Administering HDFS

Date of Publish: 2018-07-12

Hortonworks

http://docs.hortonworks.com

Contents

Cluster Maintenance

You can decommission slave nodes, manually add slave nodes to a cluster, and use DistCp to copy files between clusters.

Decommissioning slave nodes

Hadoop provides the decommission feature to retire a set of existing slave nodes (DataNodes, NodeManagers, or HBase RegionServers) in order to prevent data loss.

Slaves nodes are frequently decommissioned for maintenance. As a Hadoop administrator, you will decommission the slave nodes periodically in order to either reduce the cluster size or to gracefully remove dying nodes.

Prerequisites to decommission slave nodes

Make sure the appropriate properties are defined in the hdfs-site.xml file and yarn-site.xml files.

- Ensure that the following property is defined in your hdfs-site.xml file.

```
<property>
 <name>dfs.hosts.exclude</name>
 <value><HADOOP_CONF_DIR>/dfs.exclude</value>
 <final>true</final>
 </property>
```

  where <HADOOP_CONF_DIR> is the directory for storing the Hadoop configuration files. For example, /etc/hadoop/conf.

- Ensure that the following property is defined in your yarn-site.xml file.

```
<property>
 <name>yarn.resourcemanager.nodes.exclude-path</name>
 <value><HADOOP_CONF_DIR>/yarn.exclude</value>
 <final>true</final>
</property>
```

  where <HADOOP_CONF_DIR> is the directory for storing the Hadoop configuration files. For example, /etc/hadoop/conf.

Decommission DataNodes or NodeManagers

Nodes normally run both a DataNode and a NodeManager, and both are typically commissioned or decommissioned together.

With the replication level set to three, HDFS is resilient to individual DataNodes failures. However, there is a high chance of data loss when you terminate DataNodes without decommissioning them first. Nodes must be decommissioned on a schedule that permits replication of blocks being decommissioned.

On the other hand, if a NodeManager is shut down, the ResourceManager will reschedule the tasks on other nodes in the cluster. However, decommissioning a NodeManager may be required in situations where you want a NodeManager to stop to accepting new tasks, or when the tasks take time to execute but you still want to be agile in your cluster management.

Decommission DataNodes

Edit the configuration files and execute commands on the NameNode host machine.

About this task

Use the following instructions to decommission DataNodes in your cluster:

Procedure

1. On the NameNode host machine, edit the <HADOOP_CONF_DIR>/dfs.exclude file and add the list of DataNodes hostnames (separated by a newline character).

   where <HADOOP_CONF_DIR> is the directory for storing the Hadoop configuration files. For example, /etc/hadoop/conf.
2. Update the NameNode with the new set of excluded DataNodes. On the NameNode host machine, execute the following command:

   ```
   su <HDFS_USER>
   hdfs dfsadmin -refreshNodes
   ```

   where <HDFS_USER> is the user owning the HDFS services. For example, hdfs.
3. Open the NameNode web UI (http://<NameNode_FQDN>:50070) and navigate to the DataNodes page. Check to see whether the state has changed to Decommission In Progress for the DataNodes being decommissioned.
4. When all the DataNodes report their state as Decommissioned (on the DataNodes page, or on the Decommissioned Nodes page at http://<NameNode_FQDN>:8088/cluster/ nodes/decommissioned), all of the blocks have been replicated. You can then shut down the decommissioned nodes.
5. If your cluster utilizes a dfs.include file, remove the decommissioned nodes from the <HADOOP_CONF_DIR>/dfs.include file on the NameNode host machine, then execute the following command:

   ```
   su <HDFS_USER>
   hdfs dfsadmin -refreshNodes
   ```

   Note:

   If no dfs.include file is specified, all DataNodes are considered to be included in the cluster (unless excluded in the dfs.exclude file). The dfs.hosts and dfs.hosts.exclude properties in hdfs-site.xml are used to specify the dfs.include and dfs.exclude files.

Decommission NodeManagers

Edit the configuration files on the ResourceManager host machine and add the details of the new set of NodeManagers.

About this task

Use the following instructions to decommission NodeManagers in your cluster:

Procedure

1. On the ResourceManager host machine, edit the <HADOOP_CONF_DIR>/yarn.exclude file and add the list of NodeManager hostnames (separated by a newline character).

   where <HADOOP_CONF_DIR> is the directory for storing the Hadoop configuration files. For example, /etc/hadoop/conf.
2. If your cluster utilizes a yarn.include file, remove the decommissioned nodes from the <HADOOP_CONF_DIR>/yarn.include file on the ResourceManager host machine.

   Note:

   If no yarn.include file is specified, all NodeManagers are considered to be included in the cluster (unless excluded in the yarn.exclude file). The yarn.resourcemanager.nodes.include-path and yarn.resourcemanager.nodes.exclude-path properties in yarn-site.xml are used to specify the yarn.include and yarn.exclude files.

3. Update the ResourceManager with the new set of NodeManagers. On the ResourceManager host machine, execute the following command:

```
su <YARN_USER>
yarn rmadmin -refreshNodes
```

where <YARN_USER> is the user who owns the YARN services, for example, yarn.

Decommission HBase RegionServers

Run the decommission commands on the RegionServer.

About this task

Use the following instruction to decommission HBase RegionServers in your cluster.

At the RegionServer that you want to decommission, run the following command:

```
su <HBASE_USER>
/usr/hdp/current/hbase-client/bin/hbase-daemon.sh stop
```

where <HBASE_USER> is the user who owns the HBase Services. For example, hbase.

RegionServer closes all the regions, then shuts down.

Manually add slave nodes to an HDP cluster

You can add slave nodes as well as the HBase RegionServer.

In this section:

- Slave node prerequisites
- Add slave nodes
- Add HBase RegionServer

Prerequisites to manually add slave nodes

Make sure the ports are available, the database is deployed, and the correct JDK version is installed on all the nodes in the cluster.

Ensure that the new slave nodes meet the following prerequisites:

- The following operating systems are supported:
  - 64-bit Red Hat Enterprise Linux (RHEL) 5 or 6
  - 64-bit CentOS 5 or 6
  - 64-bit SUSE Linux Enterprise Server (SLES) 11, SP1
- At each of your hosts:
  - yum (RHEL)
  - zypper (SLES)
  - rpm
  - scp
  - curl
  - wget
  - unzip
  - tar
  - pdsh
- Ensure that all of the ports are available.

- To install Hive metastore or to use an external database for Oozie metastore, ensure that you deploy either a MySQL or an Oracle database in your cluster. For instructions, see "Meet Minimum System Requirements" in the Installing HDP Manually guide.
- Your system must have the correct JDK installed on all of the nodes in the cluster. For more information, see "Meet Minimum System Requirements" in the Installing HDP Manually guide.

Add slave nodes

Configure the remote repository, install HDFS, compression libraries, and copy the Hadoop configurations to the new slave nodes.

About this task

Use the following instructions to manually add a slave node:

Procedure

1. On each new slave node, configure the remote repository.
2. On each new slave node, install HDFS.
3. On each new slave node, install compression libraries.
4. On each new slave node, create the DataNode and YARN NodeManager local directories.
5. Copy the Hadoop configurations to the new slave nodes and set appropriate permissions.
    - Option I: Copy Hadoop config files from an existing slave node.
        - On an existing slave node, make a copy of the current configurations:

          ```
          tar zcvf hadoop_conf.tgz /etc/hadoop/conf
          ```

        - Copy this file to each of the new nodes:

          ```
          rm -rf /etc/hadoop/conf
          cd /
          tar zxvf $location_of_copied_conf_tar_file/hadoop_conf.tgz
          chmod -R 755 /etc/hadoop/confa
          ```

6. On each of the new slave nodes, start the NodeManager:

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

7. Optional - If you use a HDFS or YARN/ResourceManager .include file in your cluster, add the new slave nodes to the .include file, then run the applicable refreshNodes command.
    - To add new DataNodes to the dfs.include file:
        - On the NameNode host machine, edit the /etc/hadoop/conf/dfs.include file and add the list of the new slave node host names (separated by newline character).

          Note:

          If no dfs.include file is specified, all DataNodes are considered to be included in the cluster (unless excluded in the dfs.exclude file). The dfs.hosts and dfs.hosts.exlude properties in hdfs-site.xml are used to specify the dfs.include and dfs.exclude files.
        - On the NameNode host machine, execute the following command:

          ```
          su -l hdfs -c "hdfs dfsadmin -refreshNodes"
          ```

    - To add new NodeManagers to the yarn.include file:
        - On the ResourceManager host machine, edit the /etc/hadoop/conf/yarn.include file and add the list of the slave node host names (separated by newline character).

Note:

If no yarn.include file is specified, all NodeManagers are considered to be included in the cluster (unless excluded in the yarn.exclude file). The yarn.resourcemanager.nodes.include-path and yarn.resourcemanager.nodes.exclude-path properties in yarn-site.xml are used to specify the yarn.include and yarn.exclude files.

- On the ResourceManager host machine, execute the following command:

```
su -l yarn -c "yarn rmadmin -refreshNodes"
```

Add HBase RegionServer

Install HBase, ZooKeeper, add the HDP repository, and copy the HBase configurations to the new slave nodes.

About this task

Use the following instructions to manually add HBase RegionServer hosts:

Procedure

1. On each of the new slave nodes, install HBase and ZooKeeper.

    - For RHEL/CentOS/Oracle Linux:

    ```
    yum install zookeeper hbase
    ```

    - For SLES:

    ```
    zypper install zookeeper hbase
    ```

2. On each of the new slave nodes, add the HDP repository to yum:

    - RHEL/CentOS/Oracle Linux 6.x:

    ```
    wget -nv http://public-repo-1.hortonworks.com/HDP/centos6/2.x/
    updates/2.3.0.0/hdp.repo -O /etc/yum.repos.d/hdp.repo
    ```

    - RHEL/CentOS/Oracle Linux 7.x:

    ```
    wget -nv http://public-repo-1.hortonworks.com/HDP/centos7/2.x/
    updates/2.3.0.0/hdp.repo -O /etc/yum.repos.d/hdp.repo
    ```

    - SLES SP3/SP4:

    ```
    wget -nv http://public-repo-1.hortonworks.com/HDP/suse11sp3/2.x/
    updates/2.3.0.0/hdp.repo -O /etc/zypp/repos.d/hdp.repo
    ```

3. Copy the HBase configurations to the new slave nodes and set appropriate permissions.

    - Option I: Copy HBase config files from an existing slave node.

        - On any existing slave node, make a copy of the current configurations:

        ```
        tar zcvf hbase_conf.tgz /etc/hbase/conf
        tar zcvf zookeeper_conf.tgz /etc/zookeeper/conf
        ```

        - Copy these files to each of the new nodes:

        ```
        rm -rf /etc/hbase/conf
        mkdir -p /etc/hbase/conf
        cd /
        tar zxvf $location_of_copied_conf_tar_file/hbase_conf.tgz
        ```

```
chmod -R 755 /etc/hbase/conf
```

```
rm -rf /etc/zookeeper/conf
mkdir -p /etc/zookeeper/conf
cd /
tar zxvf $location_of_copied_conf_tar_file/zookeeper_conf.tgz
chmod -R 755 /etc/zookeeper/conf
```

- Option II: Manually add Hadoop configuration files.

4. On all of the new slave nodes, create the configuration directory, copy all of the configuration files, and set the permissions:

```
rm -r $HBASE_CONF_DIR ;
mkdir -p $HBASE_CONF_DIR ;
```

Copy all of the configuration files to $HBASE_CONF_DIR

```
chmod a+x $HBASE_CONF_DIR/;
chown -R $HBASE_USER:$HADOOP_GROUP $HBASE_CONF_DIR/../ ;
chmod -R 755 $HBASE_CONF_DIR/../
```

```
rm -r $ZOOKEEPER_CONF_DIR ;
mkdir -p $ZOOKEEPER_CONF_DIR ;
```

Copy all of the configuration files to $ZOOKEEPER_CONF_DIR

```
chmod a+x $ZOOKEEPER_CONF_DIR/;
chown -R $ZOOKEEPER_USER:$HADOOP_GROUP $ZOOKEEPER_CONF_DIR/../ ;
chmod -R 755 $ZOOKEEPER_CONF_DIR/../
```

where:

- $HBASE_CONF_DIR is the directory to store the HBase configuration files. For example, /etc/hbase/conf.
- $HBASE_USER is the user owning the HBase services. For example, hbase.
- $HADOOP_GROUP is a common group shared by services. For example, hadoop.
- $ZOOKEEPER_CONF_DIR is the directory to store the ZooKeeper configuration files. For example, /etc/zookeeper/conf
- $ZOOKEEPER_USER is the user owning the ZooKeeper services. For example, zookeeper.

5. Start HBase RegionServer node:

```
<login as $HBASE_USER>
/usr/lib/hbase/bin/hbase-daemon.sh --config $HBASE_CONF_DIR start
 regionserver
```

6. On the HBase Master host machine, edit the /usr/lib/hbase/conf file and add the list of slave nodes' hostnames. The hostnames must be separated by a newline character.

Using DistCp to Copy Files

Hadoop DistCp (distributed copy) can be used to copy data between Hadoop clusters (and also within a Hadoop cluster).

DistCp uses MapReduce to implement its distribution, error handling, and reporting. It expands a list of files and directories into map tasks, each of which copies a partition of the files specified in the source list.

Using DistCp

Use DistCp to copy files between various clusters.

The most common use of DistCp is an inter-cluster copy:

```
hadoop distcp hdfs://nn1:8020/source hdfs://nn2:8020/destination
```

Where hdfs://nn1:8020/source is the data source, and hdfs://nn2:8020/ destination is the destination. This will expand the name space under /source on NameNode "nn1" into a temporary file, partition its contents among a set of map tasks, and start copying from "nn1" to "nn2". Note that DistCp requires absolute paths.

You can also specify multiple source directories:

```
hadoop distcp hdfs://nn1:8020/source/a hdfs://nn1:8020/source/b hdfs://
 nn2:8020/destination
```

Or specify multiple source directories from a file with the -f option:

```
hadoop distcp -f hdfs://nn1:8020/srclist hdfs://nn2:8020/destination
```

Where srclist contains:

```
hdfs://nn1:8020/source/a
hdfs://nn1:8020/source/b
```

DistCp from HDP-1.3.x to HDP-2.x

When using DistCp to copy from a HDP-1.3.x cluster to a HDP-2.x cluster, the format is:

```
hadoop distcp hftp://<hdp 1.3.x namenode host>:50070/<folder path of source>
 hdfs://<hdp 2.x namenode host>/<folder path of target>
```

Here is an example of a DistCp copy from HDP 1.3.0 to HDP-2.0:

```
hadoop distcp hftp://namenodehdp130.test.com:50070/apps/hive/warehouse/db/
 hdfs://namenodehdp20.test.com/data/raw/
```

When copying from multiple sources, DistCp will abort the copy with an error message if two sources collide, but collisions at the destination are resolved based on the options specified. By default, files already existing at the destination are skipped (i.e. not replaced by the source file). A count of skipped files is reported at the end of each job, but it may be inaccurate if a copier failed for some subset of its files, but succeeded on a later attempt.

It is important that each NodeManager is able to communicate with both the source and destination file systems. For HDFS, both the source and destination must be running the same version of the protocol, or use a backwards-compatible protocol; see "Copying Between Versions".

After a copy, you should generate and cross-check a listing of the source and destination to verify that the copy was truly successful. Since DistCp employs both Map/Reduce and the FileSystem API, issues in or between any of these three could adversely and silently affect the copy. Some have had success running with -update enabled to perform a second pass, but users should be acquainted with its semantics before attempting this.

It is also worth noting that if another client is still writing to a source file, the copy will likely fail. Attempting to overwrite a file being written at the destination should also fail on HDFS. If a source file is (re)moved before it is copied, the copy will fail with a FileNotFound exception.

Command Line Options

You can use command line operations to perform DistCp copying.

For a description of DistCp command line options, see DistCp Command Line Options.

Related Information

Command Line Options

Update and Overwrite

Use the -update option to copy files from a source when they do not exist at the target. Use the -overwrite function to overwrite the target files even if the content is the same.

The DistCp -update option is used to copy files from a source that does not exist at the target, or that has different contents. The DistCp -overwrite option overwrites target files even if they exist at the source, or if they have the same contents.

The -update and -overwrite options warrant further discussion, since their handling of source-paths varies from the defaults in a very subtle manner.

Consider a copy from /source/first/ and /source/second/ to /target/, where the source paths have the following contents:

```
hdfs://nn1:8020/source/first/1
hdfs://nn1:8020/source/first/2
hdfs://nn1:8020/source/second/10
hdfs://nn1:8020/source/second/20
```

When DistCp is invoked without -update or -overwrite, the DistCp defaults would create directories first/ and second/, under /target. Thus:

```
distcp hdfs://nn1:8020/source/first hdfs://nn1:8020/source/second hdfs://
nn2:8020/target
```

would yield the following contents in /target:

```
hdfs://nn2:8020/target/first/1
hdfs://nn2:8020/target/first/2
hdfs://nn2:8020/target/second/10
hdfs://nn2:8020/target/second/20
```

When either -update or -overwrite is specified, the contents of the source directories are copied to the target, and not the source directories themselves. Thus:

```
distcp -update hdfs://nn1:8020/source/first hdfs://nn1:8020/source/second
 hdfs://nn2:8020/target
```

would yield the following contents in /target:

```
hdfs://nn2:8020/target/1
hdfs://nn2:8020/target/2
hdfs://nn2:8020/target/10
hdfs://nn2:8020/target/20
```

By extension, if both source folders contained a file with the same name ("0", for example), then both sources would map an entry to /target/0 at the destination. Rather than permit this conflict, DistCp will abort.

Now, consider the following copy operation:

```
distcp hdfs://nn1:8020/source/first hdfs://nn1:8020/source/second hdfs://
nn2:8020/target
```

With sources/sizes:

```
hdfs://nn1:8020/source/first/1 32
hdfs://nn1:8020/source/first/2 32
```

```
hdfs://nn1:8020/source/second/10 64
hdfs://nn1:8020/source/second/20 32
```

And destination/sizes:

```
hdfs://nn2:8020/target/1 32
hdfs://nn2:8020/target/10 32
hdfs://nn2:8020/target/20 64
```

Will effect:

```
hdfs://nn2:8020/target/1 32
hdfs://nn2:8020/target/2 32
hdfs://nn2:8020/target/10 64
hdfs://nn2:8020/target/20 32
```

1 is skipped because the file-length and contents match. 2 is copied because it does not exist at the target. 10 and 20 are overwritten because the contents don't match the source.

If the -update option is used, 1 is overwritten as well.

DistCp and Security Settings

Security settings dictate whether DistCp should be run on the source cluster or the destination cluster.

The general rule-of-thumb is that if one cluster is secure and the other is not secure, DistCp should be run from the secure cluster -- otherwise there may be security- related issues.

When copying data from a secure cluster to an non-secure cluster, the following configuration setting is required for the DistCp client:

```
<property>
 <name>ipc.client.fallback-to-simple-auth-allowed</name>
 <value>true</value>
</property>
```

When copying data from a secure cluster to a secure cluster, the following configuration setting is required in the core-site.xml file:

```
<property>
    <name>hadoop.security.auth_to_local</name>
    <value></value>
    <description>Maps kerberos principals to local user names</description>
</property>
```

Secure-to-Secure: Kerberos Principal Name

Assign the same principle name to applicable NameNodes in the source and destination clusters.

distcp hdfs://hdp-2.0-secure hdfs://hdp-2.0-secure One issue here is that the SASL RPC client requires that the remote server's Kerberos principal must match the server principal in its own configuration. Therefore, the same principal name must be assigned to the applicable NameNodes in the source and the destination cluster. For example, if the Kerberos principal name of the NameNode in the source cluster is nn/host1@realm, the Kerberos principal name of the NameNode in destination cluster must be nn/host2@realm, rather than nn2/host2@realm, for example.

Secure-to-Secure: ResourceManager mapping rules

When copying between two HDP2 secure clusters, or from HDP1 secure to HDP2 secure, further ResourceManager (RM) configuration is required if the two clusters have different realms.

In order for DistCP to succeed, the same RM mapping rule must be used in both clusters.

For example, if secure Cluster 1 has the following RM mapping rule:

```
<property>
    <name>hadoop.security.auth_to_local</name>
    <value>
      RULE:[2:$1@$0](rm@.*SEC1.SUP1.COM)s/.*/yarn/
      DEFAULT
    </value>
</property>
```

And secure Cluster 2 has the following RM mapping rule:

```
<property>
    <name>hadoop.security.auth_to_local</name>
    <value>
      RULE:[2:$1@$0](rm@.*BA.YISEC3.COM)s/.*/yarn/
      DEFAULT
    </value>
</property>
```

The DistCp job from Cluster 1 to Cluster 2 will fail because Cluster 2 cannot resolve the RM principle of Cluster 1 correctly to the yarn user, because the RM mapping rule in Cluster 2 is different than the RM mapping rule in Cluster 1.

The solution is to use the same RM mapping rule in both Cluster 1 and Cluster 2:

```
<property>
    <name>hadoop.security.auth_to_local</name>
    <value>
      RULE:[2:$1@$0](rm@.*SEC1.SUP1.COM)s/.*/yarn/
      RULE:[2:$1@$0](rm@.*BA.YISEC3.COM)s/.*/yarn/
      DEFAULT
   </value>
 </property>
```

DistCp between HA clusters

To copy data between HA clusters, use the dfs.internal.nameservices property in the hdfs-site.xml file to explicitly specify the name services belonging to the local cluster, while continuing to use the dfs.nameservices property to specify all of the name services in the local and remote clusters.

About this task

Use the following steps to copy data between HA clusters:

Modify the following properties in the hdfs-site.xml file for both cluster A and cluster B:

Procedure

1. Add both name services to dfs.nameservices = HAA, HAB
2. Add the dfs.internal.nameservices property:
    - In cluster A:

      dfs.internal.nameservices = HAA
    - In cluster B:

      dfs.internal.nameservices = HAB
3. Add dfs.ha.namenodes.<nameservice> to both clusters:
    - In cluster A

dfs.ha.namenodes.HAB = nn1,nn2

- In cluster B

  dfs.ha.namenodes.HAA = nn1,nn2

4. Add the dfs.namenode.rpc-address.<cluster>.<nn> property:
   - In Cluster A:

     dfs.namenode.rpc-address.HAB.nn1 = <NN1_fqdn>:8020

     dfs.namenode.rpc-address.HAB.nn2 = <NN2_fqdn>:8020
   - In Cluster B:

     dfs.namenode.rpc-address.HAA.nn1 = <NN1_fqdn>:8020

     dfs.namenode.rpc-address.HAA.nn2 = <NN2_fqdn>:8020
5. Add the following properties to enable distcp over WebHDFS and secure WebHDFS:
   - In Cluster A:

     dfs.namenode.http-address.HAB.nn1 = <NN1_fqdn>:50070

     dfs.namenode.http-address.HAB.nn2 = <NN2_fqdn>:50070

     dfs.namenode.https-address.HAB.nn1 = <NN1_fqdn>:50470

     dfs.namenode.https-address.HAB.nn2 = <NN2_fqdn>:50470
   - In Cluster B:

     dfs.namenode.http-address.HAA.nn1 = <NN1_fqdn>:50070

     dfs.namenode.http-address.HAA.nn2 = <NN2_fqdn>:50070

     dfs.namenode.https-address.HAA.nn1 = <NN1_fqdn>:50470

     dfs.namenode.https-address.HAA.nn2 = <NN2_fqdn>:50470
6. Add the dfs.client.failover.proxy.provider.<cluster> property:
   - In cluster A:

     dfs.client.failover.proxy.provider. HAB = org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider
   - In cluster B:

     dfs.client.failover.proxy.provider. HAA = org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider
7. Restart the HDFS service, then run the distcp command using the NameService. For example:

   hadoop distcp hdfs://HAA/tmp/testDistcp hdfs://HAB/tmp/

DistCp and HDP version

The HDP version of the source and destination clusters can determine which type of file systems should be used to read the source cluster and write to the destination cluster.

For example, when copying data from a 1.x cluster to a 2.x cluster, it is impossible to use “hdfs” for both the source and the destination, because HDP 1.x and 2.x have different RPC versions, and the client cannot understand both at the same time. In this case the WebHdfsFilesystem (webhdfs://) can be used in both the source and destination clusters, or the HftpFilesystem (hftp://) can be used to read data from the source cluster.

DistCp data copy matrix: HDP1/HDP2 to HDP2

To copy data from HDP1 and HDP2 clusters to HDP2 clusters using DistCp, you must configure and make changes to the settings of the source and destination clusters.

The following table provides a summary of configuration, settings and results when using DistCp to copy data from HDP1 and HDP2 clusters to HDP2 clusters.

From	To	Source Configuration	Destination Configuration	DistCp Should be Run on...	Result
HDP 1.3	HDP 2.x	insecure + hdfs	insecure + webhdfs	HDP 1.3 (source)	success
HDP 1.3	HDP 2.x	secure + hdfs	secure + webhdfs	HDP 1.3 (source)	success
HDP 1.3	HDP 2.x	secure + hftp	secure + hdfs	HDP 2.x (destination)	success
HDP 1.3	HDP 2.1	secure + hftp	secure + swebhdfs	HDP 2.1 (destination)	success
HDP 1.3	HDP 2.x	secure + hdfs	insecure + webhdfs	HDP 1.3 (source)	Possible issues
HDP 2.x	HDP 2.x	secure + hdfs	insecure + hdfs	secure HDP 2.x (source)	success
HDP 2.x	HDP 2.x	secure + hdfs	secure + hdfs	either HDP 2.x (source or destination)	success
HDP 2.x	HDP 2.x	secure + hdfs	secure + webhdfs	HDP 2.x (source)	success
HDP 2.x	HDP 2.x	secure + hftp	secure + hdfs	HDP 2.x (destination)	success

For the above table:

- The term "secure" means that Kerberos security is set up.
- HDP 2.x means HDP 2.0 or later.
- hsftp is available in both HDP-1.x and HDP-2.x. It adds https support to hftp.

Related Information

Possible 1.3 to 2.x Copy Issues

Copying Data from HDP-2.x to HDP-1.x Clusters

Copying Data from HDP-1.x to HDP-2.x Clusters is also supported, however, HDP-1.x is not aware of a new checksum introduced in HDP-2.x.

To copy data from HDP-2.x to HDP-1.x: Skip the checksum check during source 2.x --> 1.x.

-or-

Ensure that the file to be copied is in CRC32 before distcp 2.x --> 1.x.

DistCp Architecture

DistCp comprises of the DistCp driver, copy-listing generator, and InputFormats and MapReduce components.

DistCp Driver

The DistCp driver parses the arguments passed to the DistCp command on the command line.

The DistCp Driver components are responsible for:

- Parsing the arguments passed to the DistCp command on the command-line, via:
- OptionsParser
- DistCpOptionsSwitch

Assembling the command arguments into an appropriate DistCpOptions object, and initializing DistCp. These arguments include:

- Source-paths

- Target location
- Copy options (e.g. whether to update-copy, overwrite, which file attributes to preserve, etc.)

Orchestrating the copy operation by:

- Invoking the copy-listing generator to create the list of files to be copied.
- Setting up and launching the Hadoop MapReduce job to carry out the copy.
- Based on the options, either returning a handle to the Hadoop MapReduce job immediately, or waiting until completion.

The parser elements are executed only from the command-line (or if DistCp::run() is invoked). The DistCp class may also be used programmatically, by constructing the DistCpOptions object and initializing a DistCp object appropriately.

Copy-listing Generator

The copy-listing generator classes are responsible for creating the list of files/directories to be copied from source.

They examine the contents of the source paths (files/directories, including wildcards), and record all paths that need copying into a SequenceFile for consumption by the DistCp Hadoop Job. The main classes in this module include:

- CopyListing: The interface that should be implemented by any copy-listing generator implementation. Also provides the factory method by which the concrete CopyListing implementation is chosen.
- SimpleCopyListing: An implementation of CopyListing that accepts multiple source paths (files/directories), and recursively lists all of the individual files and directories under each for copy.
- GlobbedCopyListing: Another implementation of CopyListing that expands wildcards in the source paths.
- FileBasedCopyListing: An implementation of CopyListing that reads the source path list from a specified file.

Based on whether a source file list is specified in the DistCpOptions, the source listing is generated in one of the following ways:

- If there is no source file list, the GlobbedCopyListing is used. All wildcards are expanded, and all of the expansions are forwarded to the SimpleCopyListing, which in turn constructs the listing (via recursive descent of each path).
- If a source file list is specified, the FileBasedCopyListing is used. Source paths are read from the specified file, and then forwarded to the GlobbedCopyListing. The listing is then constructed as described above.

You can customize the method by which the copy-listing is constructed by providing a custom implementation of the CopyListing interface. The behaviour of DistCp differs here from the legacy DistCp, in how paths are considered for copy.

The legacy implementation only lists those paths that must definitely be copied on to the target. For example, if a file already exists at the target (and -overwrite is not specified), the file is not even considered in the MapReduce copy job. Determining this during setup (before the MapReduce Job) involves file size and checksum comparisons that are potentially time consuming.

DistCp postpones such checks until the MapReduce job, thus reducing setup time. Performance is enhanced further since these checks are parallelized across multiple maps.

InputFormats and MapReduce Components

The InputFormats and MapReduce components are responsible for the actual copying of files and directories from the source to the destination path.

The listing file created during copy-listing generation is consumed at this point, when the copy is carried out. The classes of interest here include:

- UniformSizeInputFormat: This implementation of org.apache.hadoop.mapreduce.InputFormat provides equivalence with Legacy DistCp in balancing load across maps. The aim of the UniformSizeInputFormat is to make each map copy roughly the same number of bytes. Therefore, the listing file is split into groups of paths, such that the sum of file sizes in each InputSplit is nearly equal to every other map. The splitting is not always perfect, but its trivial implementation keeps the setup time low.

- DynamicInputFormat and DynamicRecordReader: The DynamicInputFormat implements org.apache.hadoop.mapreduce.InputFormat, and is new to DistCp. The listing file is split into several "chunk files", the exact number of chunk files being a multiple of the number of maps requested for in the Hadoop Job. Each map task is "assigned" one of the chunk files (by renaming the chunk to the task's id), before the Job is launched. Paths are read from each chunk using the DynamicRecordReader, and processed in the CopyMapper. After all of the paths in a chunk are processed, the current chunk is deleted and a new chunk is acquired. The process continues until no more chunks are available. This "dynamic" approach allows faster map tasks to consume more paths than slower ones, thus speeding up the DistCp job overall.
- CopyMapper: This class implements the physical file copy. The input paths are checked against the input options (specified in the job configuration), to determine whether a file needs to be copied. A file will be copied only if at least one of the following is true:
  - A file with the same name does not exist at target.
  - A file with the same name exists at target, but has a different file size.
  - A file with the same name exists at target, but has a different checksum, and -skipcrccheck is not mentioned.
  - A file with the same name exists at target, but -overwrite is specified.
  - A file with the same name exists at target, but differs in block-size (and block-size needs to be preserved).
- CopyCommitter: This class is responsible for the commit phase of the DistCp job, including:
  - Preservation of directory permissions (if specified in the options)
  - Clean up of temporary files, work directories, etc.

DistCp Frequently Asked Questions

There are differences between DistCp latest version and the legacy DistCp versions.

- Why does -update not create the parent source directory under a pre-existing target directory? The behavior of -update and -overwrite is described in detail in the Using DistCp section of this document. In short, if either option is used with a pre-existing destination directory, the contents of each source directory are copied over, rather than the source directory itself. This behavior is consistent with the legacy DistCp implementation.
- How does the new DistCp (version 2) differ in semantics from the legacy DistCp?
  - Files that are skipped during copy previously also had their file-attributes (permissions, owner/group info, etc.) unchanged, when copied with Legacy DistCp. These are now updated, even if the file copy is skipped.
  - In Legacy DistCp, empty root directories among the source path inputs were not created at the target. These are now created.
- Why does the new DistCp (version 2) use more maps than legacy DistCp? Legacy DistCp works by figuring out what files need to be actually copied to target before the copy job is launched, and then launching as many maps as required for copy. So if a majority of the files need to be skipped (because they already exist, for example), fewer maps will be needed. As a consequence, the time spent in setup (i.e. before the MapReduce job) is higher. The new DistCp calculates only the contents of the source paths. It does not try to filter out what files can be skipped. That decision is put off until the MapReduce job runs. This is much faster (vis-a-vis execution-time), but the number of maps launched will be as specified in the -m option, or 20 (the default) if unspecified.
- Why does DistCp not run faster when more maps are specified? At present, the smallest unit of work for DistCp is a file. i.e., a file is processed by only one map. Increasing the number of maps to a value exceeding the number of files would yield no performance benefit. The number of maps launched would equal the number of files.
- Why does DistCp run out of memory? If the number of individual files/directories being copied from the source path(s) is extremely large (e.g. 1,000,000 paths), DistCp might run out of memory while determining the list of paths for copy. This is not unique to the new DistCp implementation. To get around this, consider changing the -Xmx JVM heap- size parameters, as follows:

  ```
  bash$ export HADOOP_CLIENT_OPTS="-Xms64m -Xmx1024m"
   bash$ hadoop distcp /source /target
  ```

DistCp additional considerations

DistCp also provides a strategy to “dynamically” size maps, allowing faster DataNodes to copy more bytes than slower nodes.

Map Sizing

By default, DistCp makes an attempt to size each map comparably so that each copies roughly the same number of bytes. Note that files are the finest level of granularity, so increasing the number of simultaneous copiers (i.e. maps) may not always increase the number of simultaneous copies nor the overall throughput.

Using the dynamic strategy (explained in the Architecture), rather than assigning a fixed set of source files to each map task, files are instead split into several sets. The number of sets exceeds the number of maps, usually by a factor of 2-3. Each map picks up and c opies all files listed in a chunk. When a chunk is exhausted, a new chunk is acquired and processed, until no more chunks remain.

By not assigning a source path to a fixed map, faster map tasks (i.e. DataNodes) are able to consume more chunks -- and thus copy more data -- than slower nodes. While this distribution is not uniform, it is fair with regard to each mapper’s capacity.

The dynamic strategy is implemented by the DynamicInputFormat. It provides superior performance under most conditions.

Tuning the number of maps to the size of the source and destination clusters, the size of the copy, and the available bandwidth is recommended for long-running and regularly run jobs.

Copying Between Versions of HDFS

For copying between two different versions of Hadoop, you will usually use HftpFileSystem. This is a read-only FileSystem, so DistCp must be run on the destination cluster (more specifically, on NodeManagers that can write to the destination cluster). Each source is specified as hftp://<dfs.http.address>/<path> (the default dfs.http.address is <namenode>:50070).

MapReduce and Other Side-Effects

As mentioned previously, should a map fail to copy one of its inputs, there will be several side-effects.

- Unless -overwrite is specified, files successfully copied by a previous map will be marked as “skipped” on a re-execution.
- If a map fails mapreduce.map.maxattempts times, the remaining map tasks will be killed (unless -i is set).
- If mapreduce.map.speculative is set final and true, the result of the copy is undefined.

SSL Configurations for HSFTP Sources

To use an HSFTP source (i.e. using the HSFTP protocol), a SSL configuration file needs to be specified (via the -mapredSslConf option). This must specify 3 parameters:

- ssl.client.truststore.location: The local file system location of the trust-store file, containing the certificate for the NameNode.
- ssl.client.truststore.type: (Optional) The format of the trust-store file.
- ssl.client.truststore.password: (Optional) Password for the trust-store file.

The following is an example of the contents of a SSL Configuration file:

```
<configuration>
 <property>
    <name>ssl.client.truststore.location</name>
    <value>/work/keystore.jks</value>
    <description>Truststore to be used by clients like distcp. Must be
 specified.</description>
 </property>

 <property>
    <name>ssl.client.truststore.password</name>
    <value>changeme</value>
```

```
    <description>Optional. Default value is "".</description>
 </property>

 <property>
    <name>ssl.client.truststore.type</name>
    <value>jks</value>
    <description>Optional. Default value is "jks".</description>
 </property>
</configuration>
```

The SSL configuration file must be in the classpath of the DistCp program.

Ports and Services Reference

Configure ports of various HDP services and control HDP services manually.

Configuring ports

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

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

Accumulo service ports

Note the default ports used by the various Accumulo services.

The following table lists the default ports used by the various Accumulo services. (Note: None of these services are used in a standard HDP installation.)

Table 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)	master.port.client in accumulo-site.xml
TabletServer	Slave nodes	9997		The TabletServer thrift server	Yes (client API needs)	tserver.port.client in accumulo-site.xml
Garbage Collector	GC nodes (Active GC and any standby)	50091		The GarbageCollector thrift server	No	gc.port.client in accumulo-site.xml
Monitor	Monitor nodes (Active Monitor and any standby)	50095	HTTP(S)	Metrics/Monitoring of an Accumulo instance	Yes	monitor.port.client in accumulo-site.xml
Monitor log aggregation	Monitor nodes (Active Monitor and any standby)	4560		Log4j socket which accepts logs forwarded from other Accumulo services	No	monitor.port.log4j in accumulo-site.xml
Tracer	Tracer nodes	12234		The Tracer thrift server	Yes (if enabled)	trace.port.client in accumulo-site.xml

Service	Servers	Default Ports Used	Protocol	Description	Need End User Access?	Configuration Parameters
Thrift Proxy (optional)	Proxy nodes	42424		The Thrift Proxy server	Yes (if enabled)	port in proxy.properties
TabletServer Replication Service	Slave nodes	10002		TabletServer Thrift service supporting multi-instance Accumulo replication	No	replication.receipt.service.port in accumulo-site.xml **Note:** Hive and Accumulo installed on the same host might result in a potential port conflict because hive.server2.webui.port in hive-site.xml also uses port 10002. To avoid this port conflict, consider installing Hive and Accumulo on different hosts of your cluster.
Master Replication Service	Master nodes (Active master and any standby)	10001		Master Thrift service supporting multi-instance Accumulo replication	No	master.replication.coordinator.port in accumulo-site.xml

Atlas service ports

Note the default ports use by Apache Atlas.

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

Table 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	atlas.server.http.port
Atlas	Atlas Admin Nodes	21443	HTTPS	Port for Atlas Admin web UI (with SSL)	Yes	atlas.server.https.port

Flume service ports

Note the default ports used by various Flume services.

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 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)	master.port.client in accumulo-site.xml
Flume	HDFS Sink	8020	TCP	Communication from Flume into the Hadoop cluster's NameNode	Yes (client API needs)	tserver.port.client in accumulo-site.xml
Flume	HDFS Sink	9000	TCP	Communication from Flume into the Hadoop cluster's NameNode	No	gc.port.client in accumulo-site.xml

Service	Servers	Default Ports Used	Protocol	Description	Need End User Access?	Configuration Parameters
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 the Hadoop cluster's HBase Regionserver	No	
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 specificed 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

HBase service ports

Note the default ports used by various HBase services.

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

Table 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

Service	Servers	Default Ports Used	Protocol	Description	Need End User Access?	Configuration Parameters
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 not installed by default	Yes (Typically admins, dev/support teams)	hbase.rest.info.port
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

HDFS service ports

Note the default ports used by various HDFS services.

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 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

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

Hive service ports

Note the default ports used by various Hive services.

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 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

Hue service port

Note the default port used by the Hue web listener.

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

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

Kafka service ports

Note the default port used by Kafka.

The following table lists the default ports used by Kafka.

Table 7: 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.		

Kerberos service ports

Note the default port used by the designated Kerberos KDC.

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

Table 8: 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		

Knox service ports

Note the default port used by Knox.

The following table lists the default port used by Knox.

Table 9: Knox Service Ports

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

MapReduce service ports

Note the default port used by the various MapReduce services.

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

Table 10: 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

Service	Servers	Default Ports Used	Protocol	Description	Need End User Access?	Configuration Parameters
MapReduce		19888	http	MapReduce JobHistory webapp address		mapreduce.jobhistory.webapp.address
MapReduce		13562	http	MapReduce Shuffle Port		mapreduce.shuffle.port
MapReduce		19890	https	MapReduce JobHistory webapp HTTPS address		mapreduce.jobhistory.webapp.https.address

MySQL service ports

Note the default ports used by the various MySQL services.

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

Table 11: MySQL Service Ports

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

Oozie service ports

Note the default ports used by Oozie.

The following table lists the default ports used by Oozie.

Table 12: 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

Ranger service ports

Note the default ports used by Ranger.

The following table lists the default ports used by Ranger.

Table 13: 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

Sqoop service ports

Note the default ports used by Sqoop.

The following table lists the default ports used by Sqoop.

Table 14: 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	

Storm service ports

Note the default ports used by Storm.

The following table lists the default ports used by Storm.

Table 15: 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

Service	Servers	Default Port	Default Ambari Port	Protocol	Description	Need End User Access?	Configuration Parameters
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

Tez ports

Note the default ports used by the various Tez services.

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

Table 16: 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 ClientRMProtocol proxy		tez.ampool.address

YARN service ports

Note the default ports used by the various YARN services.

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

Table 17: 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	yarn.resourcemanager.webapp.address
Resource Manager Hostname	Master Nodes (Resource Manager Node)	8090	https	Resource Manager HTTPS Address	Yes	yarn.resourcemanager.webapp.https.address

Service	Servers	Default Ports Used	Protocol	Description	Need End User Access?	Configuration Parameters
Resource Manager	Master Nodes (Resource Manager Node)	8050	IPC	For application submissions	Yes (All clients who need to submit the YARN applications including Hive, Hive server, Pig)	Embedded in URI specified by yarn.resourcemanager.address
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)	yarn.resourcemanager.resource-tracker.address
Scheduler	Master Nodes (Resource Manager Node)	8030	http	Scheduler Address	Yes (Typically admins, Dev/ Support teams)	yarn.resourcemanager.scheduler.address
Resource Manager	Master Nodes (Resource Manager Node)	8141	http	Scheduler Address	Yes (Typically admins, Dev/ Support teams)	yarn.resourcemanager.admin.address
NodeManager	Master Nodes (NodeManager) and Slave Nodes	45454	http	NodeManager Address	Yes (Typically admins, Dev/ Support teams)	yarn.nodemanager.address
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

Zeppelin service port

Note the default port used by Zeppelin.

The following table lists the default port used by Zeppelin.

Table 18: Zeppelin Service Ports

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

ZooKeeper service ports

Note the default ports used by Zookeeper service.

Table 19: 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 each other.	No	hbase.zookeeper.peerport
ZooKeeper Server	All ZooKeeper Nodes	3888		Port used by ZooKeeper peers to talk to each other.	No	hbase.zookeeper.leaderport
ZooKeeper Server	All ZooKeeper Nodes	2181		Property from ZooKeeper's config zoo.cfg. The port at which the clients connect.	Yes	hbase.zookeeper.property.clientPort

Related Information

Ports used by ZooKeeper Peers

Controlling HDP services manually

You must follow the precise order while starting and stopping the various HDP services.

Starting HDP services

Make sure to start the Hadoop services in the prescribed order.

About this task

- Ranger
- Knox
- ZooKeeper
- HDFS
- YARN
- HBase
- Hive Metastore
- HiveServer2
- WebHCat
- Oozie

- Hue
- Storm
- Kafka
- Atlas

Procedure

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

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

10. 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.

11. Start Storm services using a process controller, such as supervisord. 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.

12. 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.

13. Start the Atlas server with the following commands:

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

Stopping HDP services

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

About this task

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

Procedure

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-timelineserver/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. 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 the following command.

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