Hortonworks Data Platform

Ambari Reference Guide

(Apr 13, 2015)

docs.hortonworks.com

Hortonworks Data Platform : Ambari Reference Guide

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

Unlike other providers of platforms built using Apache Hadoop, Hortonworks contributes 100% of our code back to the Apache Software Foundation. The Hortonworks Data Platform is Apache-licensed and completely open source. We sell only expert technical support, training and partner-enablement services. All of our technology is, and will remain free and open source. Please visit the Hortonworks Data Platform page for more information on Hortonworks technology. For more information on Hortonworks services, please visit either the Support or Training page. Feel free to Contact Us directly to discuss your specific needs.

Table of Contents

1. Installing Ambari Agents Manually

Involves two steps:

1. Download the Ambari Repo
2. Install Ambari Agents

1.1. Download the Ambari Repo

Select the OS family running on your installation host.

RHEL/CentOS/Oracle Linux 6

On a server host that has Internet access, use a command line editor to perform the following steps:

1. Log in to your host as `root`.
2. Download the Ambari repository file to a directory on your installation host.

```
wget -nv http://public-repo-1.hortonworks.com/ambari/
centos6/2.x/updates/2.0.1/ambari.repo -O /etc/yum.repos.d/
ambari.repo
```

Important

Do not modify the `ambari.repo` file name. This file is expected to be available on the Ambari Server host during Agent registration.

3. Confirm that the repository is configured by checking the repo list.

```
yum repolist
```

You should see values similar to the following for Ambari repositories in the list.

Version values vary, depending on the installation.

repo id	repo name	status
AMBARI.2.0.1-2.x	Ambari 2.x	8
base	CentOS-6 - Base	6,518
extras	CentOS-6 - Extras	37
updates	CentOS-6 - Updates	785

4. Install the Ambari bits. This also installs the default PostgreSQL Ambari database.

```
yum install ambari-server
```

5. Enter y when prompted to to confirm transaction and dependency checks.

A successful installation displays output similar to the following:

Installing : postgresql-libs-8.4.20-1.el6_5.x86_64 1/4

Installing : postgresql-8.4.20-1.el6_5.x86_64 2/4

Installing : postgresql-server-8.4.20-1.el6_5.x86_64 3/4

Installing : ambari-server-2.0.1-147.noarch 4/4

Verifying : postgresql-server-8.4.20-1.el6_5.x86_64 1/4

Verifying : postgresql-libs-8.4.20-1.el6_5.x86_64 2/4

Verifying : postgresql-8.4.20-1.el6_5.x86_64 3/4

Verifying : ambari-server-2.0.1-147.noarch 4/4

Installed : ambari-server.noarch 0:2.0.1-59

Dependency

Installed :

postgresql.x86_64 0:8.4.20-1.el6_5

postgresql-libs.x86_64 0:8.4.20-1.el6_5

postgresql-server.x86_64 0:8.4.20-1.el6_5

Complete!

Note

Accept the warning about trusting the Hortonworks GPG Key. That key will be automatically downloaded and used to validate packages from Hortonworks. You will see the following message:

```
Importing GPG key 0x07513CAD: Userid: "Jenkins (HDP
Builds) <jenkin@hortonworks.com>" From : http://
s3.amazonaws.com/dev.hortonworks.com/ambari/centos6/RPM-
GPG-KEY/RPM-GPG-KEY-Jenkins
```

SLES 11

On a server host that has Internet access, use a command line editor to perform the following steps:

1. Log in to your host as `root`.
2. Download the Ambari repository file to a directory on your installation host. `wget -nv http://public-repo-1.hortonworks.com/ambari/suse11/2.x/updates/2.0.1/ambari.repo -O /etc/zypp/repos.d/ambari.repo`

Important

Do not modify the `ambari.repo` file name. This file is expected to be available on the Ambari Server host during Agent registration.

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

   `zypper repos` You should see the Ambari repositories in the list.

   Version values vary, depending on the installation.

Alias	Name	En
AMBARI.2.0.1-2.x	Ambari 2.x	Ye
http-demeter.uni-regensburg.de-c997c8f9	SUSE-Linux-Enterprise-Software-Development-Kit-11-SP1 11.1.1-1.57	Ye
opensuse	OpenSuse	Ye

4. Install the Ambari bits. This also installs PostgreSQL.

   ```
   zypper install ambari-server
   ```

5. Enter y when prompted to to confirm transaction and dependency checks.

   A successful installation displays output similar to the following:

   Retrieving package postgresql-libs-8.3.5-1.12.x86_64 (1/4), 172.0 KiB (571.0 KiB unpacked)

   Retrieving: postgresql-libs-8.3.5-1.12.x86_64.rpm [done (47.3 KiB/s)]

   Installing: postgresql-libs-8.3.5-1.12 [done]

   Retrieving package postgresql-8.3.5-1.12.x86_64 (2/4), 1.0 MiB (4.2 MiB unpacked)

   Retrieving: postgresql-8.3.5-1.12.x86_64.rpm [done (148.8 KiB/s)]

   Installing: postgresql-8.3.5-1.12 [done]

   Retrieving package postgresql-server-8.3.5-1.12.x86_64 (3/4), 3.0 MiB (12.6 MiB unpacked)

   Retrieving: postgresql-server-8.3.5-1.12.x86_64.rpm [done (452.5 KiB/s)]

   Installing: postgresql-server-8.3.5-1.12 [done]

   Updating etc/sysconfig/postgresql...

   Retrieving package ambari-server-2.0.1-59.noarch (4/4), 99.0 MiB (126.3 MiB unpacked)

   Retrieving: ambari-server-2.0.1-59.noarch.rpm [done (3.0 MiB/s)]

   Installing: ambari-server-2.0.1-59 [done]

   ambari-server 0:off 1:off 2:off 3:on 4:off 5:on 6:off

UBUNTU 12

On a server host that has Internet access, use a command line editor to perform the following steps:

1. Log in to your host as `root`.

2. Download the Ambari repository file to a directory on your installation host. `wget -nv http://public-repo-1.hortonworks.com/ambari/ubuntu12/2.x/updates/2.0.1/ambari.list -O /etc/apt/sources.list.d/ambari.list apt-key adv --recv-keys --keyserver keyserver.ubuntu.com B9733A7A07513CAD apt-get update`

Important

Do not modify the `ambari.list` file name. This file is expected to be available on the Ambari Server host during Agent registration.

3. Confirm that Ambari packages downloaded successfully by checking the package name list.

   `apt-cache pkgnames`

   You should see the Ambari packages in the list.

   Version values vary, depending on the installation.

Alias	Name
AMBARI-dev-2.x	Ambari 2.x

4. Install the Ambari bits. This also installs PostgreSQL.

   `apt-get install ambari-server`

RHEL/CentOS/ORACLE Linux 5 (DEPRECATED)

On a server host that has Internet access, use a command line editor to perform the following steps:

1. Log in to your host as `root`.

2. Download the Ambari repository file to a directory on your installation host.

   `wget -nv http://public-repo-1.hortonworks.com/ambari/centos5/2.x/updates/2.0.1/ambari.repo -O /etc/yum.repos.d/ambari.repo`

Important

Do not modify the `ambari.repo` file name. This file is expected to be available on the Ambari Server host during Agent registration.

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

   `yum repolist` You should see listed values similar to the following:

repo Id	repo Name	status
AMBARI.2.0.1-2.x	Ambari 2.x	5
base	CentOS-5 - Base	3,667
epel	Extra Packages for Enterprise Linux 5 - x86_64	7,614

repo Id	repo Name	status
puppet	Puppet	433
updates	CentOS-5 - Updates	118

4. Install the Ambari bits. This also installs PostgreSQL.

```
yum install ambari-server
```

Note

When deploying HDP on a cluster having limited or no Internet access, you should provide access to the bits using an alternative method.

- For more information about setting up local repositories, see Using a Local Repository.
- For more information about obtaining JCE policy archives for secure authentication, see Installing the JCE.

Ambari Server by default uses an embedded PostgreSQL database. When you install the Ambari Server, the PostgreSQL packages and dependencies must be available for install. These packages are typically available as part of your Operating System repositories. Please confirm you have the appropriate repositories available for the postgresql-server packages.

1.2. Install the Ambari Agents Manually

Use the instructions specific to the OS family running on your agent hosts.

RHEL/CentOS/Oracle Linux 6

1. Install the Ambari Agent on every host in your cluster.

```
yum install ambari-agent
```

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

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

[server] hostname=<your.ambari.server.hostname>

url_port=8440

secured_url_port=8441
```

3. Start the agent on every host in your cluster.

```
ambari-agent start
```

The agent registers with the Server on start.

SLES 11

1. Install the Ambari Agent on every host in your cluster.

   ```
   zypper install ambari-agent
   ```

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

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

   [server] hostname=<your.ambari.server.hostname>

   url_port=8440

   secured_url_port=8441
   ```

3. Start the agent on every host in your cluster.

   ```
   ambari-agent start
   ```

   The agent registers with the Server on start.

UBUNTU 12

1. Install the Ambari Agent on every host in your cluster.

   ```
   apt-get install ambari-agent
   ```

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

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

   [server] hostname=<your.ambari.server.hostname>

   url_port=8440

   secured_url_port=8441
   ```

3. Start the agent on every host in your cluster.

   ```
   ambari-agent start
   ```

   The agent registers with the Server on start.

RHEL/CentOS/Oracle Linux 5 (DEPRECATED)

1. Install the Ambari Agent on every host in your cluster.

   ```
   yum install ambari-agent
   ```

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

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

   [server] hostname=<your.ambari.server.hostname>
   ```

```
url_port=8440

secured_url_port=8441
```

3. Start the agent on every host in your cluster.

   ```
   ambari-agent start
   ```

   The agent registers with the Server on start.

2. Configuring Ambari for Non-Root

In most secure environments, restricting access to and limiting services that run as root is a hard requirement. For these environments, Ambari can be configured to operate without direct root access. Both Ambari Server and Ambari Agent components allow for non-root operation, and the following sections will walk you through the process.

- How to Configure Ambari Server for Non-Root
- How to Configure an Ambari Agent for Non-Root

2.1. How to Configure Ambari Server for Non-Root

You can configure the Ambari Server to run as a non-root user.

During the ambari-server setup process, when prompted to `Customize user account for ambari-server daemon?`, choose `y`.

The setup process prompts you for the appropriate, non-root user to run the Ambari Server as; for example: ambari.

Note

When running the Ambari Server as a non-root user, confirm that the /etc/login.defs file is readable by that user. The Ambari Server reads this file during startup to determine the minimum UID. A permission of 644 for /etc/login.defs is recommended.

2.2. How to Configure an Ambari Agent for Non-Root

You can configure the Ambari Agent to run as a non-privileged user as well. That user requires specific sudo access in order to su to Hadoop service accounts and perform specific privileged commands. Configuring Ambari Agents to run as non-root requires that you manually install agents on all nodes in the cluster. For these details, see Installing Ambari Agents Manually. After installing each agent, you must configure the agent to run as the desired, non-root user. In this example we will use the `ambari` user.

Change the `run_as_user` property in the `/etc/ambari-agent/conf/ambari-agent.ini` file, as illustrated below:

```
run_as_user=ambari
```

Once this change has been made, the ambari-agent must be restarted to begin running as the non-root user.

The non-root functionality relies on sudo to run specific commands that require elevated privileges as defined in the Sudoer Configuration. The sudo configuration is split into the

following sections: Customizable Users, Non-Customizable Users, Commands, and Sudo Defaults.

2.2.1. Sudoer Configuration

The Customizable Users, Non-Customizable Users, Commands, and Sudo Defaults sections will cover how sudo should be configured to enable Ambari to run as a non-root user. Each of the sections includes the specific sudo entries that should be placed in `/etc/sudoers` by running the `visudo` command.

2.2.2. Customizable Users

This section contains the `su` commands and corresponding Hadoop service accounts that are configurable on install:

```
# Ambari Customizable Users
ambari ALL=(ALL) NOPASSWD:SETENV: /bin/su hdfs *, /bin/su zookeeper *, /bin/su
 knox *,/bin/su falcon *,/bin/su flume *,/bin/su hbase *,/bin/su hive *, /bin/
su hcat *,/bin/su kafka *,/bin/su mapred *,/bin/su oozie *,/bin/su sqoop *,/
bin/su storm *,/bin/su tez *,/bin/su yarn *,/bin/su ams *, /bin/su ambari-qa
 *, /bin/su spark *, /bin/su ranger *
```

Note

These user accounts must match the service user accounts referenced in the `Customize Services > Misc` tab during the Install Wizard configuration step. For example, if you customize YARN to run as xyz_yarn, modify the `su` command above to be `/bin/su xyz_yarn`.

2.2.3. Non-Customizable Users

This section contains the `su` commands for the system accounts that cannot be modified:

```
# Ambari Non-Customizable Users
ambari ALL=(ALL) NOPASSWD:SETENV: /bin/su mysql *
```

2.2.4. Commands

This section contains the specific commands that must be issued for standard agent operations:

```
# Ambari Commands
ambari ALL=(ALL) NOPASSWD:SETENV: /usr/bin/yum,/usr/bin/zypper,/usr/bin/
apt-get, /bin/mkdir, /bin/ln,/bin/chown, /bin/chmod, /bin/chgrp, /usr/sbin/
groupadd, /usr/sbin/groupmod,/usr/sbin/useradd, /usr/sbin/usermod, /bin/cp,
 /bin/sed, /bin/mv, /bin/rm, /bin/kill,/usr/bin/unzip, /bin/tar, /usr/bin/
hdp-select, /usr/hdp/current/hadoop-client/sbin/hadoop-daemon.sh,/usr/lib/
hadoop/bin/hadoop-daemon.sh, /usr/lib/hadoop/sbin/hadoop-daemon.sh, /usr/sbin/
service mysql *,/sbin/service mysqld *, /sbin/service mysql *, /sbin/chkconfig
 gmond off,/sbin/chkconfig gmetad off, /etc/init.d/httpd *, /sbin/service hdp-
gmetad start, /sbin/service hdp-gmond start, /usr/bin/tee, /usr/sbin/gmond, /
usr/sbin/update-rc.d ganglia-monitor *, /usr/sbin/update-rc.d gmetad *, /etc/
init.d/apache2 *, /usr/sbin/service hdp-gmond *, /usr/sbin/service hdpgmetad
 *, /usr/bin/test, /bin/touch, /usr/bin/stat, /usr/sbin/setenforce, /usr/hdp/
```

```
current/ranger-admin/setup.sh, /usr/hdp/current/ranger-usersync/setup.sh, /
usr/bin/ranger-usersync-start, /usr/bin/ranger-usersync-stop
```

Important

Do not modify the command lists, only the usernames in the Customizable Users section may be modified.

To re-iterate, you must do this sudo configuration on every node in the cluster. To ensure that the configuration has been done properly, you can su to the ambari user and run sudo -l. There, you can double check that there are no warnings, and that the configuration output matches what was just applied.

2.2.5. Sudo Defaults

Some versions of sudo have a default configuration that prevents sudo from being invoked from a non-interactive shell. In order for the agent to run it's commands non-interactively, some defaults need to be overridden.

```
Defaults exempt_group = ambari
Defaults !env_reset,env_delete-=PATH
Defaults: ambari !requiretty
```

To re-iterate, this sudo configuration must be done on every node in the cluster. To ensure that the configuration has been done properly, you can su to the ambari user and run sudo -l. There, you can double-check that there are no warnings, and that the configuration output matches what was just applied.

3. Customizing HDP Services

The individual services in Hadoop run under the ownership of their respective Unix accounts. These accounts are known as service users. These service users belong to a special Unix group. "Smoke Test" is a service user dedicated specifically for running smoke tests on components during installation using the `Services` View of the Ambari Web GUI. You can also run service checks as the "Smoke Test" user on-demand after installation.

- Defining Service Users and Groups for a HDP 2.x Stack
- Setting Properties That Depend on Service Usernames/Groups

3.1. Defining Service Users and Groups for a HDP 2.x Stack

You can customize any of these users and groups using the `Misc` tab during the `Customize Services` installation step.

Note

Use the `Skip Group Modifications` option to not modify the Linux groups in the cluster. Choosing this option is typically required if your environment manages groups using LDAP and not on the local Linux machines.

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.

Service Users

Service*	Component	Default User Account
Ambari Metrics	Metrics Collector, Metrics Monitor	ams
Falcon	Falcon Server	falcon (Falcon is available with HDP 2.1 or 2.2 Stack.)
Flume	Flume Agents	flume
HBase	MasterServer RegionServer	hbase
HDFS	NameNode SecondaryNameNode DataNode	hdfs
Hive	Hive Metastore, HiveServer2	hive
Kafka	Kafka Broker	kafka
Knox	Knox Gateway	knox
MapReduce2	HistoryServer	mapred
Oozie	Oozie Server	oozie

Service*	Component	Default User Account
PostgreSQL	PostgreSQL (with Ambari Server)	postgres (Created as part of installing the default PostgreSQL database with Ambari Server. If you are not using the Ambari PostgreSQL database, this user is not needed.)
Ranger	Ranger Admin, Ranger Usersync	ranger (Ranger is available with HDP 2.2 Stack)
Spark	Spark History Server	spark (Spark is available with HDP 2.2 Stack)
Sqoop	Sqoop	sqoop
Storm	Masters (Nimbus, DRPC Server, Storm REST API, Server, Storm UI Server) Slaves (Supervisors, Logviewers)	storm (Storm is available with HDP 2.1 or 2.2 Stack.)
Tez	Tez clients	tez (Tez is available with HDP 2.1 or 2.2 Stack.)
WebHCat	WebHCat Server	hcat
YARN	NodeManager ResourceManager	yarn
ZooKeeper	ZooKeeper	zookeeper

*For all components, the 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 UI. The default user account for the smoke test user is ambari-qa.

Service Groups

Service	Components	Default Group Account
All	All	hadoop
Knox	Knox Gateway	knox
Ranger	Ranger Admin, Ranger Usersync	ranger
Spark	Spark History Server	spark

3.2. Setting Properties That Depend on Service Usernames/Groups

Some properties must be set to match specific service user names or service groups. If you have set up non-default, customized service user names for the HDFS or HBase service or the Hadoop group name, you must edit the following properties, using `Services > Service.Name > Configs > Advanced`:

HDFS Settings: Advanced

Property Name	Value
dfs.permissions.superusergroup	The same as the HDFS username. The default is "hdfs"
dfs.cluster.administrators	A single space followed by the HDFS username.
dfs.block.local-path-access.user	The HBase username. The default is "hbase".

MapReduce Settings: Advanced

Property Name	Value
mapreduce.cluster.administrators	A single space followed by the Hadoop group name.

4. Configuring Storm for Supervision

If you have installed a cluster with HDP 2.2 Stack that includes the Storm service, you can configure the Storm components to operate under supervision. This section describes those steps:

1. Stop all Storm components.

   Using Ambari Web, browse to `Services > Storm > Service Actions`, choose Stop. Wait until the Storm service stop completes.

2. Stop Ambari Server.

   ```
   ambari-server stop
   ```

3. Change Supervisor and Nimbus command scripts in the Stack definition.

   On Ambari Server host, run:

   ```
   sed -ir "s/scripts\/supervisor.py/scripts\/supervisor_prod.py/g" /var/lib/
   ambari-server/resources/common-services/STORM/0.9.1.2.1/metainfo.xml

   sed -ir "s/scripts\/nimbus.py/scripts\/nimbus_prod.py/g" /var/lib/ambari-
   server/resources/common-services/STORM/0.9.1.2.1/metainfo.xml
   ```

4. Install supervisord on all Nimbus and Supervisor hosts.

   - Install EPEL repository.

     ```
     yum install epel-release -y
     ```

   - Install supervisor package for supervisord.

     ```
     yum install supervisor -y
     ```

   - Enable supervisord on autostart.

     ```
     chkconfig supervisord on
     ```

   - Change supervisord configuration file permissions.

     ```
     chmod 600 /etc/supervisord.conf
     ```

5. Configure `supervisord` to supervise Nimbus Server and Supervisors by appending the following to `/etc/supervisord.conf` on all Supervisor host and Nimbus hosts accordingly.

   ```
   [program:storm-nimbus]
   command=env PATH=$PATH:/bin:/usr/bin/:/usr/jdk64/jdk1.7.0_67/bin/ JAVA_HOME=
   /usr/jdk64/jdk1.7.0_67 /usr/hdp/current/storm-nimbus/bin/storm nimbus
   user=storm
   autostart=true
   autorestart=true
   startsecs=10
   startretries=999
   log_stdout=true
   ```

```
log_stderr=true
logfile=/var/log/storm/nimbus.out
logfile_maxbytes=20MB
logfile_backups=10

[program:storm-supervisor]
command=env PATH=$PATH:/bin:/usr/bin/:/usr/jdk64/jdk1.7.0_67/bin/ JAVA_HOME=
/usr/jdk64/jdk1.7.0_67 /usr/hdp/current/storm-supervisor/bin/storm
 supervisor
user=storm
autostart=true
autorestart=true
startsecs=10
startretries=999
log_stdout=true
log_stderr=true
logfile=/var/log/storm/supervisor.out
logfile_maxbytes=20MB
logfile_backups=10
```

Note

Change `/usr/jdk64/jdk1.7.0_67` accordingly to the location of the JDK being used by Ambari in your environment.

6. Start Supervisord service on all Supervisor and Nimbus hosts.

   ```
   service supervisord start
   ```

7. Start Ambari Server.

   ```
   ambari-server start
   ```

8. Start all the other Storm components.

   Using Ambari Web, browse to `Services > Storm > Service Actions`, choose `Start`.

5. Using Custom Host Names

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

How to Customize the name of a host

5.1. How to Customize the name of a host

1. At the `Install Options` step in the Cluster Installer wizard, select `Perform Manual Registration for Ambari Agents`.
2. Install the Ambari Agents manually on each host, as described in Install the Ambari Agents Manually.
3. To echo the customized name of the host to which the Ambari agent registers, for every host, create a script like the following example, named `/var/lib/ambari-agent/hostname.sh`. Be sure to `chmod` the script so it is executable by the Agent.
```
#!/bin/sh
echo <ambari_hostname>
```
   where <ambari_hostname> is the host name to use for Agent registration.
4. Open `/etc/ambari-agent/conf/ambari-agent.ini` on every host, using a text editor.
5. Add to the `[agent]` section the following line:
```
hostname_script=/var/lib/ambari-agent/hostname.sh
```
   where `/var/lib/ambari-agent/hostname.sh` is the name of your custom echo script.
6. To generate a public host name for every host, create a script like the following example, named `var/lib/ambari-agent/public_hostname.sh` to show the name for that host in the UI. Be sure to `chmod` the script so it is executable by the Agent.
```
#!/bin/sh
<hostname> -f
```
   where <hostname> is the host name to use for Agent registration.
7. Open `/etc/ambari-agent/conf/ambari-agent.ini` on every host, using a text editor.
8. Add to the `[agent]` section the following line:
```
public_hostname_script=/var/lib/ambari-agent/public_hostname.sh
```
9. If applicable, add the host names to `/etc/hosts` on every host.
10. Restart the Agent on every host for these changes to take effect.
```
ambari-agent restart
```

6. Moving the Ambari Server

To transfer an Ambari Server that uses the default, PostgreSQL database to a new host, use the following instructions:

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 New Server - on a new host and populate databases with information from original Server.

Note

If your Ambari database is one of the non-default types, such as Oracle, adjust the database backup, restore, and stop/start procedures to match that database type.

6.1. Back up Current Data

1. Stop the original Ambari Server.

   ```
   ambari-server stop
   ```

2. Create a directory to hold the database backups.

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

3. Create the database backups.

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

   where <AMBARI.SERVER.USERNAME>, <MAPRED.USERNAME>, <AMBARI.SERVER.PASSWORD>, and <MAPRED.PASSWORD> are the user names and passwords that you set up during installation. Default values are: `ambari-server/bigdata` and `mapred/mapred`.

6.2. Update Agents

1. On each agent host, stop the agent.

   ```
   ambari-agent stop
   ```

2. Remove old agent certificates.

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

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

[server] hostname= <NEW FULLY.QUALIFIED.DOMAIN.NAME>

url_port=8440

secured_url_port=8441

6.3. Install the New Server and Populate the Databases

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

   ```
   ambari-server stop
   ```

3. Restart the PostgreSQL instance.

   ```
   service postgresql restart
   ```

4. Open the PostgreSQL interactive terminal.

   ```
   su - postgres psql
   ```

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

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

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

   ```
   /list
   ```

   The databases should not be listed.

7. Create new databases to hold the transferred data.

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

8. Exit the interactive terminal.

   ```
   ^d
   ```

9. Copy the saved data from Back up Current Data to the new Server.

   ```
   cd /tmp scp -i <ssh-key> root@<original.Ambari.Server>/tmp/dbdumps/
   *.sql/tmp
   ```

   Note

   `compress/transfer/uncompress` as needed from source to dest

   ```
   psql -d ambari -f /tmp/ambari.sql psql -d ambarirca -f /tmp/
   ambarirca.sql
   ```

10. Start the new Server.

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

11. On each Agent host, start the Agent.

```
ambari-agent start
```

12. Open Ambari Web. Point your 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.

7. Configuring LZO Compression

LZO is a lossless data compression library that favors speed over compression ratio. Ambari does not install nor enable LZO Compression by default. To enable LZO compression in your HDP cluster, you must Configure core-site.xml for LZO.

Optionally, you can implement LZO to optimize Hive queries in your cluster for speed. For more information about using LZO compression with Hive, see Running Compression with Hive Queries.

7.1. Configure core-site.xml for LZO

1. Browse to `Ambari Web > Services > HDFS > Configs`, then expand `Advanced core-site`.
2. Find the `io.compression.codecs` property key.
3. Append to the `io.compression.codecs` property key, the following value: `com.hadoop.compression.lzo.LzoCodec`
4. Add a description of the config modification, then choose Save.
5. Expand the `Custom core-site.xml` section.
6. Select `Add Property`.
7. Add to `Custom core-site.xml` the following property key and value

Property Key	Property Value
io.compression.codec.lzo.class	com.hadoop.compression.lzo.LzoCodec

8. Choose Save.
9. Add a description of the config modification, then choose Save.
10. Restart the HDFS, MapReduce2 and YARN services.

Note

If performing a Restart or a Restart All does not start the required package install, you may need to stop, then start the HDFS service to install the necessary LZO packages. Restart is only available for a service in the "Runnning" or "Started" state.

7.2. Running Compression with Hive Queries

Running Compression with Hive Queries requires creating LZO files. To create LZO files, use one of the following procedures:

7.2.1. Create LZO Files

1. Create LZO files as the output of the Hive query.

2. Use `lzo` command utility or your custom Java to generate `lzo.index` for the `.lzo` files.

Hive Query Parameters

Prefix the query string with these parameters:

```
SET mapreduce.output.fileoutputformat.compress.codec=com.hadoop.compression.
lzo.LzoCodec
SET hive.exec.compress.output=true
SET mapreduce.output.fileoutputformat.compress=true
```

For example:

```
hive -e "SET
mapreduce.output.fileoutputformat.compress.codec=com.hadoop.compression.lzo.Lz
hive.exec.compress.output=true;SET
mapreduce.output.fileoutputformat.compress=true;"
```

7.2.2. Write Custom Java to Create LZO Files

1. Create text files as the output of the Hive query.
2. Write custom Java code to
   - convert Hive query generated text files to `.lzo` files
   - generate `lzo.index` files for the `.lzo` files

Hive Query Parameters

Prefix the query string with these parameters:

```
SET hive.exec.compress.output=false
SET mapreduce.output.fileoutputformat.compress=false
```

For example:

```
hive -e "SET hive.exec.compress.output=false;SET
mapreduce.output.fileoutputformat.compress=false;<query-string>"
```

8. Using Non-Default Databases

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

- Using Non-Default Databases - Ambari
- Using Non-Default Databases - Hive
- Using Non-Default Databases - Oozie

8.1. Using Non-Default Databases - Ambari

The following sections describe how to use Ambari with an existing database, other than the embedded PostgreSQL database instance that Ambari Server uses by default.

- Using Ambari with Oracle
- Using Ambari with MySQL
- Using Ambari with PostgreSQL
- Troubleshooting Non-Default Databases with Ambari

8.1.1. Using Ambari with Oracle

To set up Oracle for use with Ambari:

1. On the Ambari Server host, install the appropriate `JDBC.jar` file.
   a. Download the Oracle JDBC (OJDBC) driver from http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html.
   b. Select `Oracle Database 11g Release 2 - ojdbc6.jar`.
   c. Copy the .jar file to the Java share directory.

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

   d. Make sure the .jar file has the appropriate permissions - 644.
2. Create a user for Ambari and grant that user appropriate permissions.

   For example, using the Oracle database admin utility, run the following commands:

   ```
   # sqlplus sys/root as sysdba
   ```

   CREATE USER <AMBARIUSER> IDENTIFIED BY <AMBARIPASSWORD> default tablespace "USERS" temporary tablespace "TEMP";

   ```
   GRANT unlimited tablespace to <AMBARIUSER>;

   GRANT create session to <AMBARIUSER>;
   ```

```
GRANT create TABLE to <AMBARIUSER>;

GRANT create SEQUENCE to <AMBARIUSER>;

QUIT;
```

Where <AMBARIUSER> is the Ambari user name and <AMBARIPASSWORD> is the Ambari user password.

3. Load the Ambari Server database schema.

   a. You must pre-load the Ambari database schema into your Oracle database using the schema script.

   ```
   sqlplus <AMBARIUSER>/<AMBARIPASSWORD> < Ambari-DDL-Oracle-CREATE.sql
   ```

   b. Find the Ambari-DDL-Oracle-CREATE.sql file in the `/var/lib/ambari-server/resources/` directory of the Ambari Server host after you have installed Ambari Server.

4. When setting up the Ambari Server, select `Advanced Database Configuration > Option [2] Oracle` and respond to the prompts using the username/password credentials you created in step 2.

8.1.2. Using Ambari with MySQL

To set up MySQL for use with Ambari:

1. On the Ambari Server host, install the connector.

   a. Install the connector

   RHEL/CentOS/Oracle Linux

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

   SLES

   ```
   zypper install mysql-connector-java
   ```

   >**Ubuntu**

   ```
   apt-get install mysql-connector-java
   ```

   b. Confirm that `.jar` 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. Create a user for Ambari and grant it permissions.

   - For example, using the MySQL database admin utility:

```
# mysql -u root -p

CREATE USER '<AMBARIUSER>'@'%' IDENTIFIED BY
'<AMBARIPASSWORD>';

GRANT ALL PRIVILEGES ON *.* TO '<AMBARIUSER>'@'%';

CREATE USER '<AMBARIUSER>'@'localhost' IDENTIFIED BY
'<AMBARIPASSWORD>';

GRANT ALL PRIVILEGES ON *.* TO '<AMBARIUSER>'@'localhost';

CREATE USER '<AMBARIUSER>'@'<AMBARISERVERFQDN>' IDENTIFIED BY
'<AMBARIPASSWORD>';

GRANT ALL PRIVILEGES ON *.* TO
'<AMBARIUSER>'@'<AMBARISERVERFQDN>';

FLUSH PRIVILEGES;
```

- Where <AMBARIUSER> is the Ambari user name, <AMBARIPASSWORD> is the Ambari user password and <AMBARISERVERFQDN> is the Fully Qualified Domain Name of the Ambari Server host.

3. Load the Ambari Server database schema.

   - You must pre-load the Ambari database schema into your MySQL database using the schema script.

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

   - Where <AMBARIUSER> is the Ambari user name and <AMBARIDATABASE> is the Ambari database name.

     Find the `Ambari-DDL-MySQL-CREATE.sql` file in the `/var/lib/ambari-server/resources/` directory of the Ambari Server host after you have installed Ambari Server.

4. When setting up the Ambari Server, select `Advanced Database Configuration > Option [3] MySQL` and enter the credentials you defined in Step 2. for user name, password and database name.

8.1.3. Using Ambari with PostgreSQL

To set up PostgreSQL for use with Ambari:

1. Create a user for Ambari and grant it permissions.

   - Using the PostgreSQL database admin utility:

     ```
     # sudo -u postgres psql

     CREATE DATABASE <AMBARIDATABASE>;
     ```

```
CREATE USER <AMBARIUSER> WITH PASSWORD '<AMBARIPASSWORD>';

GRANT ALL PRIVILEGES ON DATABASE <AMBARIDATABASE> TO
<AMBARIUSER>;

\connect <AMBARIDATABASE>;

CREATE SCHEMA <AMBARISCHEMA> AUTHORIZATION <AMBARIUSER>;

ALTER SCHEMA <AMBARISCHEMA> OWNER TO <AMBARIUSER>;

ALTER ROLE <AMBARIUSER> SET search_path to '<AMBARISCHEMA>',
'public';
```

- Where <AMBARIUSER> is the Ambari user name <AMBARIPASSWORD> is the Ambari user password, <AMBARIDATABASE> is the Ambari database name and <AMBARISCHEMA>is the Ambari schema name.

2. Load the Ambari Server database schema.

   - You must pre-load the Ambari database schema into your PostgreSQL database using the schema script.

     ```
     # psql -U <AMBARIUSER> -d <AMBARIDATABASE> \connect
     <AMBARIDATABASE>; \i Ambari-DDL-Postgres-CREATE.sql;
     ```

   - Find the `Ambari-DDL-Postgres-CREATE.sql` file in the `/var/lib/ambari-server/resources/` directory of the Ambari Server host after you have installed Ambari Server.

3. When setting up the Ambari Server, select `Advanced Database Configuration > Option[4] PostgreSQL` and enter the credentials you defined in Step 2. for user name, password, and database name.

8.1.4. Troubleshooting Ambari

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

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

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

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

The Oracle JDBC.jar file cannot be found.

8.1.4.1.1. Solution

Make sure the file is in the appropriate directory on the Ambari server and re-run `ambari-server setup`. Review the load database procedure appropriate for your database type in Using Non-Default Databases - Ambari.

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

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

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

Ambari Server cannot connect to the database.

8.1.4.2.1. Solution

Confirm that 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/
ambari
```

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

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

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

You are using an invalid username/password.

8.1.4.3.1. Solution

Confirm the user account is set up in the database and has the correct privileges. See Step 3 above.

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

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

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

The schema has not been loaded.

8.1.4.4.1. Solution

Confirm you have loaded the database schema. Review the load database schema procedure appropriate for your database type in Using Non-Default Databases - Ambari.

8.2. Using Non-Default Databases - Hive

The following sections describe how to use Hive with an existing database, other than the MySQL database instance that Ambari installs by default.

- Using Hive with Oracle
- Using Hive with MySQL
- Using Hive with PostgreSQL

- Troubleshooting Non-Default Databases with Hive

8.2.1. Using Hive with Oracle

To set up Oracle for use with Hive:

1. On the Ambari Server host, stage the appropriate JDBC driver file for later deployment.

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

   b. Select `Oracle Database 11g Release 2 - ojdbc6.jar` and download the file.

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

   d. Execute the following command, adding the path to the downloaded .jar file:

   ```
   ambari-server setup --jdbc-db=oracle --jdbc-driver=/path/to/
   downloaded/ojdbc6.jar
   ```

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

   - Using the Oracle database admin utility:

   ```
   # sqlplus sys/root as sysdba CREATE USER <HIVEUSER> IDENTIFIED
   BY <HIVEPASSWORD>;

   GRANT SELECT_CATALOG_ROLE TO <HIVEUSER>;

   GRANT CONNECT, RESOURCE TO <HIVEUSER>;

   QUIT;
   ```

   - Where <HIVEUSER> is the Hive user name and <HIVEPASSWORD> is the Hive user password.

3. Load the Hive database schema.

   - For a **HDP 2.2 Stack**

   Important

   Ambari sets up the Hive Metastore database schema automatically.

   You do not need to pre-load the Hive Metastore database schema into your Oracle database for a HDP 2.2 Stack.

   - For a **HDP 2.1 Stack**

   You must pre-load the Hive database schema into your Oracle database using the schema script, as follows: sqlplus <HIVEUSER>/<HIVEPASSWORD> < hive-schema-0.13.0.oracle.sql

Find the `hive-schema-0.13.0.oracle.sql` file in the `/var/lib/ambari-server/resources/stacks/HDP/2.1/services/HIVE/etc/` directory of the Ambari Server host after you have installed Ambari Server.

- For a **HDP 2.0 Stack**

  You must pre-load the Hive database schema into your Oracle database using the schema script, as follows: sqlplus <HIVEUSER>/<HIVEPASSWORD> < hive-schema-0.12.0.oracle.sql

  Find the `hive-schema-0.12.0.oracle.sql` file in the `/var/lib/ambari-server/resources/stacks/HDP/2.0.6/services/HIVE/etc/` directory of the Ambari Server host after you have installed Ambari Server.

- For a **HDP 1.3 Stack**

  You must pre-load the Hive database schema into your Oracle database using the schema script, as follows: sqlplus <HIVEUSER>/<HIVEPASSWORD> < hive-schema-0.10.0.oracle.sql

  Find the `hive-schema-0.10.0.oracle.sql` file in the `/var/lib/ambari-server/resources/stacks/HDP/1.3.2/services/HIVE/etc/` directory of the Ambari Server host after you have installed Ambari Server.

8.2.2. Using Hive with MySQL

To set up MySQL for use with Hive:

1. On the Ambari Server host, stage the appropriate MySQL connector for later deployment.

   a. Install the connector.

      RHEL/CentOS/Oracle Linux

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

      SLES

      ```
      zypper install mysql-connector-java*
      ```

      Ubuntu

      ```
      apt-get install mysql-connector-java*
      ```

   b. Confirm that `mysql-connector-java.jar` 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.

   d. Execute the following command:

```
ambari-server setup --jdbc-db=mysql --jdbc-driver=/usr/share/
java/mysql-connector-java.jar
```

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

   - Using the MySQL database admin utility:

     ```
     # mysql -u root -p CREATE USER '<HIVEUSER>'@'localhost'
     IDENTIFIED BY '<HIVEPASSWORD>';

     GRANT ALL PRIVILEGES ON *.* TO '<HIVEUSER>'@'localhost';

     CREATE USER '<HIVEUSER>'@'%' IDENTIFIED BY '<HIVEPASSWORD>';

     GRANT ALL PRIVILEGES ON *.* TO '<HIVEUSER>'@'%';

     CREATE USER '<HIVEUSER>'@'<HIVEMETASTOREFQDN>'IDENTIFIED BY
     '<HIVEPASSWORD>';

     GRANT ALL PRIVILEGES ON *.* TO
     '<HIVEUSER>'@'<HIVEMETASTOREFQDN>';

     FLUSH PRIVILEGES;
     ```

   - Where <HIVEUSER> is the Hive user name, <HIVEPASSWORD> is the Hive user password and <HIVEMETASTOREFQDN> is the Fully Qualified Domain Name of the Hive Metastore host.

3. Create the Hive database.

   The Hive database must be created before loading the Hive database schema.

   ```
   # mysql -u root -p

   CREATE DATABASE <HIVEDATABASE>
   ```

   Where <HIVEDATABASE> is the Hive database name.

4. Load the Hive database schema.

   - For a **HDP 2.2 Stack**:

     Important

     Ambari sets up the Hive Metastore database schema automatically.

     You do not need to pre-load the Hive Metastore database schema into your MySQL database for a HDP 2.2 Stack.

   - For a **HDP 2.1 Stack**:

     You must pre-load the Hive database schema into your MySQL database using the schema script, as follows. `mysql -u root -p <HIVEDATABASE> hive-schema-0.13.0.mysql.sql`

Find the `hive-schema-0.13.0.mysql.sql` file in the `/var/lib/ambari-server/resources/stacks/HDP/2.1/services/HIVE/etc/` directory of the Ambari Server host after you have installed Ambari Server.

8.2.3. Using Hive with PostgreSQL

To set up PostgreSQL for use with Hive:

1. On the Ambari Server host, stage the appropriate PostgreSQL connector for later deployment.

   a. Install the connector.

   RHEL/CentOS/Oracle Linux

   ```
   yum install postgresql-jdbc*
   ```

   SLES

   ```
   zypper install -y postgresql-jdbc
   ```

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

   ```
   cp /usr/share/pgsql/postgresql-*.jdbc3.jar /usr/share/java/
   postgresql-jdbc.jar
   ```

   c. Confirm that .jar is in the Java share directory.

   ```
   ls /usr/share/java/postgresql-jdbc.jar
   ```

   d. Change the access mode of the.jar file to 644.

   chmod 644 /usr/share/java/postgresql-jdbc.jar

   e. Execute the following command:

   ```
   ambari-server setup --jdbc-db=postgres--jdbc-driver=/usr/
   share/java/postgresql-jdbc.jar
   ```

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

   - Using the PostgreSQL database admin utility:

   ```
   echo "CREATE DATABASE <HIVEDATABASE>;" | psql -U postgres echo
   "CREATE USER <HIVEUSER> WITH PASSWORD '<HIVEPASSWORD>';" | psql
   -U postgres

   echo "GRANT ALL PRIVILEGES ON DATABASE <HIVEDATABASE> TO
   <HIVEUSER>;" | psql -U postgres
   ```

   - Where <HIVEUSER> is the Hive user name, <HIVEPASSWORD> is the Hive user password and <HIVEDATABASE> is the Hive database name.

3. Load the Hive database schema.

- For a **HDP 2.2 Stack:**

  Important

  Ambari sets up the Hive Metastore database schema automatically.

  You do not need to pre-load the Hive Metastore database schema into your PostgreSQL database for a HDP 2.2 Stack.

- For a **HDP 2.1 Stack:**

  You must pre-load the Hive database schema into your PostgreSQL database using the schema script, as follows:

  ```
  # psql -U <HIVEUSER> -d <HIVEDATABASE> \connect <HIVEDATABASE>;
  \i hive-schema-0.13.0.postgres.sql;
  ```

  Find the `hive-schema-0.13.0.postgres.sql` file in the `/var/lib/ambari-server/resources/stacks/HDP/2.1/services/HIVE/etc/` directory of the Ambari Server host after you have installed Ambari Server.

- For a **HDP 2.0 Stack:**

  You must pre-load the Hive database schema into your PostgreSQL database using the schema script, as follows:

  ```
  # sudo -u postgres psql \connect <HIVEDATABASE>; \i hive-
  schema-0.12.0.postgres.sql;
  ```

  Find the `hive-schema-0.12.0.postgres.sql` file in the `/var/lib/ambari-server/resources/stacks/HDP/2.0.6/services/HIVE/etc/` directory of the Ambari Server host after you have installed Ambari Server.

- For a **HDP 1.3 Stack:**

  You must pre-load the Hive database schema into your PostgreSQL database using the schema script, as follows:

  ```
  # sudo -u postgres psql \connect <HIVEDATABASE>; \i hive-
  schema-0.10.0.postgres.sql;
  ```

  Find the `hive-schema-0.10.0.postgres.sql` file in the `/var/lib/ambari-server/resources/stacks/HDP/1.3.2/services/HIVE/etc/` directory of the Ambari Server host after you have installed Ambari Server.

8.2.4. Troubleshooting Hive

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

8.2.4.1. 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.

8.2.4.1.1. Solution

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

8.2.4.2. 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.

8.2.4.2.1. Solution

Ignore the warning, and complete the install. Check your database to confirm the Hive Metastore schema is loaded. In the Ambari Web GUI, browse to **Services** > **Hive**. Choose `Service Actions > Service Check` to check that the schema is correctly in place.

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

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

8.2.4.3.1. Solution

Two workarounds are possible. Using Ambari Web, in **HiveConfigsAdvanced**:

- Disable `hive.security.authorization`, by setting the `hive.security.authorization.enabled` value to false.

  or

- Properly configure Hive security authorization. For example, set the following properties:

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

  Hive Security Authorization Settings

Property	Value
hive.security.authorization.manager	org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvid
hive.security.metastore.authorization.manager	org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvid
hive.security.authenticator.manager	org.apache.hadoop.hive.ql.security.ProxyUserAuthenticator

8.3. Using Non-Default Databases - Oozie

The following sections describe how to use Oozie with an existing database, other than the Derby database instance that Ambari installs by default.

- Using Oozie with Oracle
- Using Oozie with MySQL
- Using Oozie with PostgreSQL
- Troubleshooting Non-Default Databases with Oozie

8.3.1. Using Oozie with Oracle

To set up Oracle for use with Oozie:

1. On the Ambari Server host, stage the appropriate JDBC driver file for later deployment.
   a. Download the Oracle JDBC (OJDBC) driver from http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html.
   b. Select Oracle Database 11g Release 2 - ojdbc6.jar.
   c. Make sure the .jar file has the appropriate permissions - 644.
   d. Execute the following command, adding the path to the downloaded.jar file:

      ```
      ambari-server setup --jdbc-db=oracle --jdbc-driver=/path/to/
      downloaded/ojdbc6.jar
      ```

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

   Using the Oracle database admin utility, run the following commands:

   ```
   # sqlplus sys/root as sysdba CREATE USER <OOZIEUSER> IDENTIFIED
   BY <OOZIEPASSWORD>;

   GRANT ALL PRIVILEGES TO <OOZIEUSER>;

   GRANT CONNECT, RESOURCE TO <OOZIEUSER>;

   QUIT;
   ```

   Where <OOZIEUSER> is the Oozie user name and <OOZIEPASSWORD> is the Oozie user password.

8.3.2. Using Oozie with MySQL

To set up MySQL for use with Oozie:

1. On the Ambari Server host, stage the appropriate MySQL connector for later deployment.

a. Install the connector.

RHEL/CentOS/Oracle Linux

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

SLES

```
zypper install mysql-connector-java*
```

UBUNTU

```
apt-get install mysql-connector-java*
```

b. Confirm that `mysql-connector-java.jar` 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.

d. Execute the following command:

```
ambari-server setup --jdbc-db=mysql --jdbc-driver=/usr/share/
java/mysql-connector-java.jar
```

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

- Using the MySQL database admin utility:

```
# mysql -u root -p CREATE USER ‘<OOZIEUSER>’@’%’ IDENTIFIED BY
‘<OOZIEPASSWORD>’;

GRANT ALL PRIVILEGES ON *.* TO '<OOZIEUSER>'@'%';

FLUSH PRIVILEGES;
```

- Where <OOZIEUSER> is the Oozie user name and <OOZIEPASSWORD> is the Oozie user password.

3. Create the Oozie database.

- The Oozie database must be created prior.

```
# mysql -u root -p CREATE DATABASE <OOZIEDATABASE>
```

- Where <OOZIEDATABASE> is the Oozie database name.

8.3.3. Using Oozie with PostgreSQL

To set up PostgreSQL for use with Oozie:

1. On the Ambari Server host, stage the appropriate PostgreSQL connector for later deployment.

a. Install the connector.

RHEL/CentOS/Oracle Linux

```
yum install postgresql-jdbc
```

SLES

```
zypper install -y postgresql-jdbc
```

UBUNTU

```
apt-get install -y postgresql-jdbc
```

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

```
cp /usr/share/pgsql/postgresql-*.jdbc3.jar /usr/share/java/
postgresql-jdbc.jar
```

c. Confirm that .jar is in the Java share directory.

```
ls /usr/share/java/postgresql-jdbc.jar
```

d. Change the access mode of the .jar file to 644.

```
chmod 644 /usr/share/java/postgresql-jdbc.jar
```

e. Execute the following command:

```
ambari-server setup --jdbc-db=postgres --jdbc-driver=/usr/
share/java/postgresql-jdbc.jar
```

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

- Using the PostgreSQL database admin utility:

```
echo "CREATE DATABASE <OOZIEDATABASE>;" | psql -U postgres

echo "CREATE USER <OOZIEUSER> WITH PASSWORD '<OOZIEPASSWORD>';"
| psql -U postgres

echo "GRANT ALL PRIVILEGES ON DATABASE <OOZIEDATABASE> TO
<OOZIEUSER>;" | psql -U postgres
```

- Where <OOZIEUSER> is the Oozie user name, <OOZIEPASSWORD> is the Oozie user password and <OOZIEDATABASE> is the Oozie database name.

8.3.4. Troubleshooting Oozie

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

8.3.4.1. Problem: Oozie Server Install Fails Using MySQL

Check the install log:

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

The MySQL JDBC.jar file cannot be found.

8.3.4.1.1. Solution

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

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

Check the install log:

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

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

8.3.4.2.1. 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 wizard continues to fail, get more information by connecting directly to the Oozie server and executing the following command as <OOZIEUSER>:

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

9. Setting up an Internet Proxy Server for Ambari

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

How To Set Up an Internet Proxy Server for Ambari

9.1. How To Set Up an Internet Proxy Server for Ambari

1. On the Ambari Server host, add proxy settings to the following script: `/var/lib/ambari-server/ambari-env.sh`.

   ```
   -Dhttp.proxyHost=<yourProxyHost> -Dhttp.proxyPort=<yourProxyPort>
   ```

2. Optionally, to prevent some host names from accessing the proxy server, define the list of excluded hosts, as follows:

   ```
   -Dhttp.nonProxyHosts=<pipe|separated|list|of|hosts>
   ```

3. If your proxy server requires authentication, add the user name and password, as follows:

   ```
   -Dhttp.proxyUser=<username> -Dhttp.proxyPassword=<password>
   ```

4. Restart the Ambari Server to pick up this change.

If you plan to use local repositories, see Using a Local Repository. Configuring Ambari to use a proxy server and have Internet access is not required. The Ambari Server must have access to your local repositories.

10. Configuring Network Port Numbers

This chapter lists port number assignments required to maintain communication between Ambari Server, Ambari Agents, and Ambari Web.

- Default Network Port Numbers - Ambari
- Optional: Changing the Default Ambari Server Port

For more information about configuring port numbers for Stack components, see Configuring Ports in the HDP Stack documentation.

10.1. Default Network Port Numbers - Ambari

The following table lists the default ports used by Ambari Server and Ambari Agent services.

Service	Servers	Default Ports Used	Protocol	Description	Need End User Access?	Configuration Parameters
Ambari Server	Ambari Server host	8080 See Optional: Change the Ambari Server Port for instructions on changing the default port.	http See Optional: Set Up HTTPS for Ambari Server for instructions on enabling HTTPS.	Interface to Ambari Web and Ambari REST API	No	
Ambari Server	Ambari Server host	8440	https	Handshake Port for Ambari Agents to Ambari Server	No	
Ambari Server	Ambari Server host	8441	https	Registration and Heartbeat Port for Ambari Agents to Ambari Server	No	
Ambari Agent	All hosts running Ambari Agents	8670 You can change the Ambari Agent ping port in the Ambari Agent configuration.	tcp	Ping port used for alerts to check the health of the Ambari Agent	No	

10.2. Optional: Changing the Default Ambari Server Port

By default, Ambari Server uses port 8080 to access the Ambari Web UI and the REST API. To change the port number, you must edit the Ambari properties file.

Ambari Server should not be running when you change port numbers. Edit `ambari.properties` before you start Ambari Server the first time or stop Ambari Server before editing properties.

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. Ambari Server now accesses Ambari Web via the newly configured port:

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

11. Changing the JDK Version on an Existing Cluster

During your initial Ambari Server Setup, you selected the JDK to use or provided a path to a custom JDK already installed on your hosts. After setting up your cluster, you may change the JDK version using the following procedure.

How to change the JDK Version for an Existing Cluster

11.1. How to change the JDK Version for an Existing Cluster

1. Re-run Ambari Server Setup.

   ```
   ambari-server setup
   ```

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

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

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

   ```
   [1] - Oracle JDK 1.7

   [2] - Oracle JDK 1.6

   [3] - Custom JDK Enter choice: 3
   ```

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

4. If you choose `Custom JDK`, verify or add the custom JDK path on all hosts in the cluster.

5. After setup completes, you must restart each component for the new JDK to be used by the Hadoop services.

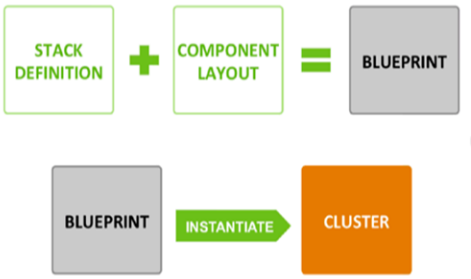
   Using the Ambari Web UI, do the following tasks:

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

For more information about managing services in your cluster, see Managing Services.

12. Using Ambari Blueprints

Ambari Blueprints provide an API to perform cluster installations. You can build a reusable "blueprint" that defines which Stack to use, how Service Components should be laid out across a cluster and what configurations to set.



After setting up a blueprint, you can call the API to instantiate the cluster by providing the list of hosts to use. The Ambari Blueprint framework promotes reusability and facilitates automating cluster installations without UI interaction.

Learn more about Ambari Blueprints API on the Ambari Wiki.

13. Configuring HDP Stack Repositories for Red Hat Satellite

As part of installing HDP Stack with Ambari, `HDP.repo` and `HDP-UTILS.repo` files are generated and distributed to the cluster hosts based on the Base URL user input from the Cluster Install Wizard during the Select Stack step. In cases where you are using Red Hat Satellite to manage your Linux infrastructure, you can disable the repositories defined in the HDP Stack .repo files and instead leverage Red Hat Satellite.

How To Configure HDP Stack Repositories for Red Hat Satellite

13.1. How To Configure HDP Stack Repositories for Red Hat Satellite

To disable the repositories defined in the HDP Stack.repo files:

1. Before starting the Ambari Server and installing a cluster, on the Ambari Server browse to the Stacks definition directory.

   ```
   cd /var/lib/ambari-server/resources/stacks/
   ```

2. Browse the install hook directory:

   For HDP 2.0 or HDP 2.1 Stack

   ```
   cd HDP/2.0.6/hooks/before-INSTALL/templates
   ```

   For HDP 1.3 Stack

   ```
   cd HDP/1.3.2/hooks/before-INSTALL/templates
   ```

3. Modify the.repo template file

   ```
   vi repo_suse_rhel.j2
   ```

4. Set the enabled property to 0 to disable the repository.

   ```
   enabled=0
   ```

5. Save and exit.

6. Start the Ambari Server and proceed with your install.

The .repo files will still be generated and distributed during cluster install but the repositories defined in the .repo files will not be enabled.

Important

You must configure Red Hat Satellite to define and enable the Stack repositories. Please refer to the Red Hat Satellite documentation for more information.

14. Tuning Ambari Performance

For clusters larger than 200 nodes, calculate and set a larger task cache size on the Ambari server.

1. Calculate the new, larger cache size, using the following relationship:

   `ecCacheSizeValue=60*`<cluster_size> where <cluster_size> is the number of nodes in the cluster.

2. On the Ambari Server host, in `etc/ambari-server/conf/ambari-properties`, add the following property and value:

   `server.ecCacheSize=`<ecCacheSizeValue> where <ecCacheSizeValue> is the value calculated previously, based on the number of nodes in the cluster.

3. Restart Ambari Server.

   ```
   ambari-server restart
   ```

15. Using Ambari Views

Ambari includes the Ambari Views Framework, which allows for developers to create UI components that "plug into" the Ambari Web interface. Ambari includes a built-in set of Views that are pre-deployed. This section describes the views that are included with Ambari and their configuration.

View	Description	HDP Stacks
Tez	View information related to Tez jobs that are executing on the cluster.	HDP 2.2 or later
Slider	A tool to help deploy and manage Slider-based applications.	HDP 2.1 or later
Jobs	A visualization tool for Hive queries that execute on the Tez engine.	HDP 2.1 or later

- **Topics**
- Using Tez View
- Using Slider View
- Using Jobs View

Learning More About Views

You can learn more about the Views Framework at the following resources:

Resource	URL
Administering Views	Ambari Administration Guide - Managing Views
Ambari Project Wiki	https://cwiki.apache.org/confluence/display/AMBARI/Views
Example Views	https://github.com/apache/ambari/tree/trunk/ambari-views/examples
View Contributions	https://github.com/apache/ambari/tree/trunk/contrib/views

15.1. Tez View

Tez is a general, next-generation execution engine like MapReduce that can efficiently execute jobs from multiple applications such as Apache Hive and Apache Pig. When you run a job such as a Hive query or Tez script using Tez, you can use the Tez View to track and debug the execution of that job. Topics in this section describe how to configure, deploy and use the Tez View to execute jobs in your cluster.

15.1.1. Configuring Tez in Your Cluster

In your cluster, confirm the following configurations are set:

Configuration	Property	Comments
yarn-site.xml	yarn.resourcemanager.system-metrics-publisher.enabled	Enable generic history service in timeline server. Verify that this property is set=true.
yarn-site.xml	yarn.timeline-service.enabled	Enabled the timeline server for logging details.
yarn-site.xml	yarn.timeline-service.webapp.address	Value must be the IP:PORT on which timeline server is running

15.1.2. Deploying the Tez View

To deploy the Tez View, you must first configure Ambari for Tez, and then configure Tez to make use of the Tez View in Ambari.

1. Configure Ambari for Tez.

   a. From the Ambari Administration interface, browse to the Views section.

   b. Click to expand the Tez view and click Create Instance.

   c. Enter the instance name, the display name and description.

   d. Enter the configuration properties for your cluster.

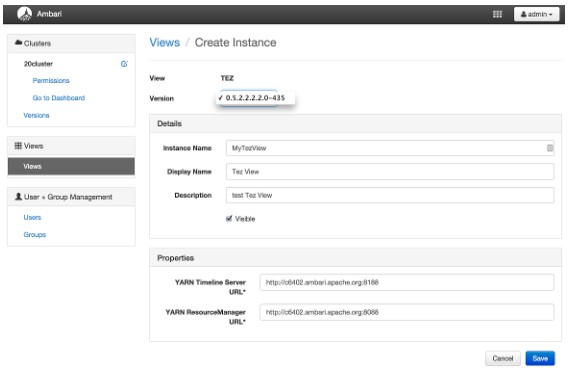
Property	Description	Example
YARN Timeline Server URL (required)	The URL to the YARN Application Timeline Server, used to provide Tez information. Typically this is the yarn.timeline-service.webapp.address property in the yarn-site.xml configuration. **URL must be accessible from Ambari Server host**.	http://yarn.timeline-service.hostname:8188
YARN ResourceManager URL (required)	The URL to the YARN ResourceManager, used to provide YARN Application data. Typically this is the yarn.resourcemanager.webapp.address property in the yarn-site.xml configuration. **URL must be accessible from Ambari Server host**.	http://yarn.timeline-service.hostname:8088

Important

As in the following examples, URLs that you provide as properties in the Views configuration for both YARN Timeline Server and YARN ResourceManager

must

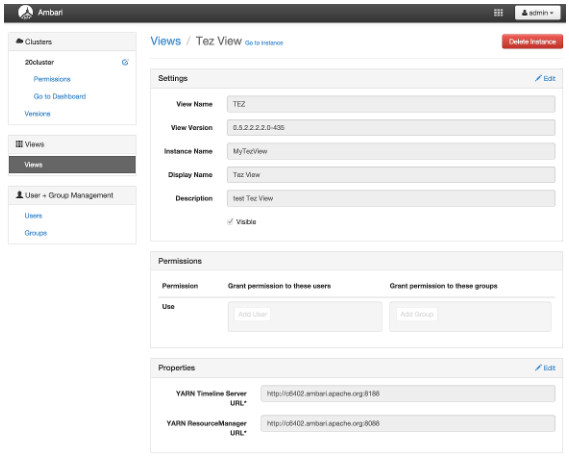
include "http://" .



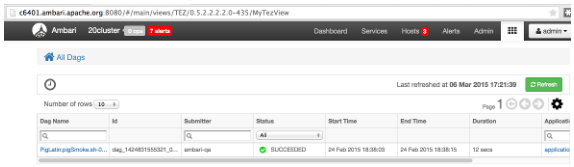
   e. Save the View.

2. Configure Tez to make use of the Tez View in Ambari:

   a. From `Ambari > Admin`, Open the Tez View, then choose "Go To Instance".



b. Copy the URL for the Tez View from your web browser's address bar.



c. Select `Services > Tez > Configs`.

d. In `custom tez-site`, add the following property: `Key: tez.tez-ui.history-url.base Value: <Tez View URL>` where `<Tez View URL>` is the the URL you copied from the browser session for the open Tez View.

e. Restart Tez.

f. Restart Hive.

For more information about managing Ambari Views, see Managing Views in the Ambari Administration Guide.

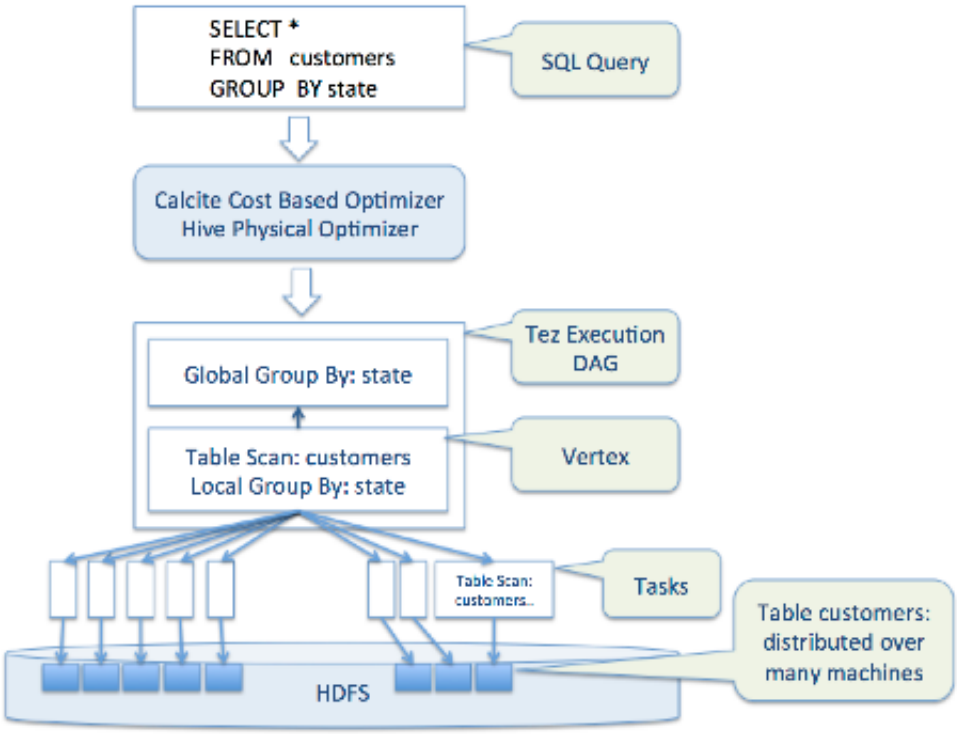
Note

If your cluster is configured for Kerberos,you must set up Ambari Server for Kerberos for the Tez view to access the ATS component. For more information, see Set Up Kerberos for Ambari Server.

15.1.3. Hive SQL on Tez - DAG, Vertex and Task

In Hive, the user query written in SQL is compiled and for execution converted into a Tez execution graph, or more precisely a Directed Acyclic Graph (DAG). A DAG is a collection of Vertices where each Vertex executes a part, or fragment of the user Query. The directed connections between Vertices determine the order in which they are executed. For example, the Vertex to read a table has to be run before a filter can be applied to the rows of that table.

Let's say that a Vertex reads a user table. This table can be very large and distributed across multiple machines and multiple racks. So, this table read is achieved by running many tasks in parallel. Here is a simplified example using a sample query that shows the execution of a SQL query in Hive.



Executing a SQL query in Hive

The Tez View tool lets your more easily understand and debug any submitted Tez job. Examples of Tez jobs include: a Hive query or Pig script executed using the Tez execution engine. Specifically, Tez helps you do the following tasks:

- Identify the Tez DAG for your Job
- Better Understand How your Query is Being Executed
- Identify the Cause of a Failed Job
- Identify the Cause of a Slow-Performing Job

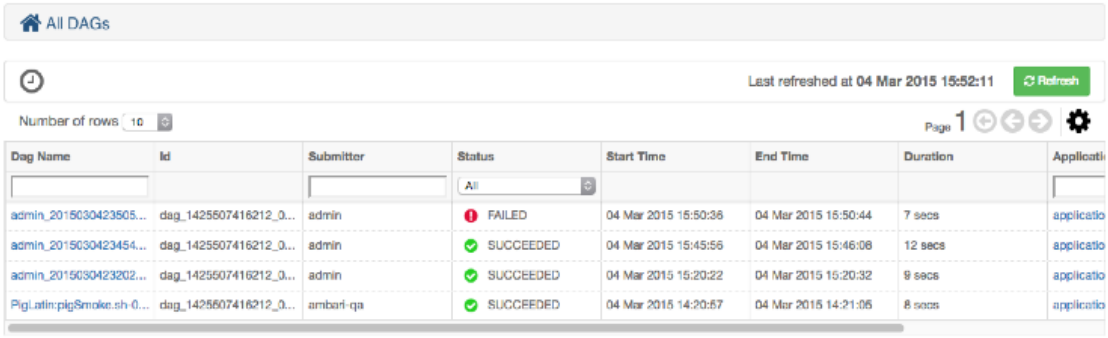
- Monitor Task Progress for a Job

15.1.4. Identify the Tez DAG for your job

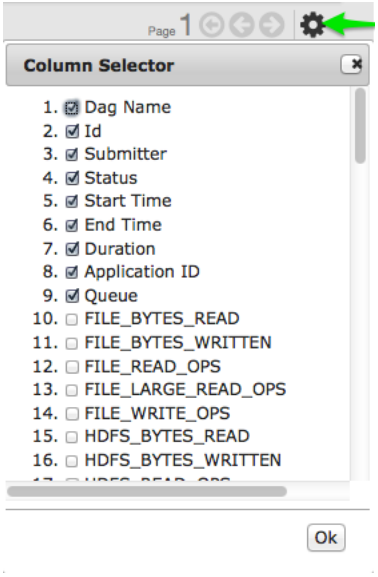
The Tez View displays a list of jobs sorted by time, latest first. You can search a job using the following fields:

- DagID
- User
- Start Time
- **Tez Job Status Descriptions**

Status	Description
Submitted	The DAG has been submitted to Tez but has not started running yet.
Running	The DAG is currently running.
Succeeded	The DAG completed successfully.
Failed	The DAG failed to complete successfully.
Killed	The DAG was stopped manually.
Error	An internal error occurred when executing the DAG.



The Tez View is the primary entry point for finding a Tez job. At this point, no other UI links to the Tez View. To select columns shown in the Tez View, choose the wheel icon, select field names, then choose OK.

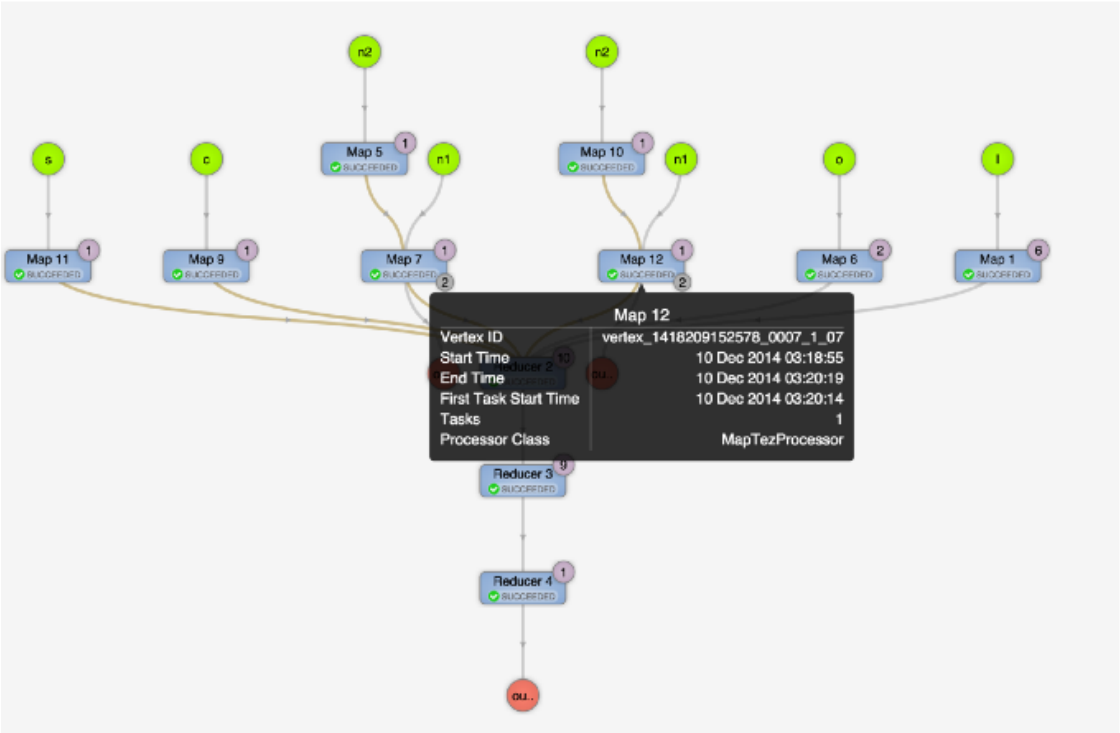


15.1.5. Better Understand How your Job is being Executed

This is the primary use case that was not available earlier. Users were not able to get insight into how their tasks are running. This allows the user to identify the complexity and progress of a running job.

The View Tab shows the following:

- The DAG graphical view
- All Vertices
- Tasks per Vertex on top right of Vertex
- Failed Vertex displays red to provide visual contrast with successful vertices that display green
- Details of timelines are available on mouse-over on a Vertex



The View Tab provides a launching place to further investigate the Vertices that have failures or are taking time.

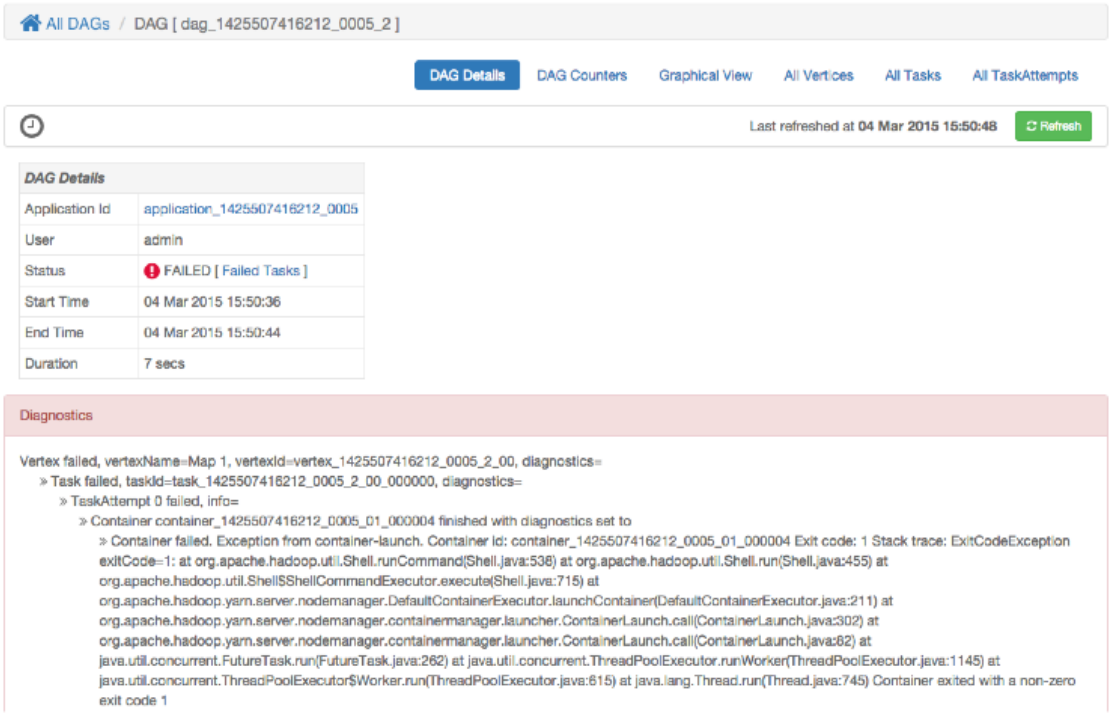
15.1.6. Identify the Cause of a Failed Job

Previously, a Tez task that failed gave an error code such as 1. Someone familiar with Tez error logs had to log in and review the logs to find why a particular task failed. The Tez View exposes errors in a way that you can more easily find and report.

When a Tez task fails, you must be able to:

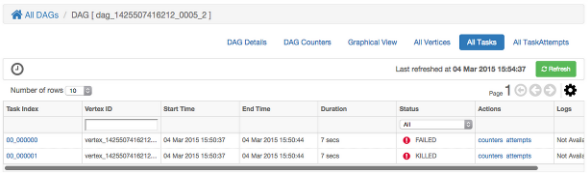
- Identify the reason for task failure
- Capture the reason for task failure

When a Tez task fails, the Tez Detail Tab show the failure as follows:



15.1.7. See Details of Failing Tasks

Multiple task failures may occur. The Tez Tasks Tab lets you see all tasks that failed and examine the reason and logs for each failure. Logs for genuine failures; not for killed tasks are available to download from the Tez Tasks Tab.



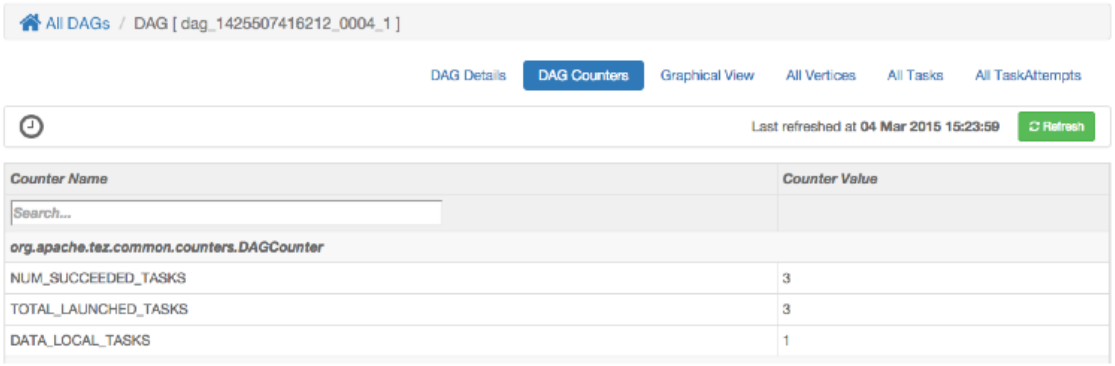
15.1.8. Identify the Cause of a Slow-Performing Job

The Tez View shows counters at the Vertex and Task levels that let you understand why a certain task is performing more slowly than expected.

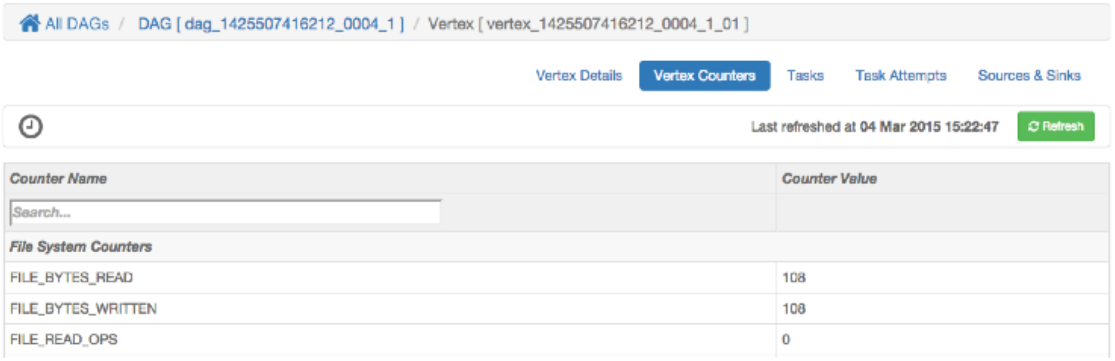
15.1.9. Counters at Vertex and Task Level

Counters are available at the DAG, Vertex, and Task levels Counters help you understand the task size better and find any anomalies. Elapsed time is one of the primary counters to look at.

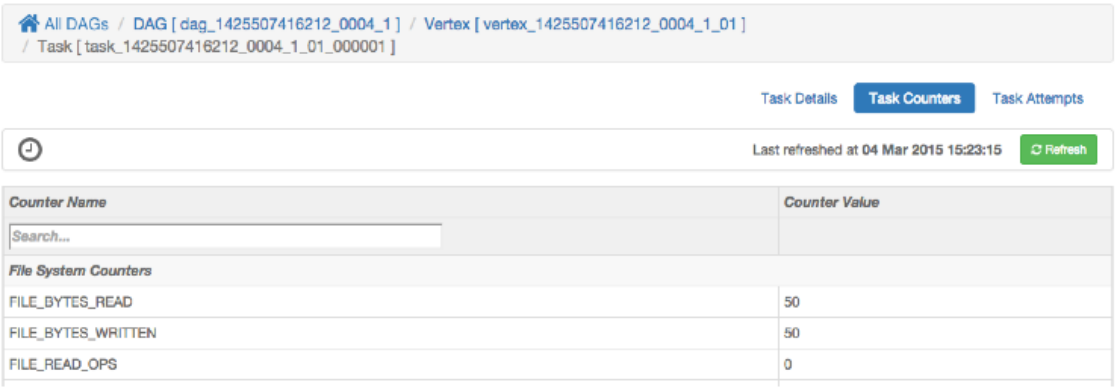
DAG-level Counters



Vertex-level Counters



Task-level Counters



15.1.10. Monitor Task Progress for a Job

The Tez View shows task progress by increasing count of completed tasks and total tasks. This allows you identify hung tasks and get insight into long running tasks.

15.2. Using the Jobs View

The Jobs view provides a visualization for Hive queries that have executed on the Tez engine.

15.2.1. Deploying the Jobs View

Refer to the Ambari Administration guide for general information about Managing Views.

1. From the Ambari Administration interface, browse to the Views section.
2. Click to expand the `Jobs` view and click `Create Instance`.
3. Enter the instance name, the display name and description.
4. Enter the configuration properties for your cluster.

Property	Description	Example
yarn.ats.url (required)	The URL to the YARN Application Timeline Server, used to provide Tez information. Typically this is the yarn.timeline-service.webapp.address property in the yarn-site.xml configuration. **URL must be accessible from Ambari Server host**.	http://yarn.timeline-service.hostname:818
yarn.resourcemanager.url (required)	The URL to the YARN ResourceManager, used to provide YARN Application data. Typically this is the yarn.resourcemanager.webapp.address property in the yarn-site.xml configuration. **URL must be accessible from Ambari Server host**.	http://yarn.timeline-service.hostname:808

5. Save the view.

Note

If your cluster is configured for Kerberos, you must set up the Ambari Server for Kerberos to provide the Jobs view access to the ATS component. For more information, see Set Up Kerberos for Ambari Server in the Ambari Security Guide.

15.3. Using the Slider View

Slider is a framework for deploying and managing long-running applications on YARN. When applications are packaged using Slider for YARN, the **Slider View** can be used to help deploy and manage those applications from Ambari.

15.3.1. Deploying the Slider View

Refer to the Ambari Administration guide for general information about Managing Views.

1. From the Ambari Administration interface, browse to the Views section.
2. Click to expand the **Slider** view and click **Create Instance**.
3. Enter the instance name, the display name and description.
4. Enter the configuration properties for your cluster.

Property	Description	Example
Ambari Server URL (required)	The Ambari REST URL to the cluster resource.	http://ambari.server:8080/api/v1/clusters/MyCluster

Property	Description	Example
Ambari Server Username (required)	The username to connect to Ambari. Must be an Ambari Admin user.	admin
Ambari Server Password (required)	The password for the Ambari user.	password
Slider User	The user to deploy slider applications as. By default, the applications will be deployed as the "yarn" service account user. To use the current logged-in Ambari user, enter `${username}`.	joe.user or ${username}
Kerberos Principal	The Kerberos principal for Ambari views. This principal identifies the process in which the view runs. Only required if your cluster is configured for Kerberos. Be sure to configure the view principal as a proxy user in core-site.	view-principal@EXAMPLE.CO
Kerberos Keytab	The Kerberos keytab for Ambari views. Only required if your cluster is configured for Kerberos.	/path/to/keytab/view-principal.headless.keytab

5. Save the view.