***MULTI NODE CLUSTER IN HADOOP BY USING UBUNTU***

**Abstract**

Hadoop is a framework written in Java for running applications on large clusters of commodity hardware and incorporates features similar to those of the Google File System (GFS) and of the MapReduce computing paradigm. Hadoop’s HDFS is a highly fault-tolerant distributed file system and, like Hadoop in general, designed to be deployed on low-cost hardware. It provides high throughput access to

In a previous tutorial, I described how to setup up a Hadoop single-node cluster on an Ubuntu box. The main goal of this tutorial is to get a more sophisticated Hadoop installation up and running, namely building a multi-node cluster using two Ubuntu boxes.

We will build a multi-node cluster using two Ubuntu boxes in this tutorial. In my humble opinion, the best way to do this for starters is to install, configure and test a “local” Hadoop setup for each of the two Ubuntu boxes, and in a second step to “merge” these two single-node clusters into one multi-node cluster in which one Ubuntu box will become the designated master (but also act as a slave with regard to data storage and processing), and the other box will become only a slave. It’s much easier to track down any problems you might encounter due to the reduced complexity of doing a single-node cluster setup first on each machine.

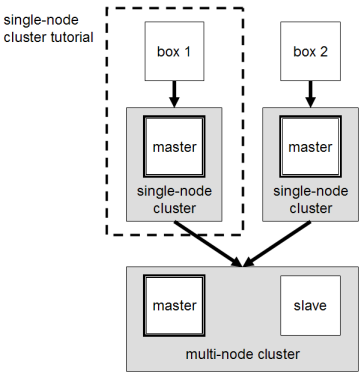
The tutorial approach outlined above means that you should read now my previous tutorial on how to setup up a Hadoop single-node cluster and follow the steps described there to build a single-node Hadoop cluster on each of the two Ubuntu boxes. It is recommended that you use the ‘‘same settings’’ (e.g., installation locations and paths) on both machines, or otherwise you might run into problems later when we will migrate the two machines to the final multi-node cluster setup.

Just keep in mind when setting up the single-node clusters that we will later connect and “merge” the two machines, so pick reasonable network settings etc. now for a smooth transition later.

**Cluster Creation:**

Now that you have two single-node clusters up and running, we will modify the Hadoop configuration to make one Ubuntu box the “master” (which will also act as a slave) and the other Ubuntu box a “slave”.

Shutdown each single-node cluster with (**bin/stop-all.sh**) before continuing if you haven’t done so already.



**Networking:**

It is the important for to connect the both devices, by using wireless or wired connection we connect the both system in a single network. While using the network it will generate the different ip’s to the both systems, it means dynamic ip address.

This should come hardly as a surprise, but for the sake of completeness I have to point out that both machines must be able to reach each other over the network. The easiest is to put both machines in the same network with regard to hardware and software configuration, for example connect both machines via a single hub or switch and configure the network interfaces to use a common network such as 192.168.0.x/24.

To make it simple, we will assign the IP address 192.168.0.1 to the master machine and 192.168.0.2 to the slave machine. Update /etc/hosts on both machines with the following lines:

# Update /etc/hosts for master AND slave

192.168.0.1 master

192.168.0.2 slave

Check the both systems are connected to the single network by using following command:

#it shows your ip address

**$ ifconfig**

**$ ping 192.162.0.1 (the slave ip address) in master’s System**

NOTE: You must get 0% packet loss.

Check above in slave also.

**# Distribute the SSH public key of hduser@master**

**hduser@master:~$ ssh-copy-id -i $HOME/.ssh/id\_rsa.pub hduser@slave**

This command will prompt you for the login password for user hduser on slave, then copy the public SSH key for you, creating the correct directory and fixing the permissions as necessary.

The final step is to test the SSH setup by connecting with user hduser from the master to the user account hduser on the slave. The step is also needed to save slave’s host key fingerprint to the hduser@master’s known\_hosts file.

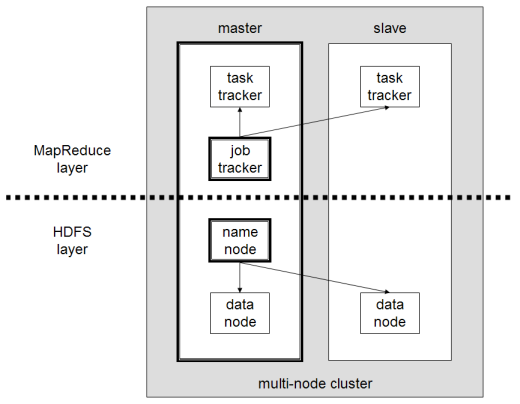
So, connecting from master to master… by using following commands

**hduser@master:~$ ssh master**

and from master to slave………

**hduser@master:~$ ssh slave**

**Cluster Overview**

The next step will describe how to configure one Ubuntu box as a master node and the other Ubuntu box as a slave node. The master node will also act as a slave because we only have two machines available in our cluster but still want to spread data storage and processing to multiple machines.

The master node will run the “master” daemons for each layer: NameNode for the HDFS storage layer, and JobTracker for the MapReduce processing layer. Both machines will run the “slave” daemons: DataNode for the HDFS layer, and TaskTracker for MapReduce processing layer. Basically, the “master” daemons are responsible for coordination and management of the “slave” daemons while the latter will do the actual data storage and data processing work.

Let, up to here we see how to connect the two systems to the single network and configuration to the both systems.

Let, move up on to the next step edit the **core-site.xml, yarn-site.xml and hdfs-site.xml**.

First, we have to change the fs.default.name parameter (in conf/core-site.xml), which specifies the NameNode (the HDFS master) host and port. In our case, this is the master machine.

By using this following command to edit the **core-site.xml, yarn-site.xml, hdfs-site.xml** using nano or gedit editor.

**$ sudo nano /usr/local/hadoop/etc/hadoop/core-site.xml**

<property>

<name>fs.default.name</name>

<value>hdfs://master:54310</value>

</property>

**$ sudo nano /usr/local/hadoop/etc/hadoop/hdfs-site.xml**

<property>

<name>dfs.replication</name>

<value>2</value> (replication is the count of datanodes)

</property>

**Keep this entry in master only, delete from slaves**

<property>

<name>dfs.namenode.name.dir</name>

<value>file:/usr/loacal/hadoop.tmp/hdfs/namenode</value>

</property>

**Keep this entry in slave only, delete from master**

<property>

<name>dfs.datanode.name.dir</name>

<value>file:/usr/loacal/hadoop.tmp/hdfs/datanodenode</value>

</property>

**$ sudo nano /usr/local/hadoop/etc/hadoop/yarn-site.xml**

<property>

<name>yarn.resourcemanager.resource-tracker.address</name>

<value>master:8025</value>

</property>

<property>

<name>yarn.resourcemanager.scheduler.address</name>

<value>master:8030</value>

</property>

<property>

<name>yarn.resourcemanager.address</name>

<value>maser:8050</value>

</property>

**$ sudo nano /usr/local/hadoop/etc/hadoop/mapred-site.xml**

<property>

<name>mapred.jobhistory.address</name>

<value>master:9000</value>

</property>

Now we are going to the next step, Update Masters and slave file(Master Node only)

If we see any entry related to localhost, feel free to delete it. This file is just helper file that are used by hadoop script to start appropriate services on master and slave nodes.

**$sudo nano /usr/local/hadoop/etc/hadoop/slaves** #edit in master

Slave1

Slave2

**$sudo nano /usr/local/hadoop/etc/hadoop/master #**edit in master

master

Recreate Name node folder in Master only

**$ sudo rm -rf /usr/local/hadoop\_tmp**

**$ mkdir -p /usr/local/hadoop\_tmp/hdfs/namenode**

**$ chown hduser:hadoop -R /usr/local/hadoop\_tmp/**

**$ sudo chmod 777 /usr/local/hadoop\_tmp/hdfs/namenode**

Recreate Name node folder in all Slaves only

**$ sudo rm -rf /usr/local/hadoop\_tmp**

**$ mkdir -p /usr/local/hadoop\_tmp/hdfs/datanode**

**$ chown hduser:hadoop -R /usr/local/hadoop\_tmp/**

**$ sudo chmod 777 /usr/local/hadoop\_tmp/hdfs/datanode**

Format the name node (Master only)

Before starting the cluster, we need to format the name node. Use the following command only on master node:

**$hdfs namenode -format**

Start the DFS & Yarn (Master Only)

**$start -all.sh**

Use the following command it show tracks the daemons in Master and Slaves

**$ jps**

**In master it shows**

**2456 NameNode**

**2609 Secondary node**

**2756 Resource Manager**

**2890 Jps**

**In Slaves it shows**

**2568 NodeManager**

**2609 Jps**

**2289 DataNode**