

Hortonworks Data Platform

Reference

(April 3, 2017)

Hortonworks Data Platform: Reference

Copyright © 2012-2017 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 4.0 License.
<http://creativecommons.org/licenses/by-sa/4.0/legalcode>

Table of Contents

| | |
|---|----|
| 1. Configuring Ports | 1 |
| 1.1. Accumulo Service Ports | 1 |
| 1.2. Atlas Service Ports | 2 |
| 1.3. Flume Service Ports | 2 |
| 1.4. HBase Service Ports | 3 |
| 1.5. HDFS Service Ports | 4 |
| 1.6. Hive Service Ports | 5 |
| 1.7. Hue Service Port | 5 |
| 1.8. Kafka Service Ports | 6 |
| 1.9. Kerberos Service Ports | 6 |
| 1.10. Knox Service Ports | 6 |
| 1.11. MapReduce Service Ports | 6 |
| 1.12. MySQL Service Ports | 7 |
| 1.13. Oozie Service Ports | 7 |
| 1.14. Ranger Service Ports | 7 |
| 1.15. Sqoop Service Ports | 8 |
| 1.16. Storm Service Ports | 8 |
| 1.17. Tez Ports | 9 |
| 1.18. YARN Service Ports | 9 |
| 1.19. Zeppelin Service Port | 11 |
| 1.20. ZooKeeper Service Ports | 11 |
| 2. Controlling HDP Services Manually | 13 |
| 2.1. Starting HDP Services | 13 |
| 2.2. Stopping HDP Services | 16 |
| 3. Deploying HDP In Production Data Centers With Firewalls | 20 |
| 3.1. Terminology | 20 |
| 3.2. Mirroring or Proxying | 21 |
| 3.3. Considerations for choosing a Mirror or Proxy solution | 21 |
| 3.4. Recommendations for Deploying HDP | 22 |
| 3.5. RPMs in the HDP repository | 22 |
| 3.6. Detailed Instructions for Creating Mirrors and Proxies | 23 |
| 3.6.1. Option I - Mirror server has no access to the Internet | 23 |
| 3.6.2. Option II - Mirror server has temporary or continuous access to the Internet | 27 |
| 3.7. Set up a trusted proxy server | 32 |
| 4. Hadoop Service Accounts | 34 |
| 5. Supported Database Matrix for the Hortonworks Data Platform | 35 |

List of Tables

| | |
|--|----|
| 1.1. Accumulo Service Ports | 1 |
| 1.2. Atlas Service Ports | 2 |
| 1.3. Flume Service Ports | 2 |
| 1.4. HBase Service Ports | 3 |
| 1.5. HDFS Service Ports | 4 |
| 1.6. Hive Service Ports | 5 |
| 1.7. Hue Service Port | 6 |
| 1.8. Kafka Service Ports | 6 |
| 1.9. Kerberos Service Ports | 6 |
| 1.10. Knox Service Ports | 6 |
| 1.11. MapReduce Service Ports | 7 |
| 1.12. MySQL Service Ports | 7 |
| 1.13. Oozie Service Ports | 7 |
| 1.14. Ranger Service Ports | 8 |
| 1.15. Sqoop Service Ports | 8 |
| 1.16. Storm Service Ports | 9 |
| 1.17. Tez Ports | 9 |
| 1.18. YARN Service Ports | 10 |
| 1.19. Zeppelin Service Ports | 11 |
| 1.20. ZooKeeper Service Ports | 11 |
| 3.1. Terminology | 20 |
| 3.2. Hortonworks Yum Repositories | 23 |
| 3.3. HDP Component Options | 25 |
| 3.4. Yum Client Options | 26 |
| 3.5. Yum Client Configuration Commands | 27 |
| 3.6. \$OS Parameter Values | 31 |

1. Configuring Ports

Tables in this section specify which ports must be opened for an ecosystem component or service to communicate with other components and services.

Make sure the appropriate ports are open before you install HDP.

1.1. Accumulo Service Ports

The following table lists the default ports used by the various Accumulo services. (**Note:** Neither of these services are used in a standard HDP installation.)

Table 1.1. Accumulo Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|----------------------------------|--|--------------------|----------|--|------------------------|--|
| Master | Master nodes (Active master and any standby) | 9999 | | The Master thrift server | Yes (client API needs) | <code>master.port.client</code> in <code>accumulo-site.xml</code> |
| TabletServer | Slave nodes | 9997 | | The TabletServer thrift server | Yes (client API needs) | <code>tserver.port.client</code> in <code>accumulo-site.xml</code> |
| Garbage Collector | GC nodes (Active GC and any standby) | 50091 | | The GarbageCollector thrift server | No | <code>gc.port.client</code> in <code>accumulo-site.xml</code> |
| Monitor | Monitor nodes (Active Monitor and any standby) | 50095 | HTTP(S) | Metrics/ Monitoring of an Accumulo instance | Yes | <code>monitor.port.client</code> in <code>accumulo-site.xml</code> |
| Monitor log aggregation | Monitor nodes (Active Monitor and any standby) | 4560 | | Log4j socket which accepts logs forwarded from other Accumulo services | No | <code>monitor.port.log4j</code> in <code>accumulo-site.xml</code> |
| Tracer | Tracer nodes | 12234 | | The Tracer thrift server | Yes (if enabled) | <code>trace.port.client</code> in <code>accumulo-site.xml</code> |
| Thrift Proxy (optional) | Proxy nodes | 42424 | | The Thrift Proxy server | Yes (if enabled) | <code>port</code> in <code>proxy.properties</code> |
| TabletServer Replication Service | Slave nodes | 10002 | | TabletServer Thrift service supporting multi-instance Accumulo replication | No | <code>replication.receipt.service.port</code> in <code>accumulo-site.xml</code> |
| Master Replication Service | Master nodes (Active master and any standby) | 10001 | | Master Thrift service supporting multi-instance Accumulo replication | No | <code>master.replication.coordinator.port</code> in <code>accumulo-site.xml</code> |

1.2. Atlas Service Ports

The following table lists the default ports used by Apache Atlas.

Table 1.2. Atlas Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|-------------|-------------------|--------------------|----------|--|-----------------------|--------------------------------------|
| Atlas Admin | Atlas Admin Nodes | 21000 | HTTP | Port for Atlas Admin web UI | Yes | <code>atlas.server.http.port</code> |
| Atlas | Atlas Admin Nodes | 21443 | HTTPS | Port for Atlas Admin web UI (with SSL) | Yes | <code>atlas.server.https.port</code> |

1.3. Flume Service Ports

The following table lists the default ports used by the various Flume services. (**Note:** Neither of these services are used in a standard HDP installation.)

Table 1.3. Flume Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|-------------|--------------------|----------|--|------------------------|--|
| Flume | Flume Agent | 41414 | TCP | Flume performance metrics in JSON format | Yes (client API needs) | <code>master.port.client</code> in <code>accumulo-site.xml</code> |
| Flume | HDFS Sink | 8020 | TCP | Communication from Flume into the Hadoop cluster's NameNode | Yes (client API needs) | <code>tserver.port.client</code> in <code>accumulo-site.xml</code> |
| Flume | HDFS Sink | 9000 | TCP | Communication from Flume into the Hadoop cluster's NameNode | No | <code>gc.port.client</code> in <code>accumulo-site.xml</code> |
| Flume | HDFS Sink | 50010 | TCP | Communication from Flume into the Hadoop cluster's HDFS DataNode | No | |
| Flume | HDFS Sink | 50020 | TCP | Communication from Flume into the Hadoop cluster's HDFS DataNode | No | |
| Flume | HDFS Sink | 2181 | TCP | Communication from Flume into the Hadoop cluster's ZooKeeper | No | |
| Flume | HDFS Sink | 16020 | TCP | Communication from Flume into | No | |

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|-----------------------------|--------------------|----------|---|-----------------------|--|
| | | | | the Hadoop cluster's HBase Regionserver | | |
| Flume | All Other Sources and Sinks | Variable | Variable | Ports and protocols used by Flume sources and sinks | No | Refer to the flume configuration file(s) for ports actually in use. Ports in use are specified using the port keyword in the Flume configuration file. By default Flume configuration files are located in /etc/flume/conf on Linux and c:\hdp\flume-1.4.0.x.y.z\conf on Windows |

1.4. HBase Service Ports

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

Table 1.4. HBase Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|-------------------------------------|---|--------------------|----------|--|--|------------------------------|
| HMaster | Master Nodes (HBase Master Node and any back-up HBase Master node) | 16000 | | | Yes | hbase.master.port |
| HMaster Info Web UI | Master Nodes (HBase master Node and back up HBase Master node if any) | 16010 | http | The port for the HBaseMaster web UI. Set to -1 if you do not want the info server to run. | Yes | hbase.master.info.port |
| Region Server | All Slave Nodes | 16020 | | | Yes (Typically admins, dev/ support teams) | hbase.regionserver.port |
| Region Server | All Slave Nodes | 16030 | http | | Yes (Typically admins, dev/ support teams) | hbase.regionserver.info.port |
| HBase REST Server (optional) | All REST Servers | 8080 | http | The port used by HBase Rest Servers. REST servers are optional, and not installed by default | Yes | hbase.rest.port |
| HBase REST Server Web UI (optional) | All REST Servers | 8085 | http | The port used by HBase Rest Servers web UI. REST servers are optional, and | Yes (Typically admins, dev/ | hbase.rest.info.port |

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------------------------------------|--------------------|--------------------|----------|---|---|--------------------------|
| | | | | not installed by default | support teams) | |
| HBase Thrift Server (optional) | All Thrift Servers | 9090 | | The port used by HBase Thrift Servers. Thrift servers are optional, and not installed by default | Yes | |
| HBase Thrift Server Web UI (optional) | All Thrift Servers | 9095 | | The port used by HBase Thrift Servers web UI. Thrift servers are optional, and not installed by default | Yes (Typically admins, dev/support teams) | hbase.thrift.info.port |

1.5. HDFS Service Ports

The following table lists the default ports used by the various HDFS services. (**Note:** Neither of these services are used in a standard HDP installation.)

Table 1.5. HDFS Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------------------------|---|--------------------|----------|---|---|---|
| NameNode WebUI | Master Nodes (NameNode and any back-up NameNodes) | 50070 | http | Web UI to look at current status of HDFS, explore file system | Yes (Typically admins, Dev/Support teams, as well as extra-cluster users who require webhdfs/hftp access, for example, to use distcp) | dfs.http.address |
| | | 50470 | https | Secure http service | | dfs.https.address |
| NameNode metadata service | | 8020/9000 | IPC | File system metadata operations | Yes (All clients who directly need to interact with the HDFS) | Embedded in URI specified by fs.defaultFS |
| DataNode | All Slave Nodes | 50075 | http | DataNode WebUI to access the status, logs, etc, and file data operations when using webhdfs or hftp | Yes (Typically admins, Dev/Support teams, as well as extra-cluster users who require webhdfs/hftp access, for example, | dfs.datanode.http.address |

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|--------------------|--|--------------------|----------------------|----------------------------------|-----------------------|----------------------------|
| | | 50475 | https | Secure http service | to use distcp) | dfs.datanode.https.address |
| | | 1019 | Custom HDFS protocol | Data transfer | | dfs.datanode.address |
| | | 50010 | Custom HDFS protocol | Data transfer | | dfs.datanode.address |
| | | 50020 | IPC | Metadata operations | No | dfs.datanode.ipc.address |
| Secondary NameNode | Secondary NameNode and any backup Secondary NameNode | 50090 | http | Checkpoint for NameNode metadata | No | dfs.secondary.http.address |

1.6. Hive Service 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 1.6. Hive Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|----------------|---|--------------------|-------------|--|--|--------------------------|
| Hive Server | Hive Server machine (Usually a utility machine) | 10000 | tcp or http | Service for programatically (Thrift/JDBC) connecting to Hive | Yes (Clients who need to connect to Hive either programatically or through UI SQL tools that use JDBC) | ENV Variable HIVE_PORT |
| Hive Web UI | Hive Server machine (Usually a utility machine) | 9999 | http | Web UI to explore Hive schemas | Yes | hive.hwi.listen.port |
| Hive Metastore | | 9083 | http | | Yes (Clients that run Hive, Pig and potentially M/R jobs that use HCatalog) | hive.metastore.uris |

1.7. Hue Service Port

The following table lists the default port used by the Hue web listener.

Table 1.7. Hue Service Port

| Service | Servers | Default Port Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|--------------------------|-------------------|----------|---|-----------------------|--|
| Hue | Node that is running Hue | 8888 | http | Port used by the Hue web listener to server web pages for Hue | Yes | http_port property in the /etc/hue/conf/hue.ini file |

1.8. Kafka Service Ports

The following table lists the default ports used by Kafka.

Table 1.8. Kafka Service Ports

| Service | Servers | Default Port | Default Ambari Port | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|--------------|--------------|---------------------|----------|----------------------------|-----------------------|--------------------------|
| Kafka | Kafka Server | 9092 | 6667 | TCP | The port for Kafka server. | | |

1.9. Kerberos Service Ports

The following table lists the default port used by the designated Kerberos KDC.

Table 1.9. Kerberos Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|---------------------|--------------------|----------|---------------------------------|-----------------------|--------------------------|
| KDC | Kerberos KDC server | 88 | | Port used by the designated KDC | | |

1.10. Knox Service Ports

The following table lists the default port used by Knox.

Table 1.10. Knox Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|-------------|--------------------|----------|-------------------|-----------------------|--------------------------|
| Knox | Knox server | 8443 | | Port used by Knox | | |

1.11. MapReduce Service Ports

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

Table 1.11. MapReduce Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|-----------|---------|--------------------|----------|-------------------------------------|-----------------------|-------------------------------------|
| MapReduce | | 10020 | http | MapReduce JobHistory server address | | mapreduce.jobhistory.address |
| MapReduce | | 19888 | http | MapReduce JobHistory webapp address | | mapreduce.jobhistory.webapp.address |
| MapReduce | | 13562 | http | MapReduce Shuffle Port | | mapreduce.shuffle.port |

1.12. MySQL Service Ports

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

Table 1.12. MySQL Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|-----------------------|--------------------|----------|-------------|-----------------------|--------------------------|
| MySQL | MySQL database server | 3306 | | | | |

1.13. Oozie Service Ports

The following table lists the default ports used by Oozie.

Table 1.13. Oozie Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|--------------|--------------------|----------|--|-----------------------|----------------------------------|
| Oozie | Oozie Server | 11000 | TCP | The port Oozie server runs. | Yes | OOZIE_HTTP_PORT in oozie_env.sh |
| Oozie | Oozie Server | 11001 | TCP | The admin port Oozie server runs. | No | OOZIE_ADMIN_PORT in oozie_env.sh |
| Oozie | Oozie Server | 11443 | TCP | The port Oozie server runs when using HTTPS. | Yes | OOZIE_HTTPS_PORT in oozie_env.sh |

1.14. Ranger Service Ports

The following table lists the default ports used by Ranger.

Table 1.14. Ranger Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------------------|----------------------|--------------------|----------|--|-----------------------|--|
| Ranger Admin | Ranger Admin Nodes | 6080 | HTTP | Port for Ranger Admin web UI. | Yes | ranger.service.http.port (in ranger-admin-site.xml) |
| Ranger Admin | Ranger Admin Nodes | 6182 | HTTPS | Port for Ranger Admin web UI (with SSL). | Yes | ranger.service.https.port (in ranger-admin-site.xml) |
| UNIX Auth Service | Ranger Usersync Node | 5151 | SSL/TCP | Port for UNIX Auth service. | No | ranger.usersync.port (in ranger-ugsync-site.xml) |
| Ranger KMS | Ranger KMS Nodes | 9292 | HTTP | Port for Ranger KMS. | No | ranger.service.http.port (in kms-site.xml) |
| Ranger KMS | Ranger KMS Nodes | 9293 | HTTPS | Port for Ranger KMS. | No | ranger.service.https.port (in kms-site.xml) |
| Solr used by Ranger | Solr | 6083,6183 | HTTP | Ports for auditing to Solr. | Yes | ranger-admin and all plug-ins |

1.15. Sqoop Service Ports

The following table lists the default ports used by Sqoop.

Table 1.15. Sqoop Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------|---------------|---|----------|---|-----------------------|-----------------------------|
| Sqoop | Metastore | 16000 | TCP | Connection between Sqoop and the metastore | No | sqoop.metastore.server.port |
| Sqoop | JDBC Listener | Varies, depends on target database. For example, if moving data from MySQL, TCP port 3306 must be open. | TCP | Outbound port from the Hadoop cluster to the database. Varies depending on Database | No | |

1.16. Storm Service Ports

The following table lists the default ports used by Storm.

Table 1.16. Storm Service Ports

| Service | Servers | Default Port | Default Ambari Port | Protocol | Description | Need End User Access? | Configuration Parameters |
|------------------------|---------|------------------------|---------------------|----------|---|-----------------------|--------------------------|
| ZooKeeper Port | | 2181 | | | Port used by localhost to talk to ZooKeeper. | | storm.zookeeper.port |
| DRPC Port | | 3772 | | | | | drpc.port |
| DRPC Invocations Port | | 3773 | | | | | drpc.invocations.port |
| Nimbus Thrift Port | | 6627 | | | | | nimbus.thrift.port |
| Supervisor Slots Ports | | 6700, 6701, 6702, 6703 | | | Defines the amount of workers that can be run on this machine. Each worker is assigned a port to use for communication. | | supervisor.slots.ports |
| Logviewer Port | | 8000 | | | | | logviewer.port |
| UI Port | | 8080 | 8744 | | | | ui.port |

1.17. Tez Ports

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

Table 1.17. Tez Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------------------|---------|--------------------|----------|--|--|--------------------------|
| Tez AM, Tez Service | | 12999 | | Port to use for AMPoolService status | Yes (Clients who need to submit Hive queries or jobs to Tez AM or Tez Service) Yes | tez.ampool.ws.port |
| | | 10030 | http | Address on which to run the ClientRMProxy. | | tez.ampool.address |

1.18. YARN Service Ports

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

Table 1.18. YARN Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|------------------------|---|--|----------|-----------------------------|---|--|
| Resource Manager WebUI | Master Nodes (Resource Manager and any back-up Resource Manager node) | 8088 | http | Web UI for Resource Manager | Yes | <code>yarn.resourcemanager.webapp.address</code> |
| Resource Manager | Master Nodes (Resource Manager Node) | 8050 – Default port number when you create your cluster using Ambari. 8032 – Default port number when you do not use Ambari to create your cluster. | IPC | For application submissions | Yes (All clients who need to submit the YARN applications including Hive, Hive server, Pig) | Embedded in URI specified by <code>yarn.resourcemanager.address</code> |
| Resource Manager | Master Nodes (Resource Manager Node) | 8025 | http | For application submissions | Yes (All clients who need to submit the YARN applications including Hive, Hive server, Pig) | <code>yarn.resourcemanager.resource-tracker.address</code> |
| Scheduler | Master Nodes (Resource Manager Node) | 8030 | http | Scheduler Address | Yes (Typically admins, Dev/ Support teams) | <code>yarn.resourcemanager.scheduler.address</code> |
| Resource Manager | Master Nodes (Resource Manager Node) | 8141 | http | Scheduler Address | Yes (Typically admins, Dev/ Support teams) | <code>yarn.resourcemanager.admin.address</code> |
| NodeManager | Master Nodes (NodeManager) and Slave Nodes | 45454 | http | NodeManager Address | Yes (Typically admins, Dev/ Support teams) | <code>yarn.nodemanager.address</code> |

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|---------------------|----------------------------|--------------------|----------|--------------------------------------|--|--|
| NodeManager | Master Nodes (NodeManager) | 8042 | http | NodeManager Webapp Address | Yes (Typically admins, Dev/ Support teams) | yarn.nodemanager.webapp.address |
| Timeline Server | Master Nodes | 10200 | http | Timeline Server Address | Yes (Typically admins, Dev/ Support teams) | yarn.timeline-service.address |
| Timeline Server | Master Nodes | 8188 | http | Timeline Server Webapp Address | Yes (Typically admins, Dev/ Support teams) | yarn.timeline-service.webapp.address |
| Timeline Server | Master Nodes | 8190 | https | Timeline Server Webapp https Address | Yes (Typically admins, Dev/ Support teams) | yarn.timeline-service.webapp.https.address |
| Job History Service | Master Nodes | 19888 | https | Job History Service | Yes (Typically admins, Dev/ Support teams) | yarn.log.server.url |

1.19. Zeppelin Service Port

The following table lists the default port used by Zeppelin.

Table 1.19. Zeppelin Service Ports

| Service | Servers | Default Port | Default Ambari Port | Protocol | Description |
|----------|---------|--------------|---------------------|----------|-----------------------------------|
| Zeppelin | UI port | 9995 | | TCP | The port for the Zeppelin web UI. |

1.20. ZooKeeper Service Ports

Table 1.20. ZooKeeper Service Ports

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|------------------|---------------------|--------------------|----------|---|-----------------------|--------------------------|
| ZooKeeper Server | All ZooKeeper Nodes | 2888 | | Port used by ZooKeeper peers to talk to | No | hbase.zookeeper.peerport |

| Service | Servers | Default Ports Used | Protocol | Description | Need End User Access? | Configuration Parameters |
|------------------|---------------------|--------------------|----------|---|-----------------------|--|
| | | | | each other. See here for more information. | | |
| ZooKeeper Server | All ZooKeeper Nodes | 3888 | | Port used by ZooKeeper peers to talk to each other. See here for more information. | No | <code>hbase.zookeeper.leaderport</code> |
| ZooKeeper Server | All ZooKeeper Nodes | 2181 | | Property from ZooKeeper's config <code>zoo.cfg</code> . The port at which the clients will connect. | No | <code>hbase.zookeeper.property.clientPort</code> |

2. Controlling HDP Services Manually

In this document:

- [Starting HDP Services \[13\]](#)
- [Stopping HDP Services \[16\]](#)

2.1. Starting HDP Services

Start the Hadoop services in the following order:

- Ranger
- Knox
- ZooKeeper
- HDFS
- YARN
- HBase
- Hive Metastore
- HiveServer2
- WebHCat
- Oozie
- Hue
- Storm
- Kafka
- Atlas

Instructions

1. Start Ranger. Execute the following commands on the Ranger host machine:

```
sudo service ranger-admin start
sudo service ranger-usersync start
```

2. Start Knox. When starting the gateway with the script below, the process runs in the background. The log output is written to `/var/log/knox` and a PID (process ID) is written to `/var/run/knox`. Execute this command on the Knox host machine.

```
su -l knox -c "/usr/hdp/current/knox-server/bin/gateway.sh start"
```



Note

If Knox has been stopped without using `gateway.sh stop`, you must start the service using `gateway.sh clean`. The clean option removes all log files in `/var/log/knox`.

3. Start ZooKeeper. Execute this command on the ZooKeeper host machine(s):

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

4. Start HDFS

- If you are running NameNode HA (High Availability), start the JournalNodes by executing these commands on the JournalNode host machines:

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-journalnode/./hadoop/sbin/hadoop-daemon.sh start journalnode"
```

where `$HDFS_USER` is the HDFS user. For example, `hdfs`.

- Execute this command on the NameNode host machine(s):

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-namenode/./hadoop/sbin/hadoop-daemon.sh start namenode"
```

- If you are running NameNode HA, start the ZooKeeper Failover Controller (ZKFC) by executing the following command on all NameNode machines. The starting sequence of the ZKFCs determines which NameNode will become Active.

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-namenode/./hadoop/sbin/hadoop-daemon.sh start zkfc"
```

- If you are not running NameNode HA, execute the following command on the Secondary NameNode host machine. If you are running NameNode HA, the Standby NameNode takes on the role of the Secondary NameNode.

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-namenode/./hadoop/sbin/hadoop-daemon.sh start secondarynamenode"
```

- Execute these commands on all DataNodes:

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-datanode/./hadoop/sbin/hadoop-daemon.sh start datanode"
```

5. Start YARN

- Execute this command on the ResourceManager host machine(s):

```
su -l yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.sh start resourcemanager"
```

- Execute this command on the History Server host machine:

```
su -l mapred -c "/usr/hdp/current/hadoop-mapreduce-historyserver/sbin/mr-jobhistory-daemon.sh start historyserver"
```

- Execute this command on the timeline server:

```
su -l yarn -c "/usr/hdp/current/hadoop-yarn-timelineserver/sbin/yarn-daemon.sh start timelineserver"
```

- Execute this command on all NodeManagers:

```
su -l yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh start nodemanager"
```

6. Start HBase

- Execute this command on the HBase Master host machine:

```
su -l hbase -c "/usr/hdp/current/hbase-master/bin/hbase-daemon.sh start master; sleep 25"
```

- Execute this command on all RegionServers:

```
su -l hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-daemon.sh start regionserver"
```

7. Start the Hive Metastore. On the Hive Metastore host machine, execute the following commands:

```
su $HIVE_USER  
nohup /usr/hdp/current/hive-metastore/bin/hive --service metastore>/var/log/hive/hive.out 2>/var/log/hive/hive.log &
```

Where `$HIVE_USER` is the Hive user. For example, `hive`.

8. Start HiveServer2. On the Hive Server2 host machine, execute the following commands:

```
su $HIVE_USER  
nohup /usr/hdp/current/hive-server2/bin/hiveserver2 -hiveconf hive.metastore.uris=/tmp/hiveserver2HD.out 2 /tmp/hiveserver2HD.log
```

Where `$HIVE_USER` is the Hive user. For example, `hive`.

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

```
su -l hcat -c "/usr/hdp/current/hive-webhcat/sbin/webhcat_server.sh start"
```

10 Start Oozie. Execute the following command on the Oozie host machine:

```
su -l oozie -c "/usr/hdp/current/oozie-server/bin/oozied.sh start"
```

11 As a root user, execute the following command on the Hue Server:

```
/etc/init.d/hue start
```

This command starts several subprocesses corresponding to the different Hue components. Even though the root user is the one calls the `init.d` script, the actual process runs with the Hue user.

12 Start Storm services using a process controller, such as `supervisord`. See "Installing and Configuring Apache Storm" in the [Non-Ambari Cluster Installation Guide](#). For example, to start the `storm-nimbus` service:

```
sudo /usr/bin/supervisorctl
storm-drpc RUNNING pid 9801, uptime 0:05:05
storm-nimbus STOPPED Dec 01 06:18 PM
storm-ui RUNNING pid 9800, uptime 0:05:05
supervisor> start storm-nimbus
storm-nimbus: started
```

where `$STORM_USER` is the operating system user that installed Storm. For example, `storm`.

13 Start Kafka with the following commands:

```
su $KAFKA_USER
/usr/hdp/current/kafka-broker/bin/kafka start
```

where `$KAFKA_USER` is the operating system user that installed Kafka. For example, `kafka`.

14 Start the Atlas server with the following commands:

```
/usr/hdp/<hdp-version>/atlas/bin/atlas_start.py -port 21000
```

2.2. Stopping HDP Services

Before performing any upgrades or uninstalling software, stop all of the Hadoop services in the following order:

- Ranger
- Knox
- Oozie
- WebHCat
- HiveServer2
- Hive Metastore
- HBase
- YARN
- HDFS
- ZooKeeper
- Hue
- Storm
- Kafka
- Atlas

Instructions

1. Stop Ranger. Execute the following commands on the Ranger host machine:

```
sudo service ranger-admin stop
sudo service ranger-usersync stop
```

2. Stop Knox. Execute the following command on the Knox host machine.

```
su -l knox -c "/usr/hdp/current/knox-server/bin/gateway.sh stop"
```

3. Stop Oozie. Execute the following command on the Oozie host machine.

```
su -l oozie -c "/usr/hdp/current/oozie-server/bin/oozied.sh stop"
```

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

```
su -l hcat -c "/usr/hdp/current/hive-webhcat/sbin/webhcat_server.sh stop"
```

5. Stop Hive. Execute this command 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
```

6. Stop HBase

- Execute this command on all RegionServers:

```
su -l hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-daemon.sh stop regionserver"
```

- Execute this command on the HBase Master host machine:

```
su -l hbase -c "/usr/hdp/current/hbase-master/bin/hbase-daemon.sh stop master"
```

7. Stop YARN

- Execute this command on all NodeManagers:

```
su -l yarn -c "/usr/hdp/current/hadoop-yarn-nodemanager/sbin/yarn-daemon.sh stop nodemanager"
```

- Execute this command on the History Server host machine:

```
su -l mapred -c "/usr/hdp/current/hadoop-mapreduce-historyserver/sbin/mr-jobhistory-daemon.sh stop historyserver"
```

- Execute this command on the timeline server host machine(s):

```
su -l yarn -c "/usr/hdp/current/hadoop-yarn-timeline-server/sbin/yarn-daemon.sh stop timelineserver"
```

- Execute this command on the ResourceManager host machine(s):

```
su -l yarn -c "/usr/hdp/current/hadoop-yarn-resourcemanager/sbin/yarn-daemon.sh stop resourcemanager"
```

8. Stop HDFS

- Execute this command on all DataNodes:

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-datanode/./hadoop/sbin/hadoop-daemon.sh stop datanode"
```

- If you are not running NameNode HA (High Availability), stop the Secondary NameNode by executing this command on the Secondary NameNode host machine:

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-namenode/./hadoop/sbin/hadoop-daemon.sh stop secondarynamenode"
```

- Execute this command on the NameNode host machine(s):

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-namenode/./hadoop/sbin/hadoop-daemon.sh stop namenode"
```

- If you are running NameNode HA, stop the ZooKeeper Failover Controllers (ZKFC) by executing this command on the NameNode host machines:

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-namenode/./hadoop/sbin/hadoop-daemon.sh stop zkfc"
```

- If you are running NameNode HA, stop the JournalNodes by executing these commands on the JournalNode host machines:

```
su -l hdfs -c "/usr/hdp/current/hadoop-hdfs-journalnode/./hadoop/sbin/hadoop-daemon.sh stop journalnode"
```

where \$HDFS_USER is the HDFS user. For example, hdfs.

9. Stop ZooKeeper. Execute this command on the ZooKeeper host machine(s):

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

10 Stop Hue. Execute the following command:

```
/etc/init.d/hue stop
```

11 Start Storm services using a process controller, such as supervisor. See "Installing and Configuring Apache Storm" in the [Non-Ambari Cluster Installation Guide](#). For example, to stop the storm-nimbus service:

```
sudo /usr/bin/supervisorctl
storm-drpc RUNNING pid 9801, uptime 0:03:20
storm-nimbus RUNNING pid 9802, uptime 0:03:20
storm-ui RUNNING pid 9800, uptime 0:03:20
supervisor> stop storm-nimbus
storm-nimbus: stopped
```

where \$STORM_USER is the operating system user that installed Storm. For example, storm.

12 Stop Kafka. Execute this command on the Kafka host machine(s):

```
su $KAFKA_USER
/usr/hdp/current/kafka-broker/bin/kafka stop
```

where `$KAFKA_USER` is the operating system user that installed Kafka. For example, `kafka`.

13 Stop Atlas. Execute

```
su $ATLAS_USER  
python /usr/hdp/current/atlas-server/bin/atlas_stop.py
```

3. Deploying HDP In Production 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. Because 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.



Note

Many of the descriptions in this section assume you are using RHEL/Centos/Oracle Linux. If you are using SLES, please adjust the commands and directories accordingly.

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.

3.1. Terminology

The table below lists the various terms used throughout this section.

Table 3.1. Terminology

| Item | Description |
|---------------------------|---|
| Yum Package Manager (yum) | A package management tool that fetches and installs software packages and performs automatic dependency resolution. |
| 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. |
| HDP Repository Tarball | A tarball image that contains the complete contents of the HDP Repositories. |

3.2. Mirroring or Proxying

HDP uses yum or zypper to install software, and this software is obtained from the HDP Repositories. If your firewall prevents Internet access, you must mirror or proxy the HDP Repositories in your Data Center.

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 cluster node must be updated, so that the given package names are associated with the local mirror server instead of the remote repository server.

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

- **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.
- **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.



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.

- **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 cluster node, so that when the client attempts to access the repository during installation, the request goes through the local proxy server instead of going directly to the remote repository server.

3.3. Considerations for choosing a Mirror or Proxy solution

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

| Advantages of repository mirroring | Advantages of creating a proxy |
|---|---|
| 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 | 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, |

| Advantages of repository mirroring | Advantages of creating a proxy |
|--|---|
| 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. | 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.

3.4. Recommendations for Deploying HDP

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

In many data centers, using a mirror for the HDP Repositories can be the best deployment strategy. The HDP Repositories are small and easily mirrored, allowing you secure control over the contents of the Hadoop packages accepted for use in your data center.



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. 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.

3.5. 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`

Two different packages 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.

3.6. Detailed Instructions for Creating Mirrors and Proxies

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

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

1. Check Your Prerequisites.

Select a mirror server host with the following characteristics:

- The server OS is Debian 7, CentOS (6,7), RHEL (6,7), Oracle Linux(6,7), SLES (11,12), or Ubuntu (12,14,16), 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 lets all cluster nodes (the servers on which you want to install HDP) access this server

2. Install the Repos.

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

Table 3.2. Hortonworks Yum Repositories

| Cluster OS | HDP Repository Tarballs |
|----------------------------|---|
| Debian 7 | <pre>wget http://public-repo-1.hortonworks.com/HDP/debian7/2.x/updates/2.6.0.3/HDP-2.6.0.3-debian7-deb.tar.gz</pre> <pre>wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/debian7/HDP-UTILS-1.1.0.21-debian7.tar.gz</pre> |
| RHEL/CentOS/Oracle LINUX 6 | <pre>wget http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.6.0.3/HDP-2.6.0.3-centos6-rpm.tar.gz</pre> <pre>wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/centos6/HDP-UTILS-1.1.0.21-centos6.tar.gz</pre> |
| RHEL/CentOS/Oracle LINUX 7 | <pre>wget http://public-repo-1.hortonworks.com/HDP/centos7/2.x/updates/2.6.0.3/HDP-2.6.0.3-centos7-rpm.tar.gz</pre> |

| Cluster OS | HDP Repository Tarballs |
|-----------------|--|
| | <pre>wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/centos7/HDP-UTILS-1.1.0.21-centos7.tar.gz</pre> |
| SLES 11 SP3/SP4 | <pre>wget http://public-repo-1.hortonworks.com/HDP/susel1sp3/2.x/updates/2.6.0.3/HDP-2.6.0.3-susel1sp3-rpm.tar.gz</pre> <pre>wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/susel1sp3/HDP-UTILS-1.1.0.21-susel1sp3.tar.gz</pre> |
| SLES 12 | <pre>wget http://public-repo-1.hortonworks.com/HDP/sles12/2.x/updates/2.6.0.3/HDP-2.6.0.3-sles12-rpm.tar.gz</pre> <pre>wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/sles12/HDP-UTILS-1.1.0.21-sles12.tar.gz</pre> |
| Ubuntu 12 | <pre>wget http://public-repo-1.hortonworks.com/HDP/ubuntu12/2.x/updates/2.6.0.3/HDP-2.6.0.3-ubuntu12-deb.tar.gz</pre> <pre>wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/ubuntu12/HDP-UTILS-1.1.0.21-ubuntu12.tar.gz</pre> |
| Ubuntu 14 | <pre>wget http://public-repo-1.hortonworks.com/HDP/ubuntu14/2.x/updates/2.6.0.3/HDP-2.6.0.3-ubuntu14-deb.tar.gz</pre> <pre>wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/ubuntu14/HDP-UTILS-1.1.0.21-ubuntu14.tar.gz</pre> |
| Ubuntu 16 | <pre>wget http://public-repo-1.hortonworks.com/HDP/ubuntu16/2.x/updates/2.6.0.3/HDP-2.6.0.3-ubuntu16-deb.tar.gz</pre> <pre>wget http://public-repo-1.hortonworks.com.s3.amazonaws.com/HDP-UTILS-1.1.0.21/repos/ubuntu16/HDP-UTILS-1.1.0.21-ubuntu16.tar.gz</pre> |

b. Create an HTTP server.

- On the mirror server, install an HTTP server (such as Apache httpd) using the instructions provided [here](#).
- Activate this web server.
- 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.

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

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

- For example, from a shell window, type:

- **For RHEL/CentOS/Oracle:**

```
mkdir -p /var/www/html/hdp/
```

- **For SLES:**

```
mkdir -p /srv/www/htdocs/rpms
```

- **For Ubuntu:**

```
mkdir -p /var/www/html/hdp/
```

- If you are using a symlink, enable the `followsymlinks` on your web server.

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

e. 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://$yourwebserver/hdp/$os/HDP-2.6.0.3/`.

You should see directory listing for all the HDP components along with the RPMs at: `$os/HDP-2.6.0.3`.



Note

If you are installing a 2.x.0 release, use: `http://$yourwebserver/hdp/$os/2.x/GA`

If you are installing a 2.x.x release, use: `http://$yourwebserver/hdp/$os/2.x/updates`

where

- `$OS` can be `debian7`, `centos6`, `centos7`, `suse11sp3`, `sles12`, `ubuntu12`, `ubuntu14`, or `ubuntu16`. Use the following options table for `$OS` parameter:

Table 3.3. HDP Component Options

| Operating System | | Value |
|------------------|----|---------|
| Debian 7 | 25 | debian7 |

| Operating System | Value |
|------------------|-----------|
| RHEL 6 | centos6 |
| RHEL 7 | centos7 |
| SLES 11 SP3/SP4 | suse11sp3 |
| SLES 12 | sles12 |
| Ubuntu 12 | ubuntu12 |
| Ubuntu 14 | ubuntu14 |
| Ubuntu 16 | ubuntu16 |

f. Configure the yum clients on all the nodes in your cluster.

- Fetch the yum configuration file from your mirror server.

```
http://$yourwebserver /hdp/$os/2.x/updates/2.6.0.3/hdp.repo
```

- Store the `hdp.repo` file to a temporary location.
- Edit the `hdp.repo` file changing the value of the base url property to point to your local repositories based on your cluster OS.

where

- `$yourwebserver` is the FQDN of your local mirror server.
- `$os` can be `centos6`, `centos7`, `suse11sp3`, `sles12`, `ubuntu12`, `ubuntu14`, or `ubuntu16`. Use the following options table for `$os` parameter:

Table 3.4. Yum Client Options

| Operating System | Value |
|------------------|-----------|
| Debian 7 | debian7 |
| RHEL 6 | centos6 |
| RHEL 7 | centos7 |
| SLES 11 SP3/SP4 | suse11sp3 |
| SLES 12 | sles12 |
| Ubuntu 12 | ubuntu12 |
| Ubuntu 14 | ubuntu14 |
| Ubuntu 16 | ubuntu16 |

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

- **For RHEL and CentOS**

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

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

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

3.6.2. Option II - Mirror server has temporary or continuous access to the Internet

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

1. Check Your Prerequisites.

Select a local mirror server host with the following characteristics:

- The server OS is Debian 7, CentOS (6,7), RHEL (6,7), Oracle Linux(6,7), SLES (11,12), or Ubuntu (12,14,16), 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.

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
```

2. Install the Repos.

- 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 3.5. Yum Client Configuration Commands

| Cluster OS | HDP Repository Tarballs |
|------------|---|
| Debian 7 | <code>wget http://public-repo-1.hortonworks.com/HDP/debian7/2.x/</code> |

| Cluster OS | HDP Repository Tarballs |
|----------------------------|---|
| | updates/2.6.0.3/hdp.list -O /etc/apt/sources.list.d/hdp.list |
| RHEL/CentOS/Oracle LINUX 6 | wget http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.6.0.3/hdp.repo -O /etc/yum.repos.d/hdp.repo |
| RHEL/CentOS/Oracle LINUX 7 | wget http://public-repo-1.hortonworks.com/HDP/centos7/2.x/updates/2.6.0.3/hdp.repo -O /etc/zypp/repos.d/hdp.repo |
| SLES 11 SP3/SP4 | wget http://public-repo-1.hortonworks.com/HDP/suse11sp3/2.x/updates/2.6.0.3/hdp.repo -O /etc/zypp/repos.d/hdp.repo |
| SLES 12 | wget http://public-repo-1.hortonworks.com/HDP/sles12/2.x/updates/2.6.0.3/hdp.repo -O /etc/zypp/repos.d/hdp.repo |
| Ubuntu 12 | wget http://public-repo-1.hortonworks.com/HDP/ubuntu12/2.x/updates/2.6.0.3/hdp.list -O /etc/apt/sources.list.d/hdp.list |
| Ubuntu 14 | wget http://public-repo-1.hortonworks.com/HDP/ubuntu14/2.x/updates/2.6.0.3/hdp.list -O /etc/apt/sources.list.d/hdp.list |
| Ubuntu 16 | wget http://public-repo-1.hortonworks.com/HDP/ubuntu16/2.x/updates/2.6.0.3/hdp.list -O /etc/apt/sources.list.d/hdp.list |

- Create an HTTP server.
 - On the mirror server, install an HTTP server (such as Apache httpd using the instructions provided)
 - Activate this web server.
 - 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.

- Optional - If your mirror server uses SLES, modify the `default-server.conf` file to enable the docs root folder listing.

```
sed -e s/Options None/Options Indexes MultiViews/ig /etc/apache2/default-server.conf /tmp/tempfile.tmp
mv /tmp/tempfile.tmp /etc/apache2/default-server.conf
```

- On your mirror server, create a directory for your web server.
 - For example, from a shell window, type:

- **For RHEL/CentOS/Oracle:**

```
mkdir -p /var/www/html/hdp/
```

- **For SLES:**

```
mkdir -p /srv/www/htdocs/rpms
```

- **For Ubuntu and Debian:**

```
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:

- **For RHEL/CentOS/Oracle/Ubuntu:**

```
cd /var/www/html/hdp
```

- **For SLES:**

```
cd /srv/www/htdocs/rpms
```

Then for all hosts, type:

- **HDP Repository**

```
reposync -r HDP reposync -r HDP-2.6.0.3 reposync -r HDP-UTILS-1.1.0.21
```

You should see both an HDP-2.6.0.3 directory and an HDP-UTILS-1.1.0.21 directory, each with several subdirectories.

- Generate appropriate metadata.

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

- **For RHEL/CentOS/Oracle:**

- **HDP Repository:**

```
createrepo /var/www/html/hdp/HDP-2.6.0.3 createrepo /var/www/html/hdp/HDP-UTILS-1.1.0.21
```

- **For SLES:**

- **HDP Repository:**

```
createrepo /srv/www/htdocs/rpms/hdp/HDP
```

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:

- HDP:`http://$yourwebserver/hdp/HDP-2.6.0.3/`
- You should now see directory listing for all the HDP components.
- At this point, you can disable external Internet access for the mirror server, so that the mirror server is again entirely within your data center firewall.
- Depending on your cluster OS, configure the yum clients on all the nodes in your cluster
- Edit the repo files, changing the value of the `baseurl` property to the local mirror URL.
- 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.

```
[HDP-2.x]
name=Hortonworks Data Platform Version - HDP-2.x baseurl=http://
$yourwebserver /HDP/ $os /2.x/GA
gpgcheck=1
gpgkey=http://public-repo-1.hortonworks.com/HDP/$os/RPM-GPG-KEY/RPM-
GPG-KEY-Jenkins
enabled=1
priority=1

[HDP-UTILS-1.1.0.21]
name=Hortonworks Data Platform Utils Version - HDP-UTILS-1.1.0.21
baseurl=http:// $yourwebserver /HDP-UTILS-1.1.0.21/repos/ $os
gpgcheck=1
gpgkey=http://public-repo-1.hortonworks.com/HDP/$os/RPM-GPG-KEY/RPM-
GPG-KEY-Jenkins
enabled=1
priority=1

[HDP-2.6.0.3]
name=Hortonworks Data Platform HDP-2.6.0.3 baseurl=http://
$yourwebserver /HDP/ $os /2.x/updates/2.6.0.3
gpgcheck=1
gpgkey=http://public-repo-1.hortonworks.com/HDP/$os/RPM-GPG-KEY/RPM-
GPG-KEY-Jenkins
enabled=1
priority=1
```

where

- `$yourwebserver` is the FQDN of your local mirror server.
- `$os` can be `centos6`, `centos7`, `suse11sp3`, `sles12`, `ubuntu12`, `ubuntu14`, or `ubuntu16`. Use the following options table for `$os` parameter:

Table 3.6. \$OS Parameter Values

| Operating System | Value |
|------------------|-----------|
| Debian 7 | debian7 |
| RHEL 6 | centos6 |
| RHEL 7 | centos7 |
| SLES 11 SP3/SP4 | suse11sp3 |
| SLES 12 | sles12 |
| Ubuntu 12 | ubuntu12 |
| Ubuntu 14 | ubuntu14 |
| Ubuntu 16 | ubuntu16 |

- Copy the yum/zypper client configuration file to all nodes in your cluster.

- RHEL/CentOS/Oracle Linux:

Use `scp` or `pdsh` to copy the client yum configuration file to `/etc/yum.repos.d/` directory on every node in the cluster.

- For SLES:

On every node, invoke the following command:

- **HDP Repository:**

```
zypper addrepo -r http://$yourwebserver/hdp/HDP/suse11/2.x/updates/2.6.0.3/hdp.repo
```

- For Ubuntu:

On every node, invoke the following command:

- **HDP Repository:**

```
sudo add-apt-repository deb http://$yourwebserver/hdp/HDP/ubuntu12/2.x/hdp.list
```

- **Optional - Ambari Repository:**

```
sudo add-apt-repository deb http://$yourwebserver/hdp/ambari/ubuntu12/1.x/updates/1.7.0/ambari.list
```

- If using Ambari, verify the configuration by deploying Ambari server on one of the cluster nodes.

```
yum install ambari-server
```

- If your cluster runs CentOS, Oracle, or RHEL and if you have multiple repositories configured in your environment, deploy the following plugin on all the nodes in your cluster.

- Install the plugin.

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

```
yum install yum-priorities
```

- For RHEL and CentOS v6.x

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

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

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

3.7. Set up a trusted proxy server

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

1. Check Your Prerequisites.

Select a mirror server host with the following characteristics:

- This server runs on either CentOS/RHEL/Oracle Linux (5.x or 6.x), SLES 11, or Ubuntu 12, 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)

2. Create a caching HTTP Proxy server on the selected host.

- 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.
- Activate this proxy server and configure its cache storage location.
- 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`.

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

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

The following description is taken from the CentOS documentation. 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/query`.)

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

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

- 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.
- 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/
```

## 4. Hadoop Service Accounts

To configure Hadoop service accounts, see the "Create System Users and Groups" section of the "Getting Ready to Install" chapter of the [HDP Installation Guide for Non-Ambari Managed Clusters](#).

## 5. Supported Database Matrix for the Hortonworks Data Platform

See [Hortonworks Support Matrix](#) for information regarding supported databases for the Hortonworks Data Platform (HDP).