

Hortonworks Data Platform 2.0

Reference

(Mar 18, 2013)

Hortonworks Data Platform 2.0 : Reference

Copyright © 2012, 2013 Hortonworks, Inc. Some rights reserved.

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

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

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



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

1. Hadoop Service Accounts	1
2. Configuring Ports	2
2.1. HDFS Ports	2
2.2. YARN Ports	2
2.3. Hive Ports	2
2.4. Tez Ports	3
2.5. HBase Ports	3
2.6. MySQL Ports	3
3. Controlling HDP Services Manually	5
3.1. Starting HDP Services	5
3.2. Stopping HDP services	7
4. Deploying HDP In Production Data Centers with Firewalls	9
4.1. Deployment Strategies for Data Centers with Firewalls	9
4.1.1. Terminology	9
4.1.2. Options for Mirroring or Proxying	10
4.1.3. Considerations for choosing a Mirror or Proxy solution	11
4.2. Recommendations for Deploying HDP	11
4.2.1. RPMs in the HDP repository	12
4.3. Detailed Instructions for Creating Mirrors and Proxies	12
4.3.1. Option I - Mirror server has no access to the Internet	13
4.3.2. Option II - Mirror server has temporary access to the Internet	15
4.3.3. Option III - Mirror server has permanent access to the Internet	19
4.3.4. Option IV - Trusted proxy server	19

List of Tables

1.1. Hadoop Service Accounts	1
2.1. HDFS Ports	2
2.2. YARN Ports	2
2.3. Hive Ports	3
2.4. Tez Ports	3
2.5. HBase Ports	3
2.6. MySQL Ports	4
4.1. Terminology	9
4.2. Comparison - HDP Deployment Strategies	11
4.3. Deploying HDP - Option I	13
4.4. Deploying HDP - Option II	17

1. Hadoop Service Accounts

This topic provides information about the service users for Hadoop. HDP Installer creates the following service users:



Note

The user names for these service users cannot be modified.

Table 1.1. Hadoop Service Accounts

HDP Service	User Name	Notes
HDFS	hdfs	NameNode, Secondary NameNode, and the DataNodes run as the <code>hdfs</code> user.
YARN	yarn	ResourceManager and the NodeManagers run as the <code>yarn</code> user.
HBase	hbase	HBase Master and the RegionServers run as the <code>hbase</code> user.
Hive Metastore	hive	Hive Metastore runs as the <code>hive</code> user.
Zookeeper	zookeeper	Zookeeper server runs as the <code>zookeeper</code> user.

2. 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 HDP.

2.1. HDFS Ports

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

Table 2.1. HDFS Ports

Service	Servers	Default Ports Used	Protocol
NameNode WebUI	Master Nodes (NameNode and any back-up NameNodes)	50070	http
		50470	https
NameNode metadata service		8020/9000	IPC
DataNode	All Slave Nodes	50075	http
		50475	https
		50010	
		50020	IPC
Secondary NameNode	Secondary NameNode and any backup Secondary NameNode	50090	http

2.2. YARN Ports

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

Table 2.2. YARN Ports

Service	Servers	Default Ports Used	Protocol
Resource Manager WebUI	Master Nodes (Resource Manager and any back-up Resource Manager node)	8088	http
Resource Manager	Master Nodes (Resource Manager Node)	8032	IPC
NodeManager Web UI	All Slave Nodes	50060	http

2.3. Hive Ports

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



Note

Neither of these services are used in a standard HDP installation.

Table 2.3. Hive Ports

Service	Servers	Default Ports Used	Protocol
Hive Server	Hive Server machine (Usually a utility machine)	10000	
Hive Web UI	Hive Server machine (Usually a utility machine)	9999	http
Hive Metastore		9933	http

2.4. Tez Ports

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

Table 2.4. Tez Ports

Service	Servers	Default Ports Used	Protocol
Tez AM, Tez Service		12999	
		10030	http

2.5. HBase Ports

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

Table 2.5. HBase Ports

Service	Servers	Default Ports Used	Protocol
HMaster	Master Nodes (HBase Master Node and any back-up HBase Master node)	60000	
HMaster Info Web UI	Master Nodes (HBase master Node and back up HBase Master node if any)	60010	http
Region Server	All Slave Nodes	60020	
Region Server	All Slave Nodes	60030	http
	All ZooKeeper Nodes	2888	
	All ZooKeeper Nodes	3888	
		2181	

2.6. MySQL Ports

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

Table 2.6. MySQL Ports

Service	Servers	Default Ports Used	Protocol
MySQL	MySQL database server	3306	

3. Controlling HDP Services Manually

In this document:

- [Starting HDP Services](#)
- [Stopping HDP Services](#)

3.1. Starting HDP Services

Start all the Hadoop services in the following order:

- HDFS
- YARN
- ZooKeeper
- HBase
- Hive Metastore
- HiveServer2
- Tez
- WebHCat

Instructions

1. Start HDFS

- a. Execute these commands on the NameNode host machine:

```
su -l hdfs -c "/usr/lib/hadoop/bin/hadoop-daemon.sh --config /etc/hadoop/conf start namenode"
```

- b. Execute these commands on the Secondary NameNode host machine:

```
su -l hdfs -c "/usr/lib/hadoop/bin/hadoop-daemon.sh --config /etc/hadoop/conf start secondarynamenode"
```

- c. Execute these commands on all DataNodes:

```
su -l hdfs -c "/usr/lib/hadoop/bin/hadoop-daemon.sh --config /etc/hadoop/conf start datanode"
```

2. Start YARN

- a. Execute these commands on the ResourceManager host machine:

```
su -l yarn
```

```
/usr/lib/hadoop-yarn/sbin/yarn-daemon.sh --config $HADOOP_CONF_DIR start resourcemanager
```

- b. Execute these commands on the JobTracker History Server host machine:

```
su -l mapred -c "/usr/lib/hadoop-mapreduce/sbin/mr-jobhistory-daemon.sh
start historyserver --config $HADOOP_CONF_DIR"
```

- c. Execute these commands on all NodeManagers:

```
su -l yarn -c "/usr/lib/hadoop-yarn/sbin/yarn-daemon.sh --
config $HADOOP_CONF_DIR start nodemanager"
```

3. Start ZooKeeper. Execute these commands on the ZooKeeper host machine machine(s).

```
su - zookeeper -c "export ZOOCFGDIR=/etc/zookeeper/conf ; export ZOOCFG=
zoo.cfg ; source /etc/zookeeper/conf/zookeeper-env.sh ; /usr/lib/zookeeper/
bin/zkServer.sh start"
```

4. Start HBase

- a. Execute these commands on the HBase Master host machine:

```
su -l hbase -c "/usr/lib/hbase/bin/hbase-daemon.sh --config /etc/hbase/
conf start master; sleep 25"
```

- b. Execute these commands on all RegionServers:

```
su -l hbase -c "/usr/lib/hbase/bin/hbase-daemon.sh --config /etc/hbase/
conf start regionserver"
```

5. Start Hive Metastore. On the Hive Metastore host machine, execute the following command:

```
su -l hive -c "env HADOOP_HOME=/usr nohup hive --service metastore
> $HIVE_LOG_DIR /hive.out 2> $HIVE_LOG_DIR /hive.log &"
```

where, `$HIVE_LOG_DIR` is the directory where Hive server logs are stored. For example, `/var/logs/hive`.

6. Start HiveServer2. On the Hive Server2 host machine, execute the following command:

```
sudo su hive -c "nohup /usr/lib/hive/bin/hiveserver2 -hiveconf hive.
metastore.uris=\" \" > $HIVE_LOG_DIR /hiveServer2.out 2>$HIVE_LOG_DIR/
hiveServer2.log &"
```

where `$HIVE_LOG_DIR` is the directory where Hive server logs are stored. For example, `/var/logs/hive`.

7. Start Tez AM and/or Tez Service.

Execute the following command on the Tez host machine:

```
$TEZ_HOME/sbin/tez-daemon.sh start ampoolservice
```

where, `$TEZ_HOME` is the location of the directory that contains all the Tez JAR files. By default, `$TEZ_HOME` is set to `/usr/lib/tez`.

8. Start WebHCat. On the WebHCat host machine, execute the following command:

```
su -l hcat -c "/usr/lib/hcatalog/sbin/webhcat_server.sh start"
```

3.2. Stopping HDP services

Before trying any upgrades or uninstalling software, stop all the hadoop services in the following order::

- WebHCat
- Tez
- HiveServer2
- Hive Metastore
- ZooKeeper
- HBase
- YARN
- HDFS

Instructions

1. Stop WebHCat. On the WebHCat host machine, execute the following command:

```
su -l hcat -c "/usr/lib/hcatalog/sbin/webhcat_server.sh stop"
```

2. Stop Tez AM and/or Tez Service.

Execute the following command on the Tez host machine:

```
$TEZ_HOME/sbin/tez-daemon.sh stop ampoolservice
```

where, *\$TEZ_HOME* is the location of the directory that contains all the Tez JAR files. By default, *\$TEZ_HOME* is set to `/usr/lib/tez`.

3. Stop Hive. Execute these commands on the Hive Metastore and Hive Server2 host machine.

```
ps aux | awk '{print $1,$2}' | grep hive | awk '{print $2}' | xargs kill >/dev/null 2>&1
```

4. Stop ZooKeeper. Execute these commands on the ZooKeeper host machine

```
su - zookeeper -c "export ZOOCFGDIR=/etc/zookeeper/conf ; export ZOOCFG=zoo.cfg ;source /etc/zookeeper/conf/zookeeper-env.sh ; /usr/lib/zookeeper/bin/zkServer.sh stop"
```

5. Stop HBase

- a. Execute these commands on all RegionServers:

```
su -l hbase -c "/usr/lib/hbase/bin/hbase-daemon.sh --config /etc/hbase/conf stop regionserver"
```

- b. Execute these commands on the HBase Master host machine:

```
su -l hbase -c "/usr/lib/hbase/bin/hbase-daemon.sh --config /etc/hbase/conf stop master"
```

6. Stop YARN

- a. Execute these commands on all NodeManagers:

```
su -l yarn -c "/usr/lib/hadoop-yarn/sbin/yarn-daemon.sh --config $HADOOP_CONF_DIR stop nodemanager"
```

- b. Execute these commands on the JobTracker History Server host machine:

```
su -l mapred -c "/usr/lib/hadoop-mapreduce/sbin/hadoop-daemon.sh --config /etc/hadoop/conf stop historyserver"
```

- c. Execute these commands on the ResourceManager host machine:

```
su -l yarn
```

```
/usr/lib/hadoop-yarn/sbin/yarn-daemon.sh --config $HADOOP_CONF_DIR stop resourcemanager
```

7. Stop HDFS

- a. Execute these commands on all DataNodes:

```
su -l hdfs -c "/usr/lib/hadoop/bin/hadoop-daemon.sh --config /etc/hadoop/conf stop datanode"
```

- b. Execute these commands on the Secondary NameNode host machine:

```
su -l hdfs -c "/usr/lib/hadoop/bin/hadoop-daemon.sh --config /etc/hadoop/conf stop secondarynamenode"
```

- c. Execute these commands on the NameNode host machine:

```
su -l hdfs -c "/usr/lib/hadoop/bin/hadoop-daemon.sh --config /etc/hadoop/conf stop namenode"
```

4. Deploying HDP In Production Data Centers with Firewalls

In this document:

- [Deployment strategies for data centers with firewall](#)
- [Recommendations for deploying HDP](#)
- [Detailed instructions for creating mirrors and proxies](#)

4.1. Deployment Strategies for Data Centers with Firewalls

A typical Hortonworks Data Platform (HDP) install requires access to the Internet in order to fetch software packages from a remote repository. Since corporate networks typically have various levels of firewalls, these firewalls may limit or restrict Internet access, making it impossible for your cluster nodes to access the HDP repository during the install process.

The solution for this is to either:

- Create a local mirror repository inside your firewall hosted on a local mirror server inside your firewall; or
- Provide a trusted proxy server inside your firewall that can access the hosted repositories.

This document will cover these two options in detail, discuss the trade-offs, provide configuration guidelines, and will also provide recommendations for your deployment strategy.

In general, before installing Hortonworks Data Platform in a production data center, it is best to ensure that both the Data Center Security team and the Data Center Networking team are informed and engaged to assist with these aspects of the deployment.

4.1.1. Terminology

Table 4.1. Terminology

Item	Description
Yum Package Manager (yum)	A package management tool that fetches and installs software packages and performs automatic dependency resolution. See http://yum.baseurl.org/ for more information.
Local Mirror Repository	The yum repository hosted on your Local Mirror Server that will serve the HDP software.
Local Mirror Server	The server in your network that will host the Local Mirror Repository. This server must be accessible from all hosts in your cluster where you will install HDP.
HDP Repositories	A set of repositories hosted by Hortonworks that contains the HDP software packages. HDP software packages include the HDP Repository and the HDP-UTILS Repository.

Item	Description
HDP Repository Tarball	A tarball image that contains the complete contents of the HDP Repositories.

4.1.2. Options for Mirroring or Proxying

HDP uses yum to install software, and this software is obtained from the: HDP Repositories, and the Extra Packages for Enterprise Linux (EPEL) repository.

If your firewall prevents Internet access, it will be necessary to mirror and/or proxy both the HDP repository and the Extra Packages for Enterprise Linux (EPEL) repository. Many Data Centers already mirror or proxy the EPEL repository, so discuss with your Data Center team whether EPEL is already available from within your firewall.

Mirroring a repository involves copying the entire repository and all its contents onto a local server and enabling an HTTPD service on that server to serve the repository locally. Once the local mirror server setup is complete, the `*.repo` configuration files on every repository client (i.e. cluster nodes) must be updated, so that the given package names are associated with the local mirror server instead of the remote repository server.

There are three options for creating a local mirror server. Each of these options is explained in detail in a later section.

- **Option I:** Mirror server has no access to Internet at all

Use a web browser on your workstation to download the HDP Repository Tarball, move the tarball to the selected mirror server using scp or an USB drive, and extract it to create the repository on the local mirror server.

- **Option II:** Mirror server has temporary access to Internet

Temporarily configure a server to have Internet access, download a copy of the HDP Repository to this server using the **reposync** command, then reconfigure the server so that it is back behind the firewall.

- **Option III:** Mirror server has permanent access to Internet (modified form of Option II)

Establish a “trusted host”, by permanently configuring a server to have Internet access, but still be accessible from within the firewall. Download a copy of the HDP Repository to this server using the **reposync** command.



Note

Option I is probably the least effort, and in some respects, is the most secure deployment option.

Option III is best if you want to be able to update your Hadoop installation periodically from the Hortonworks Repositories.

However, if you are considering Option III, you should also consider the fourth option, which is to proxy the HDP Repositories through a trusted proxy server. If you have a network administrator who has expertise in setting up proxies, and if the proxy option is acceptable within your Data Center Security policies, this can be the easiest of all the options.

- **Option IV: Trusted proxy server**

Proxying a repository involves setting up a standard HTTP proxy on a local server to forward repository access requests to the remote repository server and route responses back to the original requestor. Effectively, the proxy server makes the repository server accessible to all clients, by acting as an intermediary.

Once the proxy is configured, change the `/etc/yum.conf` file on every repository client (i.e. cluster nodes), so that when the client attempts to access the repository during installation, the request will go through the local proxy server instead of going directly to the remote repository server.

4.1.3. Considerations for choosing a Mirror or Proxy solution

The following table lists some benefits provided by these alternative deployment strategies:

Table 4.2. Comparison - HDP Deployment Strategies

Advantages of repository mirroring (Options I, II, and III)	Advantages of creating a proxy (Options IV)
<ul style="list-style-type: none"> • Minimizes network access (after the initial investment of copying the repository to local storage). The install process is therefore faster, reliable, and more cost effective (reduced WAN bandwidth minimizes the data center costs). • Allows security-conscious data centers to qualify a fixed set of repository files. It also ensures that the remote server will not change these repository files. • Large data centers may already have existing repository mirror servers for the purpose of OS upgrades and software maintenance. You can easily add the HDP Repositories to these existing servers. 	<ul style="list-style-type: none"> • Avoids the need for long term management of the repository files (including periodic updates for upgrades, new versions, and bug fixes). • Almost all data centers already have a setup of well-known proxies. In such cases, you can simply add the local proxy server to the existing proxies' configurations. This approach is easier compared to creating local mirror servers in data centers with no mirror server setup. • The network access is same as that required when using a mirror repository, but the source repository handles file management.

However, each of the above approaches are also known to have the following disadvantages:

- Mirrors have to be managed for updates, upgrades, new versions, and bug fixes.
- Proxy servers rely on the repository provider to not change the underlying files without notice.
- Caching proxies are necessary, because non-caching proxies do not decrease WAN traffic and do not speed up the install process.

4.2. Recommendations for Deploying HDP

This section provides information on the various components of the Apache Hadoop ecosystem.

In many data centers, the following deployment strategy may be optimal:

- Use a mirror for the HDP Repositories. The HDP Repositories are small and easily mirrored, thereby allowing secure control over the contents of the Hadoop packages accepted for use in your data center.

- Use a caching proxy for the EPEL repository, which is a well-known and trustworthy repository managed by the Fedora Project team, and which may be too large to mirror in your data center. If your data center already mirrors or proxies EPEL, use that mirror or proxy.



Note

The installer pulls many packages from the base OS repositories (repos). If you do not have a complete base OS available to all your machines at the time of installation, you may run into issues. For example, if you are using RHEL 6 your hosts must be able to access the “Red Hat Enterprise Linux Server 6 Optional (RPMs)” repo. If this repo is disabled, the installation is unable to access the `rubygems` package, which is necessary for HMC to operate.

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.

4.2.1. RPMs in the HDP repository

In the HDP repository, you will find two different source RPM for each component.

For example, for Hadoop, you should find the following two RPMs:

- `hadoop-x.x.x.x.el6.src.rpm`
- `hadoop-source-x.x.x.x.el6.i386.rpm`

The `src` and `source` are two different packages that serve the following purpose:

- The `src` package is used to re-create the binary in a given environment. You can use the `src` package of a particular component if you want to rebuild RPM for that component.
- The `source` package on the other hand, is used for reference or debugging purpose. The `source` package is particularly useful when you want to examine the source code of a particular component in a deployed cluster.

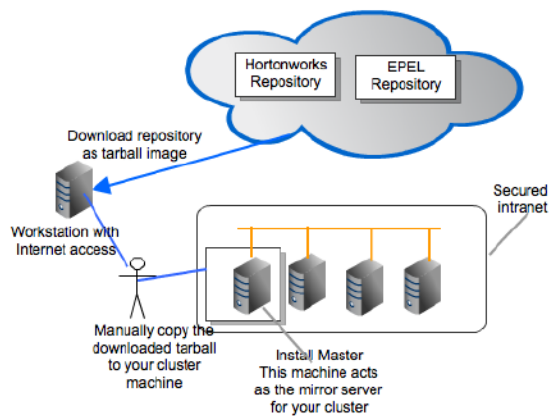
4.3. Detailed Instructions for Creating Mirrors and Proxies

In this section:

- [Option I: Mirror server has no access to the Internet](#)
- [Option II - Mirror server has temporary access to the Internet](#)
- [Option III - Mirror server has permanent access to the Internet](#)
- [Option IV - Trusted proxy server](#)

4.3.1. Option I - Mirror server has no access to the Internet

The local mirror setup for Option I is shown in the following illustration:



Complete the following instructions to set up a mirror server that has no access to the Internet:

1. [Prerequisites](#)
2. [Instructions](#)

4.3.1.1. Prerequisites

Select a mirror server host with the following characteristics:

- This server runs on either CentOS (v5.x, v6.x) or RHEL (v5.x, v6.x) and has several GB of storage available.
- This server and the cluster nodes are all running the same OS.



Note

To support repository mirroring for heterogeneous clusters requires a more complex procedure than the one documented here.

- The firewall allows all cluster nodes (the servers on which you want to install HDP) to access this server.

4.3.1.2. Instructions

1. Use a workstation with access to the Internet and download the tarball image of the appropriate Hortonworks yum repository.

Table 4.3. Deploying HDP - Option I

Cluster OS	HDP Repository Tarballs
RHEL/ CentOS 5.x	<code>wget http://public-repo-1.hortonworks.com/HDP-2.0.0.2/repos/centos5/HDP-2.0.0.2-centos5-rpmtarball.tar.gz</code>

Cluster OS	HDP Repository Tarballs
	wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.15/repos/centos5/HDP-UTILS-HDP-1.1.0.15-centos5.tar.gz
RHEL/CentOS 6.x	wget http://public-repo-1.hortonworks.com/HDP-2.0.0.2/repos/centos6/HDP-2.0.0.2-centos6-rpmtarball.tar.gz wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.15/repos/centos6/HDP-UTILS-1.1.0.15-centos6.tar.gz

2. Create an HTTP server.

- a. On the mirror server, install an HTTP server (such as Apache httpd) using the instructions provided [here](#).
- b. Activate this web server.
- c. Ensure that the firewall settings (if any) allow inbound HTTP access from your cluster nodes to your mirror server.



Note

If you are using EC2, make sure that SELinux is disabled.

3. On your mirror server, create a directory for your web server.

- For example, from a shell window, type: `mkdir -p /var/www/html/hdp/`
- If you are using a symlink, enable the `followsymlinks` on your web server.

4. Copy the HDP Repository Tarball to the directory created in step 3, and untar it.

5. Verify the configuration.

- The configuration is successful, if you can access the above directory through your web browser.

To test this out, browse to the following location: `http://yourwebserver/hdp/HDP-2.0.0.2/`

- You should see directory listing for all the HDP components along with the RPMs at: `HDP-2.0.0.2/repos/$os`.

where `$os` can be `centos5` or `centos6`.

6. Configure the `yum` clients on all the nodes in your cluster.

- a. Fetch the `yum` configuration file from your mirror server.

```
http://yourwebserver>/hdp/HDP-2.0.0.2/repos/$os/hdp.repo
```

where `$os` is either `centos5` or `centos6`.

- b. Store the `hdp.repo` file to a temporary location.

- c. Edit `hdp.repo` file changing the value of the `baseurl` property to point to your local repositories based on your cluster OS.

So, for example, if your cluster OS is CentOS 5, you would update the `hdp.repo` file to look something like this:

```
[HDP-2.0.0.2]
name=Hortonworks Data Platform Version - HDP-2.0.0.2
baseurl=http://yourwebserver/hdp/HDP-2.0.0.2/repos/centos5
gpgcheck=1
gpgkey=http://public-repo-1.hortonworks.com/HDP-2.0.0.2/repos/centos5/
RPM-GPG-KEY/RPM-GPG-KEY-Jenkins
enabled=1
priority=1

[HDP-UTILS-1.1.0.15]
name=Hortonworks Data Platform Utils Version - HDP-UTILS-1.1.0.15
baseurl=http://yourwebserver/hdp/HDP-UTILS-1.1.0.15/repos/centos5
gpgcheck=1
gpgkey=http://public-repo-1.hortonworks.com/HDP-2.0.0.2/repos/centos5/
RPM-GPG-KEY/RPM-GPG-KEY-Jenkins
enabled=1
priority=1
```

- d. Use `scp` or `pdsh` to copy the client yum configuration file to `/etc/yum.repos.d/` directory on every node in the cluster.
7. [Conditional]: If you have multiple repositories configured in your environment, deploy the following plugin on all the nodes in your cluster.
- a. Install the plugin.

- **For RHEL and CentOS v5.x**

```
yum install yum-priorities
```

- **For RHEL and CentOS v6.x**

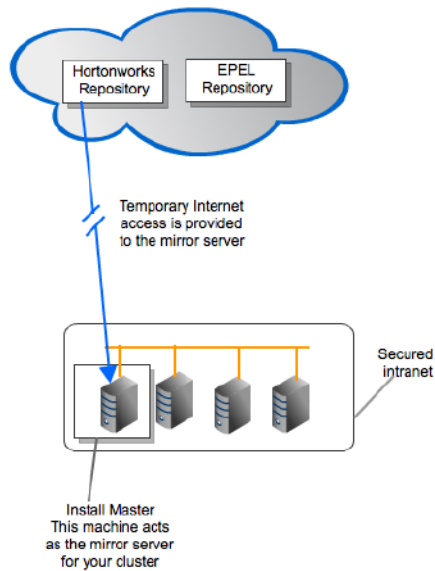
```
yum install yum-plugin-priorities
```

- b. Edit the `/etc/yum/pluginconf.d/priorities.conf` file to add the following:

```
[main]
enabled=1
gpgcheck=0
```

4.3.2. Option II - Mirror server has temporary access to the Internet

The local mirror setup for Option II is shown in the following illustration:



Complete the following instructions to set up a mirror server that has temporary access to the Internet:

1. [Prerequisites](#)
2. [Instructions](#)

4.3.2.1. Prerequisites

Select a local mirror server host with the following characteristics:

- This server runs on either CentOS (v5.x, v6.x) or RHEL (v5.x, v6.x) and has several GB of storage available.
- The local mirror server and the cluster nodes must have the same OS. If they are not running CentOS or RHEL, the mirror server must not be a member of the Hadoop cluster.



Note

To support repository mirroring for heterogeneous clusters requires a more complex procedure than the one documented here.

- The firewall allows all cluster nodes (the servers on which you want to install HDP) to access this server.
- Ensure that the mirror server has **yum** installed.
- Add the **yum-utils** and **createrepo** packages on the mirror server.

```
yum install yum-utils createrepo
```

4.3.2.2. Instructions

- Temporarily reconfigure your firewall to allow Internet access from your mirror server host.

- Execute the following command to download the appropriate Hortonworks yum client configuration file and save it in `/etc/yum.repos.d/` directory on the mirror server host.

Table 4.4. Deploying HDP - Option II

Cluster OS	HDP Repository Tarballs
RHEL/CentOS 5.x	<code>wget http://public-repo-1.hortonworks.com/HDP-2.0.0.2/repos/centos5/hdp.repo -O /etc/yum.repos.d/hdp.repo</code>
RHEL/CentOS 6.x	<code>wget http://public-repo-1.hortonworks.com/HDP-2.0.0.2/repos/centos6/hdp.repo -O /etc/yum.repos.d/hdp.repo</code>

- Create an HTTP server.
 1. On the mirror server, install an HTTP server (such as Apache `httpd`) using the instructions provided <http://httpd.apache.org/download.cgi>
 2. Activate this web server.
 3. Ensure that the firewall settings (if any) allow inbound HTTP access from your cluster nodes to your mirror server.



Note

If you are using EC2, make sure that SELinux is disabled.

- On your mirror server, create a directory for your web server.
 - For example, from a shell window, type: `mkdir -p /var/www/html/hdp/`.
 - If you are using a symlink, enable the `followsymlinks` on your web server.
- Copy the contents of entire HDP repository for your desired OS from the remote yum server to your local mirror server.
 - Continuing the previous example, from a shell window, type:

```
cd /var/www/html/hdp
reposync -r HDP-2.0.0.2
reposync -r HDP-UTILS-1.1.0.15
```

You should now see both an `HDP-2.0.0.2` directory and an `HDP-UTILS-1.1.0.15` directory, each with several subdirectories.

- Generate appropriate metadata.

This step defines each directory as a yum repository. From a shell window, type:

```
createrepo /var/www/html/hdp/HDP-2.0.0.2
createrepo /var/www/html/hdp/HDP-UTILS-1.1.0.15
```

You should see a new folder called `repodata` inside both HDP directories.

- Verify the configuration.

- The configuration is successful, if you can access the above directory through your web browser.

To test this out, browse to the following location: **`http://yourwebserver/hdp/HDP-2.0.0.2/`**

- You should now see directory listing for all the HDP components.
 - At this point, it is okay to disable external Internet access for the mirror server, so that the mirror server is once again entirely within your data center firewall.
 - Depending on your cluster OS, configure the **yum** clients on all the nodes in your cluster
1. Edit the `/etc/yum.repos.d/hdp.repo` file, changing the value of the `baseurl` property to point to your local repositories based on your cluster OS.

So, for example, if your cluster OS is CentOS 5, you would update the `hdp.repo` file to look something like this:

```
[HDP-2.0.0.2]
name=Hortonworks Data Platform Version - HDP-2.0.0.2
baseurl=http://yourwebserver/hdp/HDP-2.0.0.2/repos/centos5
gpgcheck=1
gpgkey=http://public-repo-1.hortonworks.com/HDP-2.0.0.2/repos/centos5/RPM-
GPG-KEY/RPM-GPG-KEY-Jenkins
enabled=1
priority=1

[HDP-UTILS-1.1.0.15]
name=Hortonworks Data Platform Utils Version - HDP-UTILS-1.1.0.15
baseurl=http://yourwebserver/hdp/HDP-UTILS-1.1.0.15/repos/centos5
gpgcheck=1
gpgkey=http://public-repo-1.hortonworks.com/HDP-2.0.0.2/repos/centos5/RPM-
GPG-KEY/RPM-GPG-KEY-Jenkins
enabled=1
priority=1
```

2. Use **scp** or **pdsh** to copy the client yum configuration file to `/etc/yum.repos.d/` directory on every node in the cluster.
- If you have multiple repositories configured in your environment, deploy the following plugin on all the nodes in your cluster.

1. Install the plugin.

- For RHEL and CentOS v5.x

```
yum install yum-priorities
```

- For RHEL and CentOS v6.x

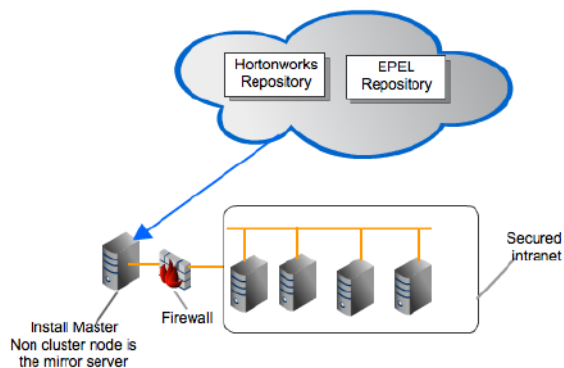
```
yum install yum-plugin-priorities
```

2. Edit the `/etc/yum/pluginconf.d/priorities.conf` file to add the following:

```
[main]
enabled=1
gpgcheck=0
```

4.3.3. Option III - Mirror server has permanent access to the Internet

The local mirror setup for Option III is shown in the following illustration:



Complete the following instructions to set up a mirror server that has permanent access to the Internet:

1. [Prerequisites](#)
2. [Instructions](#)

4.3.3.1. Prerequisites

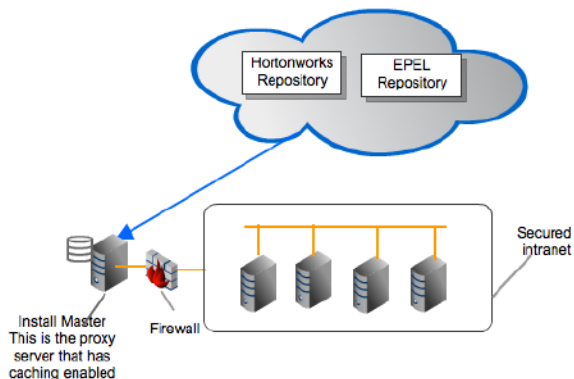
Same as [Option II](#).

4.3.3.2. Instructions

Same as [Option II](#) but Step 1 and Step 8 are unnecessary and may be skipped.

4.3.4. Option IV - Trusted proxy server

The local mirror setup for Option IV is shown in the following illustration:



Complete the following instructions to set up a trusted proxy server:

1. [Prerequisites](#)
2. [Instructions](#)

4.3.4.1. Prerequisites

Select a mirror server host with the following characteristics:

- This server runs on either CentOS (v5.x, v6.x) or RHEL (v5.x, v6.x) and has several GB of storage available.
- The firewall allows all cluster nodes (the servers on which you want to install HDP) to access this server, and allows this server to access the Internet (at least those Internet servers for the repositories to be proxied).

4.3.4.2. Instructions

1. Create a caching HTTP PROXY server on the selected host.
 - a. It is beyond the scope of this document to show how to set up an HTTP PROXY server, given the many variations that may be required, depending on your data center's network security policy. If you choose to use the Apache HTTPD server, it starts by installing `httpd`, using the instructions provided [here](#), and then adding the `mod_proxy` and `mod_cache` modules, as stated [here](#).

Please engage your network security specialists to correctly set up the proxy server.
 - b. Activate this proxy server and configure its cache storage location.
 - c. Ensure that the firewall settings (if any) allow inbound HTTP access from your cluster nodes to your mirror server, and outbound access to the desired repo sites, including `public-repo-1.hortonworks.com`.



Note

If you are using EC2, make sure that SELinux is disabled.

2. Depending on your cluster OS, configure the `yum` clients on all the nodes in your cluster.



Note

The following description is taken from the CentOS documentation [here](#).

- a. On each cluster node, add the following lines to the `/etc/yum.conf` file.

(As an example, the settings below will enable **yum** to use the proxy server **mycache.mydomain.com**, connecting to port **3128**, with the following credentials **yum-user/qwerty**.)

```
# proxy server:port number
proxy=http://mycache.mydomain.com:3128
```

```
# account details for secure yum proxy connections
proxy_username=yum-user
proxy_password=qwerty
```

- b. Once all nodes have their `/etc/yum.conf` file updated with appropriate configuration info, you can proceed with the HDP installation just as though the nodes had direct access to the Internet repositories.
- c. If this proxy configuration does not seem to work, try adding a `/` at the end of the proxy URL. For example:

```
proxy=http://mycache.mydomain.com:3128/
```