HADOOP - MULTI NODE CLUSTER

This chapter explains the setup of the Hadoop Multi-Node cluster on a distributed environment.

As the whole cluster cannot be demonstrated, we are explaining the Hadoop cluster environment using three systems *onemasterandtwoslaves*; given below are their IP addresses.

- Hadoop Master: 192.168.1.15 $hadoop - master$
- Hadoop Slave: 192.168.1.16 $hadoop - slave - 1$
- Hadoop Slave: 192.168.1.17 $hadoop - slave - 2$

Follow the steps given below to have Hadoop Multi-Node cluster setup.

Installing Java

Java is the main prerequisite for Hadoop. First of all, you should verify the existence of java in your system using “java -version”. The syntax of java version command is given below.

```
$ java -version
```

If everything works fine it will give you the following output.

```
java version "1.7.0_71"
Java(TM) SE Runtime Environment (build 1.7.0_71-b13)
Java HotSpot(TM) Client VM (build 25.0-b02, mixed mode)
```

If java is not installed in your system, then follow the given steps for installing java.

Step 1

Download java (JDK - X64.tar.gz) by visiting the following link
http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html

Then **jdk-7u71-linux-x64.tar.gz** will be downloaded into your system.

Step 2

Generally you will find the downloaded java file in Downloads folder. Verify it and extract the **jdk-7u71-linux-x64.gz** file using the following commands.

```
$ cd Downloads/
$ ls
jdk-7u71-Linux-x64.gz
$ tar zxf jdk-7u71-Linux-x64.gz
$ ls
jdk1.7.0_71 jdk-7u71-Linux-x64.gz
```

Step 3

To make java available to all the users, you have to move it to the location “/usr/local/”. Open the root, and type the following commands.

```
$ su
password:
# mv jdk1.7.0_71 /usr/local/
# exit
```

Step 4

For setting up **PATH** and **JAVA_HOME** variables, add the following commands to **~/.bashrc** file.

```
export JAVA_HOME=/usr/local/jdk1.7.0_71
export PATH=PATH:$JAVA_HOME/bin
```

Now verify the **java -version** command from the terminal as explained above. Follow the above process and install java in all your cluster nodes.

Creating User Account

Create a system user account on both master and slave systems to use the Hadoop installation.

```
# useradd hadoop
# passwd hadoop
```

Mapping the nodes

You have to edit **hosts** file in **/etc/** folder on all nodes, specify the IP address of each system followed by their host names.

```
# vi /etc/hosts
enter the following lines in the /etc/hosts file.
192.168.1.109 hadoop-master
192.168.1.145 hadoop-slave-1
192.168.56.1 hadoop-slave-2
```

Configuring Key Based Login

Setup ssh in every node such that they can communicate with one another without any prompt for password.

```
# su hadoop
$ ssh-keygen -t rsa
$ ssh-copy-id -i ~/.ssh/id_rsa.pub tutorialspoint@hadoop-master
$ ssh-copy-id -i ~/.ssh/id_rsa.pub hadoop_tp1@hadoop-slave-1
$ ssh-copy-id -i ~/.ssh/id_rsa.pub hadoop_tp2@hadoop-slave-2
$ chmod 0600 ~/.ssh/authorized_keys
$ exit
```

Installing Hadoop

In the Master server, download and install Hadoop using the following commands.

```
# mkdir /opt/hadoop
# cd /opt/hadoop/
# wget http://apache.mesi.com.ar/hadoop/common/hadoop-1.2.1/hadoop-1.2.0.tar.gz
# tar -xzf hadoop-1.2.0.tar.gz
# mv hadoop-1.2.0 hadoop
# chown -R hadoop /opt/hadoop
# cd /opt/hadoop/hadoop/
```

Configuring Hadoop

You have to configure Hadoop server by making the following changes as given below.

core-site.xml

Open the **core-site.xml** file and edit it as shown below.

```
<configuration>
   <property>
      <name>fs.default.name</name>
      <value>hdfs://hadoop-master:9000/</value>
```

```
   </property>
   <property>
      <name>dfs.permissions</name>
      <value>false</value>
   </property>
</configuration>
```

hdfs-site.xml

Open the **hdfs-site.xml** file and edit it as shown below.

```
<configuration>
   <property>
      <name>dfs.data.dir</name>
      <value>/opt/hadoop/hadoop/dfs/name/data</value>
      <final>true</final>
   </property>

   <property>
      <name>dfs.name.dir</name>
      <value>/opt/hadoop/hadoop/dfs/name</value>
      <final>true</final>
   </property>

   <property>
      <name>dfs.replication</name>
      <value>1</value>
   </property>
</configuration>
```

mapred-site.xml

Open the **mapred-site.xml** file and edit it as shown below.

```
<configuration>
   <property>
      <name>mapred.job.tracker</name>
      <value>hadoop-master:9001</value>
   </property>
</configuration>
```

hadoop-env.sh

Open the **hadoop-env.sh** file and edit JAVA_HOME, HADOOP_CONF_DIR, and HADOOP_OPTS as shown below.

Note: Set the JAVA_HOME as per your system configuration.

```
export JAVA_HOME=/opt/jdk1.7.0_17 export HADOOP_OPTS=-Djava.net.preferIPv4Stack=true
export HADOOP_CONF_DIR=/opt/hadoop/hadoop/conf
```

Installing Hadoop on Slave Servers

Install Hadoop on all the slave servers by following the given commands.

```
# su hadoop
$ cd /opt/hadoop
$ scp -r hadoop hadoop-slave-1:/opt/hadoop
$ scp -r hadoop hadoop-slave-2:/opt/hadoop
```

Configuring Hadoop on Master Server

Open the master server and configure it by following the given commands.

```
# su hadoop
$ cd /opt/hadoop/hadoop
```

Configuring Master Node

```
$ vi etc/hadoop/masters
hadoop-master
```

Configuring Slave Node

```
$ vi etc/hadoop/slaves
hadoop-slave-1
hadoop-slave-2
```

Format Name Node on Hadoop Master

```
# su hadoop
$ cd /opt/hadoop/hadoop
$ bin/hadoop namenode –format
```

```
11/10/14 10:58:07 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = hadoop-master/192.168.1.109
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 1.2.0
STARTUP_MSG: build = https://svn.apache.org/repos/asf/hadoop/common/branches/branch-1.2 -
r 1479473; compiled by 'hortonfo' on Mon May 6 06:59:37 UTC 2013
STARTUP_MSG: java = 1.7.0_71
************************************************************/ 11/10/14
10:58:08 INFO util.GSet: Computing capacity for map BlocksMap
editlog=/opt/hadoop/hadoop/dfs/name/current/edits
……….
……….
……. 11/10/14 10:58:08 INFO common.Storage: Storage directory
/opt/hadoop/hadoop/dfs/name has been successfully formatted. 11/10/14 10:58:08 INFO
namenode.NameNode:
SHUTDOWN_MSG: /************************************************************
SHUTDOWN_MSG: Shutting down NameNode at hadoop-master/192.168.1.15
************************************************************/
```

Starting Hadoop Services

The following command is to start all the Hadoop services on the Hadoop-Master.

```
$ cd $HADOOP_HOME/sbin
$ start-all.sh
```

Adding a New DataNode in the Hadoop Cluster

Given below are the steps to be followed for adding new nodes to a Hadoop cluster.

Networking

Add new nodes to an existing Hadoop cluster with some appropriate network configuration. Assume the following network configuration.

For New node Configuration:

```
IP address : 192.168.1.103
netmask : 255.255.255.0
hostname : slave3.in
```

Adding User and SSH Access

Add a User

On a new node, add "hadoop" user and set password of Hadoop user to "hadoop123" or anything you want by using the following commands.

```
useradd hadoop
passwd hadoop
```

Setup Password less connectivity from master to new slave.

Execute the following on the master

```
mkdir -p $HOME/.ssh
chmod 700 $HOME/.ssh
ssh-keygen -t rsa -P '' -f $HOME/.ssh/id_rsa
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
chmod 644 $HOME/.ssh/authorized_keys
Copy the public key to new slave node in hadoop user $HOME directory
scp $HOME/.ssh/id_rsa.pub hadoop@192.168.1.103:/home/hadoop/
```

Execute the following on the slaves

Login to hadoop. If not, login to hadoop user.

```
su hadoop ssh -X hadoop@192.168.1.103
```

Copy the content of public key into file **"$HOME/.ssh/authorized_keys"** and then change the permission for the same by executing the following commands.

```
cd $HOME
mkdir -p $HOME/.ssh
chmod 700 $HOME/.ssh
cat id_rsa.pub >>$HOME/.ssh/authorized_keys
chmod 644 $HOME/.ssh/authorized_keys
```

Check ssh login from the master machine. Now check if you can ssh to the new node without a password from the master.

```
ssh hadoop@192.168.1.103 or hadoop@slave3
```

Set Hostname of New Node

You can set hostname in file **/etc/sysconfig/network**

```
On new slave3 machine
NETWORKING=yes
HOSTNAME=slave3.in
```

To make the changes effective, either restart the machine or run hostname command to a new machine with the respective hostname *restartisagoodoption*.

On slave3 node machine:

hostname slave3.in

Update **/etc/hosts** on all machines of the cluster with the following lines:

```
192.168.1.102 slave3.in slave3
```

Now try to ping the machine with hostnames to check whether it is resolving to IP or not.

On new node machine:

```
ping master.in
```

Start the DataNode on New Node

Start the datanode daemon manually using **$HADOOP_HOME/bin/hadoop-daemon.sh script**. It will automatically contact the master *NameNode* and join the cluster. We should also add the new node to the conf/slaves file in the master server. The script-based commands will recognize the new node.

Login to new node

```
su hadoop or ssh -X hadoop@192.168.1.103
```

Start HDFS on a newly added slave node by using the following command

```
./bin/hadoop-daemon.sh start datanode
```

Check the output of jps command on a new node. It looks as follows.

```
$ jps
7141 DataNode
10312 Jps
```

Removing a DataNode from the Hadoop Cluster

We can remove a node from a cluster on the fly, while it is running, without any data loss. HDFS provides a decommissioning feature, which ensures that removing a node is performed safely. To use it, follow the steps as given below:

Step 1: Login to master

Login to master machine user where Hadoop is installed.

```
$ su hadoop
```

Step 2: Change cluster configuration

An exclude file must be configured before starting the cluster. Add a key named dfs.hosts.exclude to our **$HADOOP_HOME/etc/hadoop/hdfs-site.xml** file. The value associated with this key provides the full path to a file on the NameNode's local file system which contains a list of machines which are not permitted to connect to HDFS.

For example, add these lines to **etc/hadoop/hdfs-site.xml** file.

```
<property>
   <name>dfs.hosts.exclude</name>
   <value>/home/hadoop/hadoop-1.2.1/hdfs_exclude.txt</value>
   <description>DFS exclude</description>
</property>
```

Step 3: Determine hosts to decommission

Each machine to be decommissioned should be added to the file identified by the hdfs_exclude.txt, one domain name per line. This will prevent them from connecting to the NameNode. Content of the **"/home/hadoop/hadoop-1.2.1/hdfs_exclude.txt"** file is shown below, if you want to remove DataNode2.

```
slave2.in
```

Step 4: Force configuration reload

Run the command **"$HADOOP_HOME/bin/hadoop dfsadmin -refreshNodes"** without the quotes.

```
$ $HADOOP_HOME/bin/hadoop dfsadmin -refreshNodes
```

This will force the NameNode to re-read its configuration, including the newly updated ‘excludes’ file. It will decommission the nodes over a period of time, allowing time for each node's blocks to be replicated onto machines which are scheduled to remain active.

On **slave2.in**, check the jps command output. After some time, you will see the DataNode process is shutdown automatically.

Step 5: Shutdown nodes

After the decommission process has been completed, the decommissioned hardware can be safely shut down for maintenance. Run the report command to dfsadmin to check the status of decommission. The following command will describe the status of the decommission node and the connected nodes to the cluster.

```
$ $HADOOP_HOME/bin/hadoop dfsadmin -report
```

Step 6: Edit excludes file again

Once the machines have been decommissioned, they can be removed from the ‘excludes’ file. Running **"$HADOOP_HOME/bin/hadoop dfsadmin -refreshNodes"** again will read the excludes file back into the NameNode; allowing the DataNodes to rejoin the cluster after the maintenance has been completed, or additional capacity is needed in the cluster again, etc.

Special Note: If the above process is followed and the tasktracker process is still running on the node, it needs to be shut down. One way is to disconnect the machine as we did in the above steps. The Master will recognize the process automatically and will declare as dead. There is no need to follow the same process for removing the tasktracker because it is NOT much crucial as compared to the DataNode. DataNode contains the data that you want to remove safely without any loss of data.

The tasktracker can be run/shutdown on the fly by the following command at any point of time.

```
$ $HADOOP_HOME/bin/hadoop-daemon.sh stop tasktracker $HADOOP_HOME/bin/hadoop-daemon.sh
start tasktracker
```