

Commissioning and Decommissioning Nodes in a Hadoop Cluster

One of the most attractive features of Hadoop framework is its **utilization of commodity hardware**. However, this leads to frequent Data Node crashes in a Hadoop cluster. Another striking feature of Hadoop Framework is the **ease of scale in accordance to the rapid growth in data volume**. Because of these two reasons, one of the most common task of a Hadoop administrator is to **commission** (Add) and **decommission** (Remove) Data Nodes in a Hadoop Cluster.

You cannot directly remove any datanode in large cluster or a real-time cluster, as it will cause a lot of disturbance. And if you want to take a machine away for hardware up-gradation purpose, or if you want to bring down one or more than one node, decommissioning will required because you cannot suddenly shut down the datanode/slave-nodes. Similarly, if you want to scale your cluster or add new data nodes without shutting down the cluster, you need commissioning.

We can decommission the node by following below steps or using ambari UI.

Steps for Decommissioning of datanode.

1. First make sure that no jobs are running on the datanode which you want to decommission. You can verify this by accessing resource manager GUI.
2. Create the file `$HADOOP_HOME/conf/dfs-exclude.txt` with the following content:
slave1

The `dfs-exclude.txt` file contains the DataNode hostnames, one per line, that are to be decommissioned from the cluster

3. Add the following property to the file `$HADOOP_HOME/conf/hdfs-site.xml`:

```
<property>  
<name>dfs.hosts.exclude</name>  
<value>$HADOOP_HOME/conf/dfs-exclude.txt</value>  
</property>
```

4. Force the NameNode to reload the active DataNodes using the following command:

```
hadoop dfsadmin -refreshNodes
```

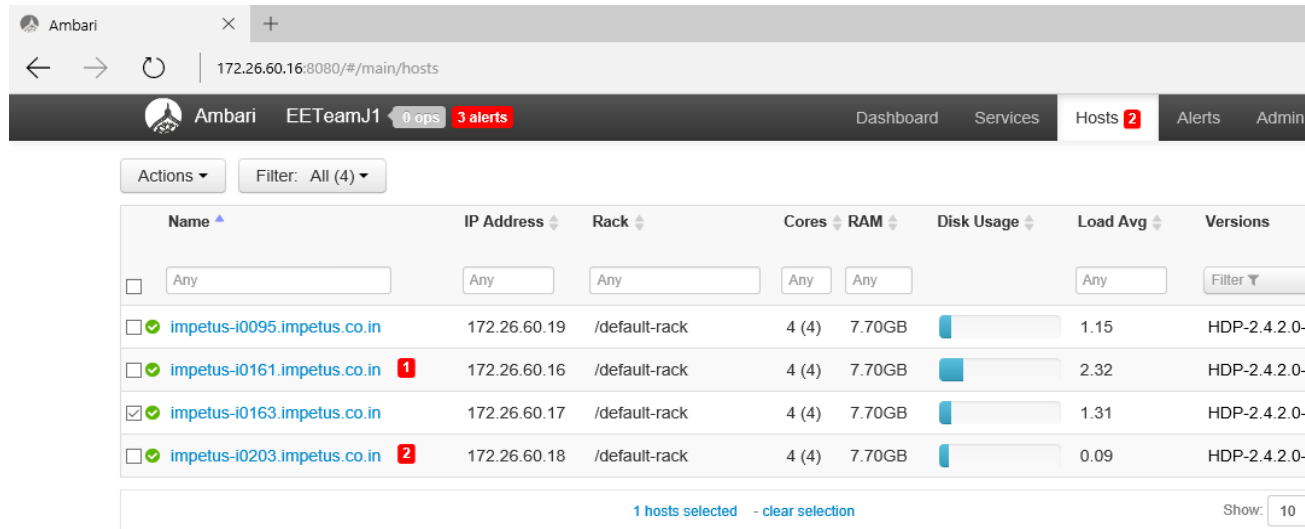
5. Now the decommission process will be started and we can verify the status of data nodes in namenode webUI.

Get a description report of each active DataNode with the following command

```
hadoop dfsadmin -report
```

Steps for Decommissioning of datanode using Ambari UI

1. Logging to ambari gui and go to the click on tab, then select the host on which the datanode needs to be decommissioned:-



Ambari EETeamJ1 0 ops 3 alerts Dashboard Services Hosts 2 Alerts Admin

Actions Filter: All (4)

Name	IP Address	Rack	Cores	RAM	Disk Usage	Load Avg	Versions
<input type="checkbox"/> Any	Any	Any	Any	Any		Any	Filter
<input type="checkbox"/> impetus-i0095.impetus.co.in	172.26.60.19	/default-rack	4 (4)	7.70GB	<div></div>	1.15	HDP-2.4.2.0-
<input type="checkbox"/> impetus-i0161.impetus.co.in 1	172.26.60.16	/default-rack	4 (4)	7.70GB	<div></div>	2.32	HDP-2.4.2.0-
<input checked="" type="checkbox"/> impetus-i0163.impetus.co.in	172.26.60.17	/default-rack	4 (4)	7.70GB	<div></div>	1.31	HDP-2.4.2.0-
<input type="checkbox"/> impetus-i0203.impetus.co.in 2	172.26.60.18	/default-rack	4 (4)	7.70GB	<div></div>	0.09	HDP-2.4.2.0-

1 hosts selected - clear selection Show: 10

Licensed under the [Apache License, Version 2.0](#).
See [third-party tools/resources](#) that Ambari uses and their respective authors



2. Then click on **Action** button on the upper left side of the screen, then on **selected hosts**, **then** on DataNode then decommission

The screenshot shows the Ambari web interface at the URL `172.26.60.16:8080/#/main/hosts`. The top navigation bar includes 'Dashboard', 'Services', 'Hosts' (with a red '2' indicating alerts), 'Alerts', and 'Admin'. The 'Hosts' tab is active, displaying a table of hosts. A context menu is open for the host 'impetus-i0163.impetus.co.in', showing options: 'Start', 'Stop', 'Restart', 'Decommission' (highlighted), and 'Recommission'. The table lists the following hosts:

Host	IP Address	Rack	Cores	RAM	Disk Usage	Load Avg	Versions	Components
impetus-i0161.impetus.co.in	172.26.60.16	Any	4 (4)	7.70GB	1.15	HDP-2.4.2.0-258	21 Components	
impetus-i0163.impetus.co.in	172.26.60.16	Any	4 (4)	7.70GB	2.65	HDP-2.4.2.0-258	22 Components	
impetus-i0163.impetus.co.in	172.26.60.16	Any	4 (4)	7.70GB	4.32	HDP-2.4.2.0-258	14 Components	
impetus-i0203.impetus.co.in	172.26.60.16	Any	4 (4)	7.70GB	2.31	HDP-2.4.2.0-258	18 Components	

At the bottom of the page, there is a license notice: 'Licensed under the Apache License, Version 2.0. See third-party tools/resources that Ambari uses and their respective authors'.

- One window appears for confirmation, click ok. It will start decommissioning the node

The screenshot shows the Ambari web interface at the URL `172.26.60.16:8080/#/main/hosts/impetus-i0163.impetus.co.in/summary`. The 'Hosts' tab is active, displaying the summary for the host 'impetus-i0163.impetus.co.in'. The page is divided into two main sections: 'Components' and 'Host Metrics'.

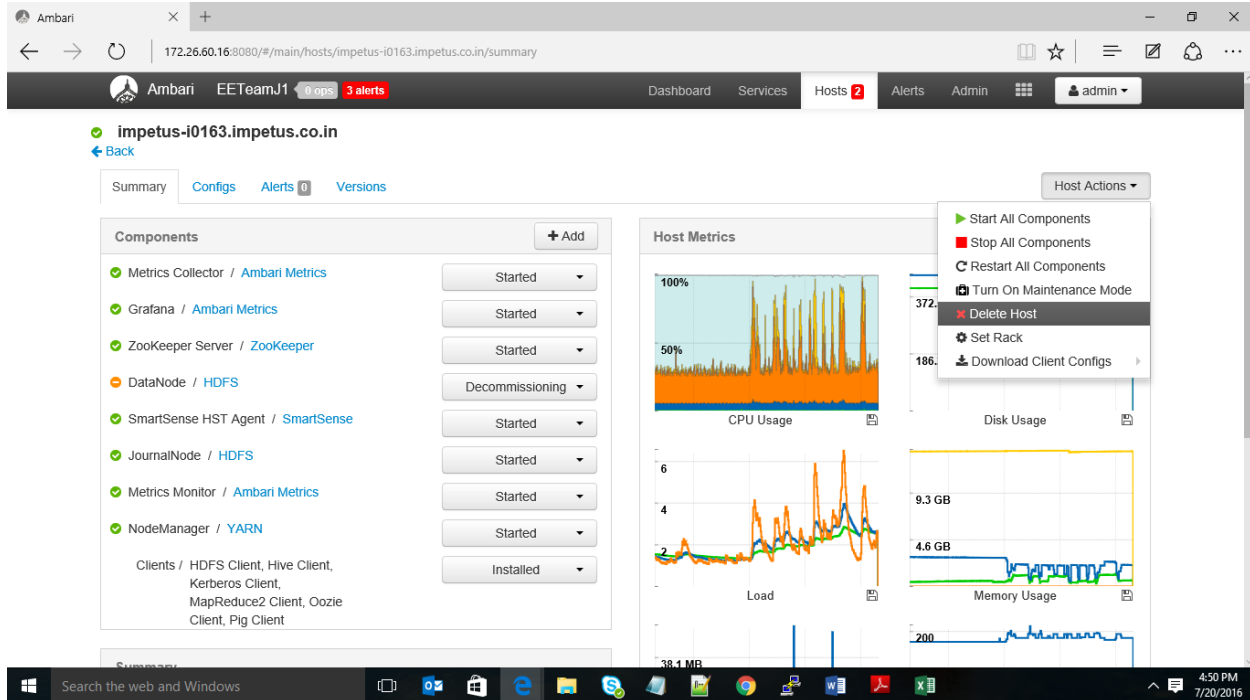
Components:

- Metrics Collector / Ambari Metrics: Started
- Grafana / Ambari Metrics: Started
- ZooKeeper Server / ZooKeeper: Started
- DataNode / HDFS: Decommissioning
- SmartSense HST Agent / SmartSense: Started
- JournalNode / HDFS: Started
- Metrics Monitor / Ambari Metrics: Started
- NodeManager / YARN: Started
- Clients / HDFS Client, Hive Client, Kerberos Client, MapReduce2 Client, Oozie Client, Pig Client: Installed

Host Metrics:

- CPU Usage: 100%
- Disk Usage: 372.5 GB
- Load: 38.1 MB
- Memory Usage: 9.3 GB

- After decommission is done click on **Host Actions** in right hand side of the window, and click on the delete host. (if required*)



6. During the decommission process the data present in *node* is dispersed to other *nodes*. Hadoop takes care of this procedure. Once the decommission process is completed we can safely remove the decommissioned node from the cluster.

Steps to commission a node:-

1. Add the network addresses of the new nodes to the include file in `hdfs-site.xml`

```
<property>
<name>dfs.hosts</name>
<value>/<hadoop-home>/conf/includes</value>
<final>true</final>
</property>
mapred-site.xml
<property>
<name>mapred.hosts</name>
<value>/<hadoop-home>/conf/includes</value>
<final>true</final>
</property>
```

Datanodes that are permitted to connect to the namenode are specified in a file whose name is specified by the `dfs.hosts` property. Includes file resides on the namenode's local filesystem, and it

contains a line for each datanode, specified by network address (as reported by the datanode; you can see what this is by looking at the namenode's web UI). If you need to specify multiple network addresses for a datanode, put them on one line, separated by whitespace.

eg :

slave01

slave02

slave03

.....

Similarly, tasktrackers that may connect to the jobtracker are specified in a file whose name is specified by the `mapred.hosts` property. In most cases, there is one shared file, referred to as the include file, that both `dfs.hosts` and `mapred.hosts` refer to, since nodes in the cluster run both datanode and tasktracker daemons.

2. Update the namenode with the new set of permitted datanodes using this

command:

% *hadoop dfsadmin -refreshNodes*

3. Update the job tracker with the new set of permitted task trackers using this command:

% *hadoop mradmin -refreshNodes*

4. Update the slaves file with the new nodes, so that they are included in future operations performed by the Hadoop control scripts.

5. Start the new data nodes and task trackers.

6. Check that the new data nodes and task trackers appear in the web UI.