now changed you must start the NameNode manually. On the NameNode host:

```bash
sudo su -l $HDFS_USER -c "/usr/lib/hadoop/bin/hadoop-daemon.sh start namenode -upgrade"
```

Depending on the size of your system, this step may take up to 10 minutes.

7. Track the status of the upgrade:

```bash
hadoop dfsadmin -upgradeProgress status
```

Continue tracking until you see:

- **Upgrade for version -44 has been completed.**
- **Upgrade is not finalized.**

**Note**

You finalize the upgrade later.

8. Open the Ambari Web GUI. If you have continued to run the Ambari Web GUI, do a hard reset on your browser. Use **Services View** to start the HDFS service. This starts the SecondaryNameNode and the DataNodes.

9. After the DataNodes are started, HDFS exits safemode. To monitor the status:

```bash
hadoop dfsadmin -safemode get
```

Depending on the size of your system, this may take up to 10 minutes or so. When HDFS exits safemode, this is displayed as a response to the command:

- **Safe mode is OFF**

10. Make sure that the HDFS upgrade was successful. Go through steps 2 and 3 in Section 9.1 to create new versions of the logs and reports. Substitute "new" for "old" in the file names as necessary.

11. Compare the old and new versions of the following:

   - `dfs-old-fsck-1.log` **versus** `dfs-new-fsck-1.log`
     
     The files should be identical unless the `hadoop fsck` reporting format has changed in the new version.

   - `dfs-old-lsr-1.log` **versus** `dfs-new-lsr-1.log`
     
     The files should be identical unless the the format of `hadoop fs -lsr` reporting or the data structures have changed in the new version.

   - `dfs-old-report-1.log` **versus** `fs-new-report-1.log`
     
     Make sure all DataNodes previously belonging to the cluster are up and running.

12. Use the Ambari Web **Services** view-> Services Navigation->**Start All** to start services back up.
13. The upgrade is now fully functional but not yet finalized. Using the `finalize` command removes the previous version of the NameNode and DataNode’s storage directories.

**Important**

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.

To finalize the upgrade:

```
sudo su -l $HDFS_USER -c "hadoop dfsadmin -finalizeUpgrade"
```

where `$HDFS_USER` is the HDFS Service user (by default, `hdfs`).

### 12.5. Upgrading the Stack (from 1.3.0 to 1.3.3)

1. Update the stack version in the Server database, depending on if you are using a local repository:

   ```bash
   ambari-server upgradestack HDP-1.3.3
   
   OR
   
   ambari-server upgradestack HDPLocal-1.3.3
   ```

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

   **Important**

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

   - For RHEL/CentOS/Oracle Linux 5
     
     ```bash
     wget http://public-repo-1.hortonworks.com/HDP/centos5/1.x/updates/1.3.3.0/hdp.repo
     mv hdp.repo /etc/yum.repos.d/HDP.repo
     ```

   - For RHEL/CentOS/Oracle Linux 6
     
     ```bash
     wget http://public-repo-1.hortonworks.com/HDP/centos6/1.x/updates/1.3.3.0/hdp.repo
     mv hdp.repo /etc/yum.repos.d/HDP.repo
     ```

   - For SLES 11
     
     ```bash
     wget http://public-repo-1.hortonworks.com/HDP/suse11/1.x/updates/1.3.3.0/hdp.repo
     mv hdp.repo /etc/zypp/repos.d/HDP.repo
     ```
3. Upgrade the stack on all Agent hosts. Skip any components your installation does not use:

- For RHEL/CentOS/Oracle Linux
  a. Upgrade the following components:

  ```bash
  yum upgrade "collectd*" "epel-release*" "gccxml*" "pig*" "hadoop*"
  "sqoop*" "zookeeper*" "hbase*" "hive*" "hcatalog*" "webhcat-tar*"
  "oozie*" hdp_mon_nagios_addons
  ```

  b. Check to see if those components have been upgraded:

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

  The only non-upgraded component you may see in this list is extjs, which does not need to be upgraded.

- For SLES
  a. Upgrade the following components:

  ```bash
  zypper up collectd gccxml* pig* hadoop* sqoop* hive* hcatalog* webhcat-
  tar* zookeeper* oozie* hbase* hdp_mon_nagios_addons*
  yast --update hadoop hcatalog hive
  ```

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

```bash
ambari-server start
```

5. Start each Ambari Agent. On all Agent hosts:

```bash
ambari-agent start
```

6. Because the file system version has now changed you must start the NameNode manually. On the NameNode host:

```bash
sudo su -l $HDFS_USER -c "/usr/lib/hadoop/bin/hadoop-daemon.sh start
  namenode -upgrade"
```

Depending on the size of your system, this step may take up to 10 minutes.

7. Track the status of the upgrade:

```bash
hadoop dfsadmin -upgradeProgress status
```

Continue tracking until you see:

```
Upgrade for version -44 has been completed.
Upgrade is not finalized.
```

**Note**

You finalize the upgrade later.

8. Open the Ambari Web GUI. If you have continued to run the Ambari Web GUI, do a hard reset on your browser. Use Services View to start the HDFS service. This starts the SecondaryNameNode and the DataNodes.
9. After the DataNodes are started, HDFS exits safemode. To monitor the status:

```
hadoop dfsadmin -safemode get
```

Depending on the size of your system, this may take up to 10 minutes or so. When HDFS exits safemode, this is displayed as a response to the command:

```
Safe mode is OFF
```

10. Make sure that the HDFS upgrade succeeded. Go through steps 2 and 3 in Section 9.1 to create new versions of the logs and reports. Substitute "new" for "old" in the file names as necessary.

11. Compare the old and new versions of the following 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 all DataNodes previously belonging to the cluster are up and running.

12. Use the Ambari Web Services view-> Services Navigation-> Start All to start services back up.

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

```
Important
```

Once 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, however, before another upgrade can be performed.

```
Note
```

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:

```
sudo su -l $HDFS_USER -c "hadoop dfsadmin -finalizeUpgrade"
```

where `$HDFS_USER` is the HDFS Service user (by default, `hdfs`).
13. Appendix: Configuring Ports

The tables below specify which ports must be opened for which ecosystem components to communicate with each other. Make sure the appropriate ports are opened before you install Hadoop.

- HDFS Ports
- MapReduce Ports
- Hive Ports
- HBase Ports
- ZooKeeper Ports
- WebHCat Port
- Ganglia Ports
- MySQL Port
- Ambari Ports
- Nagios Port

13.1. HDFS Ports

The following table lists the default ports used by the various HDFS services.

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode WebUI</td>
<td>Master Node hosts (NameNode and any back-up NameNodes)</td>
<td>50070 http</td>
<td></td>
<td>Web UI to look at current status of HDFS, explore file system</td>
<td>Yes (Typically admins, Dev/Support teams)</td>
<td>dfs.http.address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50470 https</td>
<td></td>
<td>Secure http service</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NameNode metadata service</td>
<td>Master Node hosts (NameNode and any back-up NameNodes)</td>
<td>8020/9000</td>
<td>IPC</td>
<td>File system metadata operations</td>
<td>Yes (All clients who directly need to interact with the HDFS)</td>
<td>Embedded in URI specified by fs.default.name</td>
</tr>
<tr>
<td>DataNode</td>
<td>All Slave Node hosts</td>
<td>50075 http</td>
<td></td>
<td>DataNode WebUI to access the status, logs etc.</td>
<td>Yes (Typically admins, Dev/Support teams)</td>
<td>dfs.datanode.http.address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50475 https</td>
<td></td>
<td>Secure http service</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>50010</td>
<td>Data transfer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>50020</td>
<td>IPC</td>
<td>Metadata operations</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>
13.2. MapReduce Ports

The following table lists the default ports used by the various MapReduce services.

Table 13.2. MapReduce Ports

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>JobTracker WebUI</td>
<td>Master Node hosts (JobTracker Node and any back-up Job-Tracker node)</td>
<td>50030</td>
<td>http</td>
<td>Web UI for JobTracker</td>
<td>Yes</td>
<td>mapred.job.tracker.http.address</td>
</tr>
<tr>
<td>JobTracker</td>
<td>Master Node hosts (JobTracker Node)</td>
<td>8021</td>
<td>IPC</td>
<td>For job submissions</td>
<td>Yes (All clients who need to submit the MapReduce jobs including Hive, Hive server, Pig)</td>
<td>mapred.job.tracker</td>
</tr>
<tr>
<td>TaskTracker Web UI and Shuffle</td>
<td>All Slave Node hosts</td>
<td>50060</td>
<td>http</td>
<td>DataNode Web UI to access status, logs, etc.</td>
<td>Yes (Typically admins, Dev/Support teams)</td>
<td>mapred.task.tracker.http.address</td>
</tr>
<tr>
<td>History Server WebUI</td>
<td></td>
<td>51111</td>
<td>http</td>
<td>Web UI for Job History</td>
<td>Yes</td>
<td>mapreduce.history.server.http.address</td>
</tr>
</tbody>
</table>

13.3. Hive Ports

The following table lists the default ports used by the various Hive services.

Note
Neither of these services is used in a standard HDP installation.

Table 13.3. Hive Ports

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hive Server2</td>
<td>Hive Server machine (Usually a utility machine)</td>
<td>10000</td>
<td>thrift</td>
<td>Service for programatically (Thrift/JDBC) connecting to Hive</td>
<td>Yes (Clients who need to connect to Hive either programatically or through UI SQL tools that use JDBC)</td>
<td>ENV Variable HIVE_PORT</td>
</tr>
</tbody>
</table>
### 13.4. HBase Ports

The following table lists the default ports used by the various HBase services.

**Table 13.4. HBase Ports**

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMaster</td>
<td>Master Node hosts (HBase Master Node and any back-up HBase Master node)</td>
<td>60000</td>
<td></td>
<td></td>
<td></td>
<td>hbase.master.port</td>
</tr>
<tr>
<td>HMaster Info Web UI</td>
<td>Master Node hosts (HBase master Node and back up HBase Master node if any)</td>
<td>60010</td>
<td>http</td>
<td>The port for the HBaseMaster web UI. Set to -1 if you do not want the info server to run.</td>
<td>Yes</td>
<td>hbase.master.info.port</td>
</tr>
<tr>
<td>Region Server</td>
<td>All Slave Node hosts</td>
<td>60020</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Region Server</td>
<td>All Slave Node hosts</td>
<td>60030</td>
<td>http</td>
<td></td>
<td>Yes</td>
<td>hbase.regionserver.port</td>
</tr>
<tr>
<td>HBase REST Server</td>
<td>All REST Servers</td>
<td>8080</td>
<td>http</td>
<td></td>
<td></td>
<td>hbase.rest.port</td>
</tr>
<tr>
<td>HBase REST Server</td>
<td>All REST Servers</td>
<td>8085</td>
<td>http</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HBase Thrift Server</td>
<td>All Thrift Servers</td>
<td>9090</td>
<td></td>
<td></td>
<td></td>
<td>hbase.thrift.server.port</td>
</tr>
<tr>
<td>HBase Thrift Server</td>
<td>All Thrift Servers</td>
<td>9095</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* To change the metastore port, use this `hive` command: `hive --service metastore -p port_number`

### 13.5. ZooKeeper Ports

The following table lists the default ports used by the various ZooKeeper services.

**Table 13.5. HBase Ports**

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZooKeeper Server</td>
<td>All ZooKeeper Node hosts</td>
<td>2888</td>
<td></td>
<td>Port used by ZooKeeper peers to talk to each other. See</td>
<td>No</td>
<td>hbase.zookeeper.peerport</td>
</tr>
<tr>
<td>Service</td>
<td>Servers</td>
<td>Default Ports Used</td>
<td>Protocol</td>
<td>Description</td>
<td>Need End User Access?</td>
<td>Configuration Parameters</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------</td>
<td>--------------------</td>
<td>----------</td>
<td>------------------------------------------------------------------------------</td>
<td>-----------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>ZooKeeper Server</td>
<td>All ZooKeeper Node hosts</td>
<td>3888</td>
<td></td>
<td>Port used by ZooKeeper peers to talk to each other. See here for more information.</td>
<td>No</td>
<td>hbase.zookeeper.leaderport</td>
</tr>
<tr>
<td>ZooKeeper Server</td>
<td>All ZooKeeper Hosts</td>
<td>2181</td>
<td></td>
<td>Property from ZooKeeper's config zoo.cfg. The port at which the clients will connect.</td>
<td>Yes</td>
<td>hbase.zookeeper.property.clientPort</td>
</tr>
</tbody>
</table>

### 13.6. WebHCat Port

The following table lists the default port used by the WebHCat service.

**Table 13.6. WebHCat Port**

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebHCat Server</td>
<td>Any utility machine</td>
<td>50111</td>
<td>http</td>
<td>Web API on top of HCatalog and other Hadoop services</td>
<td>Yes</td>
<td>templeton.port</td>
</tr>
</tbody>
</table>

### 13.7. Ganglia Ports

The following table lists the default ports used by the various Ganglia services.

**Table 13.7. Ganglia Ports**

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ganglia Monitor</td>
<td>All Slave Node hosts</td>
<td>8660</td>
<td></td>
<td>For monitoring (gmond) agents</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Ganglia Server</td>
<td>Ganglia server host</td>
<td>8651</td>
<td></td>
<td>For ganglia gmetad</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*See [Optional: Set Up HTTPS for Ganglia](#) for instructions on enabling HTTPS.*

### 13.8. MySQL Port

The following table lists the default port used by the MySQL service.
### 13.9. Ambari Ports

The following table lists the default ports used by Ambari.

#### Table 13.9. Ambari Web

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambari Server</td>
<td>Ambari Server host</td>
<td>8080(^a)</td>
<td>http(^b)</td>
<td>Interface to Ambari Web and Ambari REST API</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Ambari Server</td>
<td>Ambari Server host</td>
<td>8440</td>
<td>https</td>
<td>Handshake Port for Ambari Agents to Ambari Server</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Ambari Server</td>
<td>Ambari Server host</td>
<td>8441</td>
<td>https</td>
<td>Registration and Heartbeat Port for Ambari Agents to Ambari Server</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

\(^a\)See Optional: Change the Ambari Server Port for instructions on changing the default port.

\(^b\)See Optional: Set Up HTTPS for Ambari Web for instructions on enabling HTTPS.

### 13.10. Nagios Ports

The following table lists the default port used by Ambari Web.

#### Table 13.10. Nagios Web

<table>
<thead>
<tr>
<th>Service</th>
<th>Servers</th>
<th>Default Ports Used</th>
<th>Protocol</th>
<th>Description</th>
<th>Need End User Access?</th>
<th>Configuration Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nagios</td>
<td>Nagios server host</td>
<td>80</td>
<td>http(^a)</td>
<td>Nagios Web UI</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

\(^a\)See Optional: Set Up HTTPS for Nagios for instructions on enabling HTTPS.
Part IV. Additional Tasks with Ambari

This section describes additional tasks you may need to do with your Hadoop installation. It includes:

- Installing Ambari Agents Manually
- Using Custom Hostnames
- Upgrading Operating Systems on an Ambari-based Hadoop Installation
- Moving the Ambari Server
- Using Non-Default Databases
- Setting Up Kerberos for Use with Ambari

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.

### 14.1. RHEL/CentOS/Oracle Linux 5.x and 6.x

1. Install the Ambari Agent

   ```bash
   yum install ambari-agent
   ```

2. Using a text editor, Configure the Ambari Agent by editing the `ambari-agent.ini` file. For example:

   ```bash
   vi /etc/ambari-agent/conf/ambari-agent.ini
   ```
   ```ini
   [server]
   hostname={your.ambari.server.hostname}
   url_port=8440
   secured_url_port=8441
   ```

3. Start the agent. The agent registers with the Server on start.

   ```bash
   ambari-agent start
   ```

### 14.2. SLES

1. Install the Ambari Agent

   ```bash
   zypper install ambari-agent
   ```

2. Configure the Ambari Agent by editing the `ambari-agent.ini` file.

   ```bash
   vi /etc/ambari-agent/conf/ambari-agent.ini
   ```
   ```ini
   [server]
   hostname={your.ambari.server.hostname}
   url_port=8440
   secured_url_port=8441
   ```

3. Start the agent. The agent registers with the Server on start.

   ```bash
   ambari-agent start
   ```
15. Appendix: Using Custom Hostnames

Use the following instructions to use custom hostnames in your cluster:


2. Install the Agents manually as described in Installing Ambari Agents Manually.

3. For every host, create a script (for example named /tmp/hostname.sh) to echo the custom name you want to use for that host. For example:

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

4. With a text editor, open /etc/ambari-agent/conf/ambari-agent.ini on every host and add the following information:

<table>
<thead>
<tr>
<th>Table 15.1. ambari-agent.ini</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Section</strong></td>
</tr>
<tr>
<td>[server]</td>
</tr>
<tr>
<td>[agent]</td>
</tr>
</tbody>
</table>

5. Add the hostnames to /etc/hosts on all nodes.
16. Appendix: Upgrading Operating Systems on an Ambari-based Hadoop Installation

Ambari requires specific versions of the files for components that it uses. There are three steps you should take to make sure that these versions continue to be available:

• Disable automatic OS updates

• Do not update any HDP components such as MySQL, Ganglia, etc.

• If you must perform an OS update, do a manual kernel update only.
17. Appendix: 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 for Ambari (such as Oracle), 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.**

### 17.1. Back up Current Data

1. **Stop the original Ambari Server.**
   ```bash
   ambari-server stop
   ```
2. **Create a directory to hold the database backups.**
   ```bash
   cd /tmp
   mkdir dbdumps
   cd dbdumps/
   ```
3. **Create the database backups.**
   ```bash
   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`, and `MAPRED-PASSWORD`. Defaults are `ambari-server/bigdata` and `mapred/mapred`.

### 17.2. Update Agents

1. **On each Agent node, stop the Agent.**
   ```bash
   ambari-agent stop
   ```
2. **Remove old Agent certificates.**
   ```bash
   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.**
17.3. Install the New Server and Populate the Databases

1. On the new host, install the Server as described in Running the Installer, Sections 2.1 and 2.2.

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.

```
ambavi-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.
18. Appendix: 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 setup the Ambari Server by running `ambari-server setup`.

18.1. Hive/HCatalog

1. On the Hive Metastore machine, install the appropriate JDBC .jar file:
   - For **Oracle**:
        
        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**:
        
        Select Oracle Database 11g Release 2 - ojdbc6.jar
     b. Copy the .jar file to the Java share directory
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-0.10.0.oracle.sql
  ```

  The file `hive-schema-0.10.0.oracle.sql` is found in the `/var/lib/ambari-server/resources/` directory of the Ambari Server machine, once you have completed the Set Up the Bits step in the install process.

### 18.1.1. Troubleshooting Hive/HCatalog

Use these entries to help you troubleshoot any issues you might have installing Hive/HCatalog with non-default databases.
18.1.1.1. 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.

18.1.1.1.1. Solution

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

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

Check the install log:

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

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

18.1.1.2.1. Solution

Complete the install with the warning. Check your database to confirm the Hive Metastore schema is loaded. Once in the Ambari Web GIU, 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.

18.2. Oozie

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

   • For Oracle:
     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:


  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'@'%' ;
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;
```

18.2.1. Troubleshooting Oozie

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

18.2.1.1. 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
```

The MySQL JDBC .jar file cannot be found.

18.2.1.1. Solution

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

18.2.1.2. 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 ]
```

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

18.2.1.2. Solution

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 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
```

18.3. Ambari

1. On the Ambari Server machine, install the Oracle JDBC .jar file:

Select Oracle Database 11g Release 2 - ojdbc6.jar

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

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

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

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

```sql
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.

3. Load the Ambari Server schema:

   • To set up Ambari Server to load the schema automatically:

   a. Download the Oracle Instant Client (for Linux x-86 or x86-64), Basic and the Instant Client Package - SQL*Plus, version 11.2.0.3.0, on the Ambari Server host.

      For information on the Oracle Database Instant Client, see [here](http://www.oracle.com). To download the x86 client, see [here](http://www.oracle.com). To download the x86-64 client, see [here](http://www.oracle.com).

   b. Extract the zip files on your Ambari Server

      ```bash
      mkdir /home/oracle
      cd /home/oracle
      unzip /tmp/instantclientsqlpluslinux.x6411.2.0.3.0.zip
      unzip /tmp/instantclientbasiclinux.x6411.2.0.3.0.zip
      ```

   c. Update your PATH and LD_LIBRARY_PATH variables. For example, in BASH:

      ```bash
      export PATH=/home/oracle/instantclient_11_2:${PATH}
      export LD_LIBRARY_PATH=/home/oracle/instantclient_11_2:
      $$LD_LIBRARYPATH
      ```

   • To load the schema manually, create the Ambari Server schema by running a script:

      ```bash
      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 Set Up the Bits step in the install process.
18.3.1. Troubleshooting Ambari

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

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

Check /var/log/ambari-server/ambari-server.log for:

**ExceptionDescription**: Configuration error.
Class[oracle.jdbc.driver.OracleDriver] not found.

The Oracle JDBC .jar file cannot be found.

18.3.1.1.1. Solution

Make sure the file is in the appropriate directory on the Ambari server and re-run ambari-server setup. See Step one above.

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

Check /var/log/ambari-server/ambari-server.log for:

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

Ambari Server cannot connect to the database.

18.3.1.2.1. Solution

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

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

Check /var/log/ambari-server/ambari-server.log for:

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

You are using an invalid username/password.

18.3.1.3.1. Solution

Confirm the user account is set up in the database and has the correct privileges. See Step 2 above.

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

Check /var/log/ambari-server/ambari-server.log for:
Internal Exception: java.sql.SQLSyntaxErrorException: ORA-00942: table or view does not exist

The schema has not been loaded.

18.3.1.4.1. Soution

Confirm you have loaded the database schema. See Step 3 above.
19. Setting Up Kerberos for Use with Ambari

This section provides information on setting up Kerberos for an Ambari-installed version of Hadoop.

- Setting Up Kerberos for Hadoop 2.x
- Setting Up Kerberos for Hadoop 1.x

19.1. Setting Up Kerberos for Hadoop 2.x

Use the following instructions to prepare to deploy a secure Hadoop cluster version 2.x:

1. Preparing Kerberos
2. Setting Up Hadoop Users
3. Enabling Kerberos Security

19.1.1. Preparing Kerberos

This section provides information on setting up Kerberos.

1. Kerberos Overview
2. Installing and Configuring the KDC
3. Creating the Database
4. Starting the KDC
5. Installing and Configuring the Kerberos Clients
6. Creating Service Principals and Keytab Files for Hadoop
7. Setup Kerberos JAAS Configuration for Ambari

19.1.1.1. Kerberos Overview

Establishing identity with strong authentication is the basis for secure access in Hadoop. Users need to be able to reliably “identify” themselves and then have that identity propagated throughout the Hadoop cluster. Once this is done those users can access resources (such as files or directories) or interact with the cluster (like running MapReduce jobs). As well, Hadoop cluster resources themselves (such as Hosts and Services) need to authenticate with each other to avoid potential malicious systems “posing as” part of the cluster to gain access to data.

To create that secure communication among its various components, Hadoop uses Kerberos. Kerberos is a third party authentication mechanism, in which users and services that users want to access rely on a third party - the Kerberos server - to authenticate each
to the other. The Kerberos server itself is known as the *Key Distribution Center*, or KDC. At a high level, it has three parts:

- A database of the users and services (known as *principals*) that it knows about and their respective Kerberos passwords

- An *authentication server* (AS) which performs the initial authentication and issues a *Ticket Granting Ticket* (TGT)

- A *Ticket Granting Server* (TGS) that issues subsequent service tickets based on the initial TGT

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

Because cluster resources (hosts or services) cannot provide a password each time to decrypt the TGT, they use a special file, called a *keytab*, which contains the resource principal’s authentication credentials.

The set of hosts, users, and services over which the Kerberos server has control is called a *realm*.

### Table 19.1. Kerberos terminology

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Distribution Center, or KDC</td>
<td>The trusted source for authentication in a Kerberos-enabled environment.</td>
</tr>
<tr>
<td>Kerberos KDC Server</td>
<td>The machine, or server, that serves as the Key Distribution Center.</td>
</tr>
<tr>
<td>Kerberos Client</td>
<td>Any machine in the cluster that authenticates against the KDC.</td>
</tr>
<tr>
<td>Principal</td>
<td>The unique name of a user or service that authenticates against the KDC.</td>
</tr>
<tr>
<td>Keytab</td>
<td>A file that includes one or more principals and their keys.</td>
</tr>
<tr>
<td>Realm</td>
<td>The Kerberos network that includes a KDC and a number of Clients.</td>
</tr>
</tbody>
</table>

#### 19.1.1.2. Installing and Configuring the KDC

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

**Note**

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

1. To install a new version of the server:

   [On RHEL, CentOS, or Oracle Linux]
   ```
yum install krb5-server krb5-libs krb5-auth-dialog krb5-workstation
   ```
   OR

   [On SLES]
   ```
   zypper install krb5 krb5-server krb5-client
   ```

   **Note**

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

2. When the server is installed use a text editor to edit the configuration file, located by default here:

   ```
   /etc/krb5.conf
   ```

   Change the `[realms]` section of this file by replacing the default “kerberos.example.com” setting for the `kdc` and `admin_server` properties with the Fully Qualified Domain Name of the KDC server. In this example below, “kerberos.example.com” has been replaced with “my.kdc.server”.

   ```
   [realms]
   EXAMPLE.COM = {
     kdc = my.kdc.server
     admin_server = my.kdc.server
   }
   ```

   **19.1.1.3. Creating the Database**

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

   [on RHEL, CentOS, or Oracle Linux]
   ```
   /usr/sbin/kdb5_util create -s
   ```
   OR

   [on SLES]
   ```
   kdb5_util create -s
   ```

   The `-s` option stores the master server key for the database in a *stash* file. If the stash file is not present, you must log into the KDC with the master password (specified during installation) each time it starts.

   **19.1.1.4. Starting the KDC**

   Start the KDC.

   [on RHEL, CentOS, or Oracle Linux]
   ```
   /etc/rc.d/init.d/krb5kdc start
   /etc/rc.d/init.d/kadmin start
   ```
   OR
19.1.1.5. Installing and Configuring the Kerberos Clients

1. To install the Kerberos clients, on every server in the cluster:

   [on RHEL, CentOS, or Oracle Linux]
   yum install krb5-workstation

   OR

   [on SLES]
   zypper install krb5-client

2. Copy the krb5.conf file you modified in Installing and Configuring the KDC to all the servers in the cluster.

19.1.1.6. Creating Service Principals and Keytab Files for Hadoop 2.x

Each service and sub-service in Hadoop must have its own principal. A principal name in a given realm consists of a primary name and an instance name, which in this case is the FQDN of the host that runs that service. As services do not login with a password to acquire their tickets, their principal's authentication credentials are stored in a keytab file, which is extracted from the Kerberos database and stored locally with the service principal on the service component host.

First you must create the principal, using mandatory naming conventions.

Then you must create the keytab file with that principal's information and copy the file to the keytab directory on the appropriate service host.

Note

Principals can be created either on the KDC machine itself or through the network, using an "admin" principal. The following instructions assume you are using the KDC machine and using the kadmin.local command line administration utility. Using kadmin.local on the KDC machine allows you to create principals without needing to create a separate "admin" principal before you start.

1. Open the kadmin.local utility on the KDC machine

   /usr/sbin/kadmin.local

2. Create the service principals:

   $kadmin.local
   addprinc -randkey $primary_name/$fully.qualified.domain.name@EXAMPLE.COM

   The -randkey option is used to generate the password.

   Note that in the example each service principal's primary name has appended to it the instance name, the FQDN of the host on which it runs. This provides a unique principal
name for services that run on multiple hosts, like DataNodes and NodeManagers. The addition of the hostname serves to distinguish, for example, a request from DataNode A from a request from DataNode B. This is important for two reasons:

- If the Kerberos credentials for one DataNode are compromised, it does not automatically lead to all DataNodes being compromised
- If multiple DataNodes have exactly the same principal and are simultaneously connecting to the NameNode, and if the Kerberos authenticator being sent happens to have same timestamp, then the authentication would be rejected as a replay request.

The principal name must match the values in the table below:

### Table 19.2. Service Principals

<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Mandatory Principal Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS</td>
<td>NameNode</td>
<td><code>nn/$FQDN</code></td>
</tr>
<tr>
<td>HDFS</td>
<td>NameNode HTTP</td>
<td><code>HTTP/$FQDN</code></td>
</tr>
<tr>
<td>HDFS</td>
<td>SecondaryNameNode</td>
<td><code>nn/$FQDN</code></td>
</tr>
<tr>
<td>HDFS</td>
<td>SecondaryNameNode HTTP</td>
<td><code>HTTP/$FQDN</code></td>
</tr>
<tr>
<td>HDFS</td>
<td>DataNode</td>
<td><code>dn/$FQDN</code></td>
</tr>
<tr>
<td>MR2</td>
<td>History Server</td>
<td><code>jhs/$FQDN</code></td>
</tr>
<tr>
<td>MR2</td>
<td>History Server HTTP</td>
<td><code>HTTP/$FQDN</code></td>
</tr>
<tr>
<td>YARN</td>
<td>ResourceManager</td>
<td><code>rm/$FQDN</code></td>
</tr>
<tr>
<td>YARN</td>
<td>NodeManager</td>
<td><code>nm/$FQDN</code></td>
</tr>
<tr>
<td>Oozie</td>
<td>Oozie Server</td>
<td><code>oozie/$FQDN</code></td>
</tr>
<tr>
<td>Oozie</td>
<td>Oozie HTTP</td>
<td><code>HTTP/$FQDN</code></td>
</tr>
<tr>
<td>Hive</td>
<td>Hive Metastore</td>
<td><code>hive/$FQDN</code></td>
</tr>
<tr>
<td>Hive</td>
<td>HiveServer2</td>
<td><code>hive/$FQDN</code></td>
</tr>
<tr>
<td>Hive</td>
<td>WebHCat</td>
<td><code>HTTP/$FQDN</code></td>
</tr>
<tr>
<td>HBase</td>
<td>MasterServer</td>
<td><code>hbase/$FQDN</code></td>
</tr>
<tr>
<td>HBase</td>
<td>RegionServer</td>
<td><code>hbase/$FQDN</code></td>
</tr>
<tr>
<td>ZooKeeper</td>
<td>ZooKeeper</td>
<td><code>zookeeper/$FQDN</code></td>
</tr>
<tr>
<td>Nagios Server</td>
<td>Nagios</td>
<td><code>nagios/$FQDN</code></td>
</tr>
<tr>
<td>JournalNode Server*</td>
<td>JournalNode</td>
<td><code>jn/$FQDN</code></td>
</tr>
</tbody>
</table>

*Only required if you are setting up NameNode HA.

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

```
$ kadmin.local
addprinc -randkey dn/\$DataNode-Host@EXAMPLE.COM
```

3. In addition you must create four special principals for Ambari’s own use.

### Note

The names in table below can be customized in the Customize Services step of the Ambari Install Wizard. If this is the case in your installation, the principal names should match the customized names. For example, if the
HDFS Service User has been set to `hdfs1`, the respective principal for the Ambari HDFS User should also be created as `hdfs1`.

These principals do not have the FQDN appended to the primary name:

**Table 19.3. Ambari Principals**

<table>
<thead>
<tr>
<th>User</th>
<th>Mandatory Principal Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambari User</td>
<td><code>ambari</code></td>
</tr>
<tr>
<td>Ambari Smoke Test User</td>
<td><code>ambari-qa</code></td>
</tr>
<tr>
<td>Ambari HDFS User</td>
<td><code>hdfs</code></td>
</tr>
<tr>
<td>Ambari HBase User</td>
<td><code>hbase</code></td>
</tr>
</tbody>
</table>

This principal is used with the JAAS configuration. See [Setup Kerberos JAAS Configuration for Ambari](#) for more information.

4. Once the principals are created in the database, you can extract the related keytab files for transfer to the appropriate host:

```
$kadmin.local
xst -norandkey -k $keytab_file_name $primary_name/fully.qualified.domain.name@EXAMPLE.COM
```

You must use the mandatory names for the `$keytab_file_name` variable shown in this table.

**Note**

Some older versions of Kerberos do not support the `xst -norandkey` option. You can use the command without the `-norandkey` flag, except in cases where you need to copy a principal from one keytab file to another keytab file on a host. This might be a requirement if the Hadoop configurations on a host have keytab path properties that point to different keytab locations but have corresponding principal name properties that have the same values.

In situations like this, you can use the two step `kadmin/kutil` procedure. This description assumes MIT Kerberos. If you are using another version, please check your documentation.

a. Extract the keytab file information:

```
$kadmin
xst -k $keytab_file_name-temp1 $primary_name/fully.qualified.domain.name@EXAMPLE.COM
xst -k $keytab_file_name-temp2 $primary_name/fully.qualified.domain.name@EXAMPLE.COM
```

b. Write the keytab to a file:

```
$kutil
kutil: rkt $keytab_file_name-temp1
kutil: rkt $keytab_file_name-temp2
kutil: wkt $keytab_file_name
kutil: clear
```
### Table 19.4. Service Keytab File Names

<table>
<thead>
<tr>
<th>Component</th>
<th>Principal Name</th>
<th>Mandatory Keytab File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode</td>
<td>nn/$FQDN</td>
<td>nn.service.keytab</td>
</tr>
<tr>
<td>NameNode HTTP</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>SecondaryNameNode</td>
<td>nn/$FQDN</td>
<td>nn.service.keytab</td>
</tr>
<tr>
<td>SecondaryNameNode HTTP</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>DataNode</td>
<td>dn/$FQDN</td>
<td>dn.service.keytab</td>
</tr>
<tr>
<td>MR2 History Server</td>
<td>jhs/$FQDN</td>
<td>nm.service.keytab</td>
</tr>
<tr>
<td>MR2 History Server HTTP</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>YARN</td>
<td>rm/$FQDN</td>
<td>rm.service.keytab</td>
</tr>
<tr>
<td>YARN</td>
<td>nm/$FQDN</td>
<td>nm.service.keytab</td>
</tr>
<tr>
<td>Oozie Server</td>
<td>oozie/$FQDN</td>
<td>oozie.service.keytab</td>
</tr>
<tr>
<td>Oozie HTTP</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>Hive Metastore</td>
<td>hive/$FQDN</td>
<td>hive.service.keytab</td>
</tr>
<tr>
<td>HiveServer2</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>HBase Master Server</td>
<td>hbase/$FQDN</td>
<td>hbase.service.keytab</td>
</tr>
<tr>
<td>HBase RegionServer</td>
<td>hbase/$FQDN</td>
<td>hbase.service.keytab</td>
</tr>
<tr>
<td>ZooKeeper</td>
<td>zookeeper/$FQDN</td>
<td>zk.service.keytab</td>
</tr>
<tr>
<td>Nagios Server</td>
<td>nagios/$FQDN</td>
<td>nagios.service.keytab</td>
</tr>
<tr>
<td>Journal Server&lt;sup&gt;a&lt;/sup&gt;</td>
<td>jn/$FQDN</td>
<td>jn.service.keytab</td>
</tr>
<tr>
<td>Ambari User</td>
<td>ambari</td>
<td>ambari.keytab</td>
</tr>
<tr>
<td>Ambari Smoke Test User</td>
<td>ambari-qa</td>
<td>smokeuser.headless.keytab</td>
</tr>
<tr>
<td>Ambari HDFS User</td>
<td>hdfs</td>
<td>hdfs.headless.keytab</td>
</tr>
<tr>
<td>Ambari HBase User</td>
<td>hbase</td>
<td>hbase.headless.keytab</td>
</tr>
</tbody>
</table>

<sup>a</sup>Only required if you are setting up NameNode HA.

<sup>b</sup>This principal is used with the JAAS configuration. See [Setup Kerberos JAAS Configuration for Ambari](#) for more information.

**For example:** To create the keytab files for NameNode HTTP, issue this command:

```bash
$ kadmin.local
xst -norandkey -k spnego.service.keytab HTTP/<namenode-host>
```

**Note**

If you have a large cluster, you may want to create a script to automate creating your principals and keytabs. To help with that, you can download a CSV-formatted file of all the required principal names and keytab files from the Ambari Web GUI. Select Admin view->Security->Enable Security-> and run the Enable Security Wizard, using the default values. At the bottom of the third page, Create Principals and Keytabs, click Download CSV.

5. When the keytab files have been created, on each host create a directory for them and set appropriate permissions.

```bash
mkdir -p /etc/security/keytabs/
chown root:hadoop /etc/security/keytabs
```
chmod 750 /etc/security/keytabs

6. Copy the appropriate keytab file to each host. If a host runs more than one component (for example, both NodeManager and DataNode), copy keytabs for both components. The Ambari Smoke Test User, the Ambari HDFS User, and the Ambari HBase User keytabs should be copied to the all hosts on the cluster.

Important

If you have customized service user names, replace the default values below with your appropriate service user, group, and keytab names.

7. Set appropriate permissions for the keytabs.

   a. Optionally, if you have Setup Kerberos JAAS Configuration for Ambari on the Ambari server host:

```
chown ambari:ambari /etc/security/keytabs/ambari.keytab
chmod 400 /etc/security/keytabs/ambari.keytab
```

   b. On the HDFS NameNode and SecondaryNameNode hosts:

```
chown hdfs:hadoop /etc/security/keytabs/nn.service.keytab
chmod 400 /etc/security/keytabs/nn.service.keytab
chown root:hadoop /etc/security/keytabs/spnego.service.keytab
chmod 440 /etc/security/keytabs/spnego.service.keytab
```

   c. On the HDFS NameNode host, for the Ambari Test Users:

```
chown ambari-qa:hadoop /etc/security/keytabs/smokeuser.headless.keytab
chmod 440 /etc/security/keytabs/smokeuser.headless.keytab
chown hdfs:hadoop /etc/security/keytabs/hdfs.headless.keytab
chmod 440 /etc/security/keytabs/hdfs.headless.keytab
chown hbase:hadoop /etc/security/keytabs/hbase.headless.keytab
chmod 440 /etc/security/keytabs/hbase.headless.keytab
```

   d. On each host that runs an HDFS DataNode:

```
chown hdfs:hadoop /etc/security/keytabs/dn.service.keytab
chmod 400 /etc/security/keytabs/dn.service.keytab
```

   e. On the host that runs the MR2 History Server:

```
chown mapred:hadoop /etc/security/keytabs/jhs.service.keytab
chmod 400 /etc/security/keytabs/jhs.service.keytab
chown root:hadoop /etc/security/keytabs/spnego.service.keytab
chmod 440 /etc/security/keytabs/spnego.service.keytab
```

   f. On the host that runs the YARN ResourceManager:

```
chown yarn:hadoop /etc/security/keytabs/rm.service.keytab
chmod 400 /etc/security/keytabs/rm.service.keytab
```

   g. On each host that runs a YARN NodeManager:

```
chown yarn:hadoop /etc/security/keytabs/nm.service.keytab
chmod 400 /etc/security/keytabs/nm.service.keytab
```
h. On the host that runs the Oozie Server:

```bash
chown oozie:hadoop /etc/security/keytabs/oozie.service.keytab
chmod 400 /etc/security/keytabs/oozie.service.keytab
chown root:hadoop /etc/security/keytabs/spnego.service.keytab
chmod 440 /etc/security/keytabs/spnego.service.keytab
```

i. On the host that runs the Hive Metastore, HiveServer2 and WebHCat:

```bash
chown hive:hadoop /etc/security/keytabs/hive.service.keytab
chmod 400 /etc/security/keytabs/hive.service.keytab
chown root:hadoop /etc/security/keytabs/spnego.service.keytab
chmod 440 /etc/security/keytabs/spnego.service.keytab
```

j. On hosts that run the HBase MasterServer, RegionServer and ZooKeeper:

```bash
chown hbase:hadoop /etc/security/keytabs/hbase.service.keytab
chmod 400 /etc/security/keytabs/hbase.service.keytab
chown zookeeper:hadoop /etc/security/keytabs/zk.service.keytab
chmod 400 /etc/security/keytabs/zk.service.keytab
```

k. On the host that runs the Nagios server:

```bash
chown nagios:nagios /etc/security/keytabs/nagios.service.keytab
chmod 400 /etc/security/keytabs/nagios.service.keytab
```

l. On each host that runs a JournalNode, if you are setting up NameNode HA:

```bash
chown hdfs:hadoop /etc/security/keytabs/jn.service.keytab
chmod 400 /etc/security/keytabs/jn.service.keytab
```

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

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

Do this on each respective service in your cluster.

**19.1.1.7. Setup Kerberos JAAS Configuration for Ambari**

If you want to set up JAAS configuration for Ambari so that independent access to native Hadoop GUIs like the NameNode UI are secure, use the [Apache community docs](https://cwiki.apache.org/confluence/display/ambari/Ambari+Kerberos+JAAS) to setup your configurations, and then do the following:

1. Log into the Ambari server host.

   **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.

2. Run the special setup command and answer the prompts:

   ```bash
   ambari-server setup-security
   ```

   a. Select 5 for **Setup Ambari kerberos JAAS configuration**.
b. Enter the Kerberos principal name for the Ambari server you set up here.

c. Enter the path to the keytab for Ambari's principal.

d. Restart Ambari Server:

```
  ambari-server restart
```

19.1.2. Setting Up Hadoop Users

This section provides information on setting up Hadoop users for Kerberos.

- **Overview**
- **Creating Mappings Between Principals and UNIX Usernames**

19.1.2.1. Overview

Hadoop uses users' group memberships at various places for things like determining group ownership for files or for access control. To configure Hadoop for use with Kerberos and Ambari you must create a mapping between service principals and these UNIX usernames.

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

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

19.1.2.2. Creating Mappings Between Principals and UNIX Usernames

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

The default rule is simply named `DEFAULT`. It translates all principals in your default domain to their first component. For example, `myusername@APACHE.ORG` and `myusername/admin@APACHE.ORG` both become `myusername`, assuming your default domain is `APACHE.ORG`.

Use the following instructions to configure the mappings between principals and UNIX usernames:

1. **Creating Rules**
2. **Examples**

19.1.2.2.1. Creating Rules

- **Simple Rules**
To make a simple map between principal names and UNIX users, you create a straightforward substitution rule. For example, to map the ResourceManager(TBD) and NodeManager(TBD) principals in the EXAMPLE.COM realm to the UNIX $YARN_USER user and the NameNode(nn) and DataNode(dn) principals to the UNIX $HDFS_USER user, you would make this the value for the hadoop.security.auth_to_local key in core-site.xml.

```
RULE: [2:$1@$0](jt@.*EXAMPLE.COM) s/.*/$YARN_USER /
RULE: [2:$1@$0](ndn@.*EXAMPLE.COM) s/.*/$HDFS_USER/
DEFAULT
```

• Complex Rules

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

• The Base:

The base begins with the number of components in the principal name (excluding the realm), followed by a colon, and the pattern for building the username from the sections of the principal name. In the pattern section $0 translates to the realm, $1 translates to the first component and $2 to the second component.

For example:

```
[1:$1@$0] translates myusername@APACHE.ORG to myusername@APACHE.ORG
[2:$1] translates myusername/admin@APACHE.ORG to myusername
[2:$1%$2] translates myusername/admin@APACHE.ORG to myusername%admin
```

• The Filter:

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

For example:

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

• The Substitution:

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

For example:

```
s/@ACME\..COM// removes the first instance of @SOME.DOMAIN.
s/[@A-Z]*\..COM// removes the first instance of @ followed by a name followed by COM.
s/X/Y/g replaces all of the X in the name with Y
```
19.1.2.2. Examples

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

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

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

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

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

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

19.1.3. Enabling Kerberos Security

To turn on Kerberos-based security in the Ambari Web GUI you must:

1. Have already set up Kerberos for your cluster. For more information, see Setting Up Kerberos for Hadoop 2.x.

2. Go to the Admin tab.


4. Click Enable Security and follow the Enable Security Wizard.

a. Get Started: This step just reminds you that you need to set up Kerberos before you start.

b. Configure Services: This step asks you for your Kerberos information: principals and paths to keytabs, path to your Kerberos tools, realm names and so on. For
more information about a specific field, hover over it, and a popup with a definition appears.

c. **Create Principals and Keytabs**: Use this step to check that all your information is correct. Click **Back** to make any changes. Click **Apply** when you are satisfied with the assignments.

**Note**

If you have a large cluster, you may want to go to the **Create Principals and Keytabs** step first. Step through the wizard accepting the defaults to get to the appropriate page. On the page, use the **Download CSV** button to get a list of all the necessary principals and keytabs in CSV form, which can be used to set up a script. The list includes hostname, principal description, principal name, keytab user, keytab group, keytab permissions, absolute keytab path, and keytab filename.

d. **Save and Apply Configuration**: This step displays a bar showing the progress of integrating the Kerberos information into your Ambari Server.

### 19.2. Setting Up Kerberos for Hadoop 1.x

- **Preparing Kerberos**
- **Setting Up Hadoop Users**
- **Enabling Kerberos Security**
19.2.1. Preparing Kerberos

This section provides information on setting up Kerberos.

1. Kerberos Overview
2. Installing and Configuring the KDC
3. Creating the Database
4. Starting the KDC
5. Installing and Configuring the Kerberos Clients
6. Creating Service Principals and Keytab Files for Hadoop
7. Setup Kerberos JAAS Configuration for Ambari

19.2.1.1. Kerberos Overview

Establishing identity with strong authentication is the basis for secure access in Hadoop. Users need to be able to reliably "identify" themselves and then have that identity propagated throughout the Hadoop cluster. Once this is done those users can access resources (such as files or directories) or interact with the cluster (like running MapReduce jobs). As well, Hadoop cluster resources themselves (such as Hosts and Services) need to authenticate with each other to avoid potential malicious systems "posing as" part of the cluster to gain access to data.

To create that secure communication among its various components, Hadoop uses Kerberos. Kerberos is a third party authentication mechanism, in which users and services that users want to access rely on a third party - the Kerberos server - to authenticate each to the other. The Kerberos server itself is known as the Key Distribution Center, or KDC. At a high level, it has three parts:

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

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

Because cluster resources (hosts or services) cannot provide a password each time to decrypt the TGT, it uses a special file, called a keytab, which contains the resource principal's authentication credentials.

The set of hosts, users, and services over which the Kerberos server has control is called a realm.
Table 19.5. Kerberos terminology

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Distribution Center, or KDC</td>
<td>The trusted source for authentication in a Kerberos-enabled environment.</td>
</tr>
<tr>
<td>Kerberos KDC Server</td>
<td>The machine, or server, that serves as the Key Distribution Center.</td>
</tr>
<tr>
<td>Kerberos Client</td>
<td>Any machine in the cluster that authenticates against the KDC.</td>
</tr>
<tr>
<td>Principal</td>
<td>The unique name of a user or service that authenticates against the KDC.</td>
</tr>
<tr>
<td>Keytab</td>
<td>A file that includes one or more principals and their keys.</td>
</tr>
<tr>
<td>Realm</td>
<td>The Kerberos network that includes a KDC and a number of Clients.</td>
</tr>
</tbody>
</table>

19.2.1.2. Installing and Configuring the KDC

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

Note

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

1. To install a new version of the server:

   [On RHEL, CentOS, or Oracle Linux]
   
   ```bash
   yum install krb5-server krb5-libs krb5-auth-dialog krb5-workstation
   ```
   OR

   [On SLES]
   
   ```bash
   zypper install krb5 krb5-server krb5-client
   ```

   Note

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

2. When the server is installed use a text editor to edit the configuration file, located by default here:

   ```bash
   /etc/krb5.conf
   ```

   Change the [realms] section of this file by replacing the default “kerberos.example.com” setting for the kdc and admin_server properties with the Fully Qualified Domain Name of the KDC server. In this example below, “kerberos.example.com” has been replaced with “my.kdc.server”.

   [realms]
EXAMPLE.COM = {
  kdc = my.kdc.server
  admin_server = my.kdc.server
}

19.2.1.3. Creating the Database

Use the utility kdb5_util to create the Kerberos database.

[on RHEL, CentOS, or Oracle Linux]
/usr/sbin/kdb5_util create -s

OR

[on SLES]
kdb5_util create -s

The -s option stores the master server key for the database in a stash file. If the stash file is not present, you must log into the KDC with the master password (specified during installation) each time it starts.

19.2.1.4. Starting the KDC

Start the KDC.

[on RHEL, CentOS, or Oracle Linux]
/etc/rc.d/init.d/krb5kdc start
/etc/rc.d/init.d/kadmin start

OR

[on SLES]
rckrb5kdc start
rckadmin start

19.2.1.5. Installing and Configuring the Kerberos Clients

1. To install the Kerberos clients, on every server in the cluster:

   [on RHEL, CentOS, or Oracle Linux]
yum install krb5-workstation

   OR

   [on SLES]
zypper install krb5-client

2. Copy the krb5.conf file you modified in Installing and Configuring the KDC to all the servers in the cluster.

19.2.1.6. Creating Service Principals and Keytab Files for Hadoop

Each service and sub-service in Hadoop must have its own principal. A principal name in a given realm consists of a primary name and an instance name, which in this case is the FQDN of the host that runs that service. As services do not login with a password to acquire their tickets, their principal's authentication credentials are stored in a keytab file, which is
extracted from the Kerberos database and stored locally with the service principal on the service component host.

First you must create the principal, using mandatory naming conventions.

Then you must create the keytab file with that principal's information and copy the file to the keytab directory on the appropriate service host.

### Note

Principals can be created either on the KDC machine itself or through the network, using an “admin” principal. The following instructions assume you are using the KDC machine and using the `kadmin.local` command line administration utility. Using `kadmin.local` on the KDC machine allows you to create principals without needing to create a separate "admin" principal before you start.

1. Open the `kadmin.local` utility on the KDC machine

   `/usr/sbin/kadmin.local`

2. Create the service principals:

   ```bash
   $kadmin.local addprinc -randkey $primary_name/$fully.qualified.domain.name@EXAMPLE.COM
   ```

   The `-randkey` option is used to generate the password.

   Note that in the example each service principal's primary name has appended to it the instance name, the FQDN of the host on which it runs. This provides a unique principal name for services that run on multiple hosts, like DataNodes and TaskTrackers. The addition of the hostname serves to distinguish, for example, a request from DataNode A from a request from DataNode B. This is important for two reasons:

   - If the Kerberos credentials for one DataNode are compromised, it does not automatically lead to all DataNodes being compromised
   - If multiple DataNodes have exactly the same principal and are simultaneously connecting to the NameNode, and if the Kerberos authenticator being sent happens to have same timestamp, then the authentication would be rejected as a replay request.

   The principal name must match the values in the table below:

**Table 19.6. Service Principals**

<table>
<thead>
<tr>
<th>Service</th>
<th>Component</th>
<th>Mandatory Principal Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS</td>
<td>NameNode</td>
<td>nn/$FQDN</td>
</tr>
<tr>
<td>HDFS</td>
<td>NameNode HTTP</td>
<td>HTTP/$FQDN</td>
</tr>
<tr>
<td>HDFS</td>
<td>SecondaryNameNode</td>
<td>nn/$FQDN</td>
</tr>
<tr>
<td>HDFS</td>
<td>SecondaryNameNode HTTP</td>
<td>HTTP/$FQDN</td>
</tr>
<tr>
<td>HDFS</td>
<td>DataNode</td>
<td>dn/$FQDN</td>
</tr>
<tr>
<td>MapReduce</td>
<td>JobTracker</td>
<td>jt/$FQDN</td>
</tr>
<tr>
<td>MapReduce</td>
<td>TaskTracker</td>
<td>tt/$FQDN</td>
</tr>
</tbody>
</table>
For example: To create the principal for a DataNode service, issue this command:

```bash
$ kadmin.local
ddprinc -randkey dn/$DataNode-Host@EXAMPLE.COM
```

3. In addition you must create four special principals for Ambari’s own use.

**Note**

The names in table below can be customized in the Customize Services step of the Ambari Install Wizard. If this is the case in your installation, the principal names should match the customized names. For example, if the HDFS Service User has been set to `hdfs1`, the respective principal for the Ambari HDFS User should also be created as `hdfs1`.

These principals do not need the FQDN appended to the primary name:

### Table 19.7. Ambari Principals

<table>
<thead>
<tr>
<th>User</th>
<th>Default Principal Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambari User</td>
<td>ambari</td>
</tr>
<tr>
<td>Ambari Smoke Test User</td>
<td>ambari-qa</td>
</tr>
<tr>
<td>Ambari HDFS User</td>
<td>hdfs</td>
</tr>
<tr>
<td>Ambari HBase User</td>
<td>hbase</td>
</tr>
</tbody>
</table>

*This principal is used with the JAAS configuration. See [Setup Kerberos JAAS Configuration for Ambari](#) for more information.*

4. Once the principals are created in the database, you can extract the related keytab files for transfer to the appropriate host:

```bash
$ kadmin.local
xst -norandkey -k $keytab_file_name $primary_name/fully.qualified.domain.name@EXAMPLE.COM
```

**Note**

Some older versions of Kerberos do not support the `xst -norandkey` option. You can use the command without the `-norandkey` flag, except in cases where you need to merge two principals with the same name into a single keytab file for a single host. In this case, you can use the two step
kadmin/kutil procedure. This description assumes MIT Kerberos. If you are using another version, please check your documentation.

a. Extract the keytab file information:

```
$kadmin
xst -k $keytab_file_name-temp1 $primary_name/fully.qualified.
domain.name@EXAMPLE.COM
xst -k $keytab_file_name-temp2 $primary_name/fully.qualified.
domain.name@EXAMPLE.COM
```

b. Merge the keytabs into a single file:

```
$kutil
rkt $keytab_file_name-temp1
rkt $keytab_file_name-temp2
wkt $keytab_file_name
clear
```

You must use the mandatory names for the `keytab_file_name` variable shown in this table. Adjust the principal names if necessary.

### Table 19.8. Service Keytab File Names

<table>
<thead>
<tr>
<th>Component</th>
<th>Principal Name</th>
<th>Mandatory Keytab File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NameNode</td>
<td>nn/$FQDN</td>
<td>nn.service.keytab</td>
</tr>
<tr>
<td>NameNode HTTP</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>SecondaryNameNode</td>
<td>nn/$FQDN</td>
<td>nn.service.keytab</td>
</tr>
<tr>
<td>SecondaryNameNode HTTP</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>DataNode</td>
<td>dn/$FQDN</td>
<td>dn.service.keytab</td>
</tr>
<tr>
<td>JobTracker</td>
<td>jt/$FQDN</td>
<td>jt.service.keytab</td>
</tr>
<tr>
<td>TaskTracker</td>
<td>tt/$FQDN</td>
<td>tt.service.keytab</td>
</tr>
<tr>
<td>Oozie Server</td>
<td>oozie/$FQDN</td>
<td>oozie.service.keytab</td>
</tr>
<tr>
<td>Oozie HTTP</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>Hive Metastore</td>
<td>hive/$FQDN</td>
<td>hive.service.keytab</td>
</tr>
<tr>
<td>HiveServer2</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>WebHCat</td>
<td>HTTP/$FQDN</td>
<td>spnego.service.keytab</td>
</tr>
<tr>
<td>HBase Master Server</td>
<td>hbase/$FQDN</td>
<td>hbase.service.keytab</td>
</tr>
<tr>
<td>HBase RegionServer</td>
<td>hbase/$FQDN</td>
<td>hbase.service.keytab</td>
</tr>
<tr>
<td>ZooKeeper</td>
<td>zookeeper/$FQDN</td>
<td>zk.service.keytab</td>
</tr>
<tr>
<td>Nagios Server</td>
<td>nagios/$FQDN</td>
<td>nagios.service.keytab</td>
</tr>
<tr>
<td>Ambari User</td>
<td>ambari</td>
<td>ambari.keytab</td>
</tr>
<tr>
<td>Ambari Smoke Test User</td>
<td>ambari-qa</td>
<td>smokeuser.headless.keytab</td>
</tr>
<tr>
<td>Ambari HDFS User</td>
<td>hdfs</td>
<td>hdfs.headless.keytab</td>
</tr>
<tr>
<td>Ambari HBase User</td>
<td>hbase</td>
<td>hbase.headless.keytab</td>
</tr>
</tbody>
</table>

*This principal is used with the JAAS configuration. See Setup Kerberos JAAS Configuration for Ambari for more information.

For example: To create the keytab files for NameNode HTTP, issue this command:

```
xst -norandkey -k spnego.service.keytab HTTP/<namenode-host>
```
Note

If you have a large cluster, you may want to create a script to automate creating your principals and keytabs. The Ambari Web GUI can help. See Create Principals and Keytabs [150] for more information.

5. When the keytab files have been created, on each host create a directory for them and set appropriate permissions.

```bash
mkdir -p /etc/security/keytabs/
chown root:hadoop /etc/security/keytabs
chmod 750 /etc/security/keytabs
```

6. Copy the appropriate keytab file to each host. If a host runs more than one component (for example, both TaskTracker and DataNode), copy keytabs for both components. The Ambari Smoke Test User, the Ambari HDFS User, and the Ambari HBase User keytabs should be copied to all hosts.

7. Set appropriate permissions for the keytabs.

Important

If you have customized service user names, replace the default values below with your appropriate service user, group, and keytab names.

a. Optionally, if you have Setup Kerberos JAAS Configuration for Ambari on the Ambari server host:

```bash
chown ambari:ambari /etc/security/keytabs/ambari.keytab
chmod 400 /etc/security/keytabs/ambari.keytab
```

b. On the HDFS NameNode and SecondaryNameNode hosts:

```bash
chown hdfs:hadoop /etc/security/keytabs/nn.service.keytab
chmod 400 /etc/security/keytabs/nn.service.keytab
chown root:hadoop /etc/security/keytabs/spnego.service.keytab
chmod 440 /etc/security/keytabs/spnego.service.keytab
```

c. On the HDFS NameNode host, for the Ambari Test Users:

```bash
chown ambari-qa:hadoop /etc/security/keytabs/smokeuser.headless.keytab
chmod 440 /etc/security/keytabs/smokeuser.headless.keytab
chown hdfs:hadoop /etc/security/keytabs/hdfs.headless.keytab
chmod 440 /etc/security/keytabs/hdfs.headless.keytab
chown hbase:hadoop /etc/security/keytabs/hbase.headless.keytab
chmod 440 /etc/security/keytabs/hbase.headless.keytab
```

d. On each host that runs an HDFS DataNode:

```bash
chown hdfs:hadoop /etc/security/keytabs/dn.service.keytab
chmod 400 /etc/security/keytabs/dn.service.keytab
```

e. On the host that runs the MapReduce JobTracker:

```bash
chown mapred:hadoop /etc/security/keytabs/jt.service.keytab
chmod 400 /etc/security/keytabs/jt.service.keytab
```
f. On each host that runs a MapReduce TaskTracker:

```
chown mapred:hadoop /etc/security/keytabs/tt.service.keytab
chmod 400 /etc/security/keytabs/tt.service.keytab
```

```
g. On the host that runs the Oozie Server:

chown oozie:hadoop /etc/security/keytabs/oozie.service.keytab
chmod 400 /etc/security/keytabs/oozie.service.keytab
chown root:hadoop /etc/security/keytabs/spnego.service.keytab
chmod 440 /etc/security/keytabs/spnego.service.keytab
```

h. On the host that runs the Hive Metastore, HiveServer2 and WebHCat:

```
chown hive:hadoop /etc/security/keytabs/hive.service.keytab
chmod 400 /etc/security/keytabs/hive.service.keytab
chown root:hadoop /etc/security/keytabs/spnego.service.keytab
chmod 440 /etc/security/keytabs/spnego.service.keytab
```

i. On hosts that run the HBase MasterServer, RegionServer and ZooKeeper:

```
chown hbase:hadoop /etc/security/keytabs/hbase.service.keytab
chmod 400 /etc/security/keytabs/hbase.service.keytab
chown zookeeper:hadoop /etc/security/keytabs/zk.service.keytab
chmod 400 /etc/security/keytabs/zk.service.keytab
```

j. On the host that runs the Nagios server:

```
chown nagios:nagios /etc/security/keytabs/nagios.service.keytab
chmod 400 /etc/security/keytabs/nagios.service.keytab
```

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

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

Do this on each respective service in your cluster.

### 19.2.1.7. Setup Kerberos JAAS Configuration for Ambari

If you want to set up JAAS configuration for Ambari so that independent access to native Hadoop GUIs like the NameNode UI are secure, use the Apache community docs to setup your configurations, and then do the following:

**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.

Run the special setup command and answer the prompts

```
ambari-server setup-security
```

1. Select 5 for **Setup Ambari kerberos JAAS configuration**.

2. Enter the Kerberos principal name for the Ambari server you set up here.
3. Enter the path to the keytab for Ambari’s principal.

4. Restart Ambari Server:

   ```
   ambari-server restart
   ```

For more information

## 19.2.2. Setting Up Hadoop Users

This section provides information on setting up Hadoop users for Kerberos.

- **Overview**
  - Creating Mappings Between Principals and UNIX Usernames

### 19.2.2.1. Overview

Hadoop uses users’ group memberships at various places for things like determining group ownership for files or for access control. To configure Hadoop for use with Kerberos and Ambari you must create a mapping between service principals and these UNIX usernames.

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

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

### 19.2.2.2. Creating Mappings Between Principals and UNIX Usernames

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

The default rule is simply named `DEFAULT`. It translates all principals in your default domain to their first component. For example, `myusername@APACHE.ORG` and `myusername/admin@APACHE.ORG` both become `myusername`, assuming your default domain is `APACHE.ORG`.

Use the following instructions to configure the mappings between principals and UNIX usernames:

1. Creating Rules
2. Examples

#### 19.2.2.2.1. Creating Rules

- Simple Rules
To make a simple map between principal names and UNIX users, you create a straightforward substitution rule. For example, to map the JobTracker(jt) and TaskTracker(tt) principals in the EXAMPLE.COM realm to the UNIX $MAPREDUCE_USER user and the NameNode(nn) and DataNode(dn) principals to the UNIX $HDFS_USER user, you would make this the value for the hadoop.security.auth_to_local key in core-site.xml.

```
RULE:[2:$1@$0]([jt]t@.*EXAMPLE.COM)s/.*/$MAPREDUCE_USER/
RULE:[2:$1@$0]([nd]n@.*EXAMPLE.COM)s/.*/$HDFS_USER/
DEFAULT
```

- **Complex Rules**

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

- **The Base:**

  The base begins with the number of components in the principal name (excluding the realm), followed by a colon, and the pattern for building the username from the sections of the principal name. In the pattern section $0 translates to the realm, $1 translates to the first component and $2 to the second component.

  For example:

  `[1:$1@$0] translates myusername@APACHE.ORG to myusername@APACHE.ORG`  
  `[2:$1] translates myusername/admin@APACHE.ORG to myusername`  
  `[2:$1%$2] translates myusername/admin@APACHE.ORG to myusername%admin`

- **The Filter:**

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

  For example:

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

- **The Substitution:**

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

  For example:

  `s/@ACME\.COM// removes the first instance of @SOME.DOMAIN.`  
  `s/[@A-Z]*\..COM// removes the first instance of @ followed by a name followed by COM.`  
  `s/X/Y/g replaces all of the X in the name with Y`
19.2.2.2. Examples

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

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

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

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

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

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

19.2.3. Enabling Kerberos Security

To turn on Kerberos-based security in the Ambari Web GUI you must:

1. Have already set up Kerberos for your cluster. For more information, see Setting Up Kerberos for Hadoop 1.x.

2. Go to the Admin tab.

3. Click Security.

4. Click Enable Security and follow the Add security wizard.

   a. Get Started: This step just reminds you that you need to set up Kerberos before you start.

   b. Configure Services: This step asks you for your Kerberos information: principals and paths to keytabs, path to your Kerberos tools, realm names and so on. For more information about a specific field, hover over it, and a popup with a definition appears.
c. **Create Principals and Keytabs:** Use this step to check that all your information is correct. Click **Back** to make any changes. Click **Apply** when you are satisfied with the assignments.

**Note**

If you have a large cluster, you may want to go to the **Create Principals and Keytabs** step first. Step through the wizard accepting the defaults to get to the appropriate page. On the page, use the **Download CSV** button to get a list of all the necessary principals and keytabs in CSV form, which can be used to set up a script. The list includes hostname, principal description, principal name, keytab user, keytab group, keytab permissions, absolute keytab path, and keytab filename.

![Create Principals and Keytabs](image)

**d. Save and Apply Configuration:** This step displays a bar showing the progress of integrating the Kerberos information into your Ambari Server.