**CREATING HADOOP CLUSTER ON AWS**

**Sriram Cherukupalli**

**3144766**

AMI Name- Sriram\_Cherukupalli

AMI ID- ami-5b14233e

**HADOOP and JAVA INSTALLATION**

Tools required: Putty, WinSCP and AWS account

* I have created a Hadoop cluster using public IP and created a password less setup so anyone can access my cluster.

You can access my NameNode from here-

<http://ec2-18-222-18-49.us-east-2.compute.amazonaws.com:50070>

My data node links:

**Datanode001**

[http://ec2-18-218-183-133.us-east-2.compute.amazonaws.com:50075](http://ec2-18-218-183-133.us-east-2.compute.amazonaws.com:50075/)

**Datanode002**

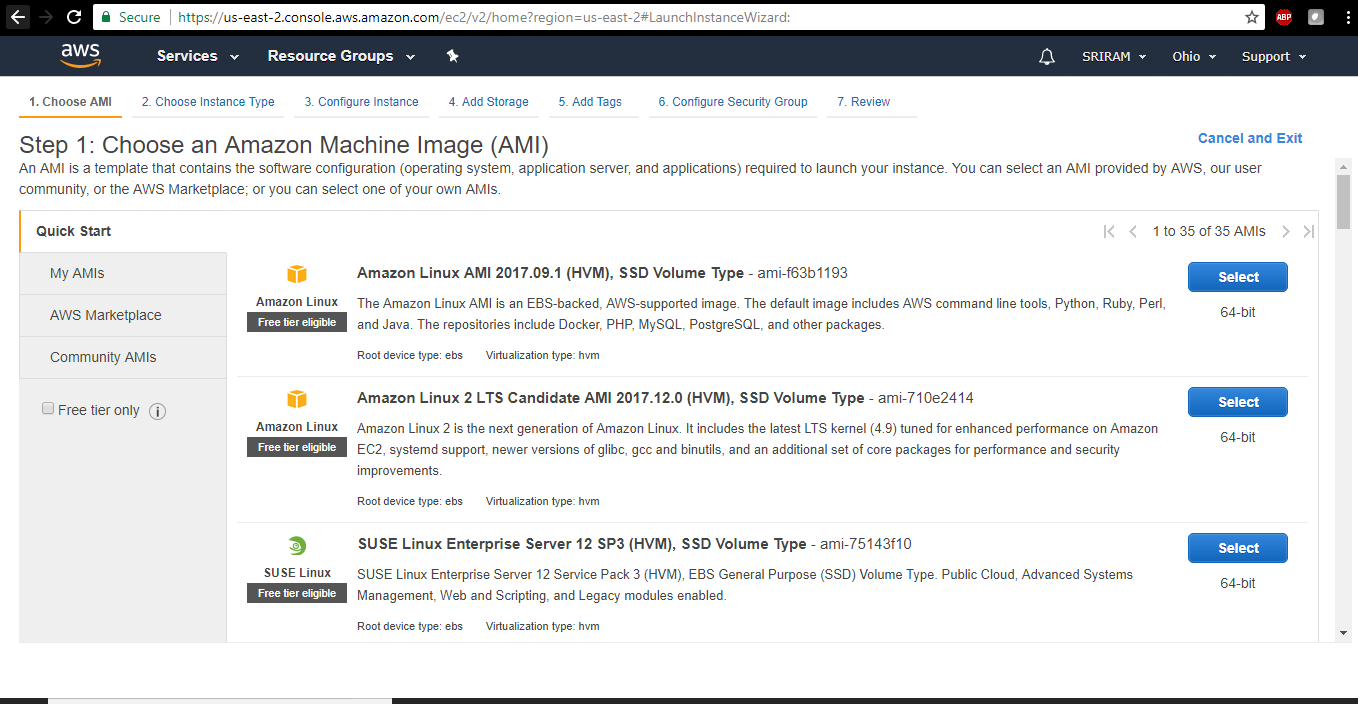
[http://ec2-18-219-76-71.us-east-2.compute.amazonaws.com:50075](http://ec2-18-219-76-71.us-east-2.compute.amazonaws.com:50075/)

**Datanode003**

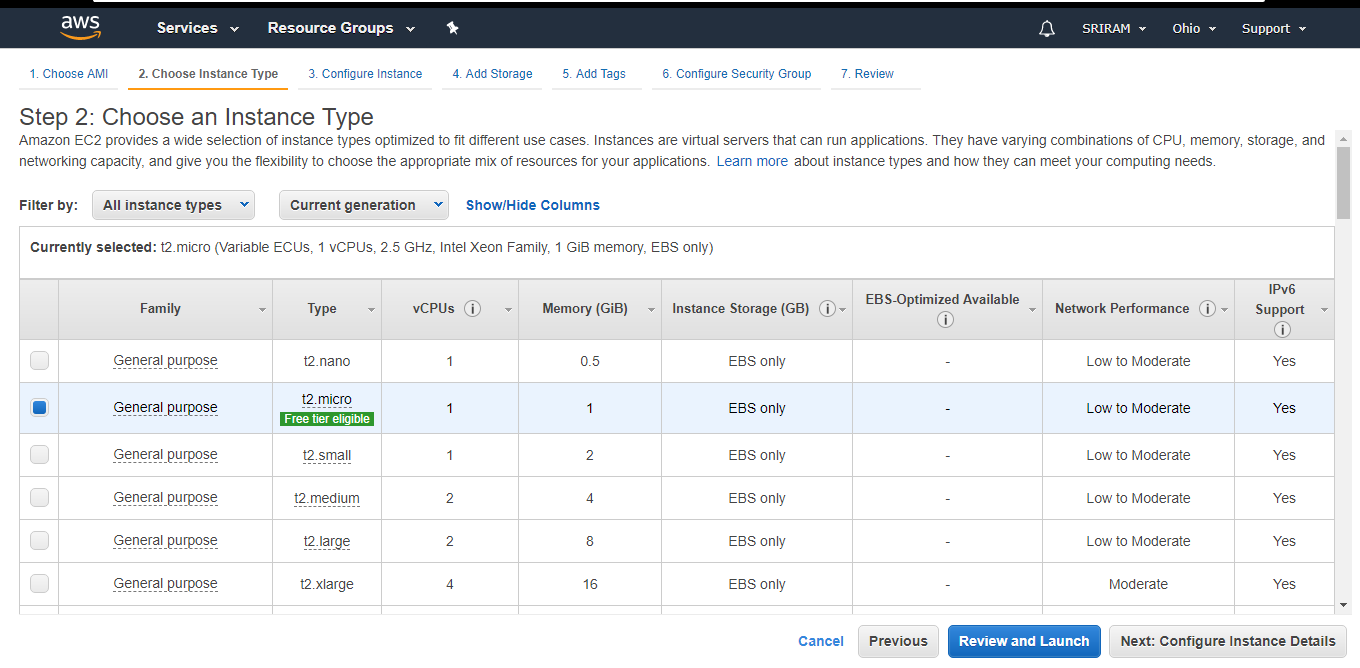
[http://ec2-18-222-11-253.us-east-2.compute.amazonaws.com:50075](http://ec2-18-222-11-253.us-east-2.compute.amazonaws.com:50075/)

STEPS:

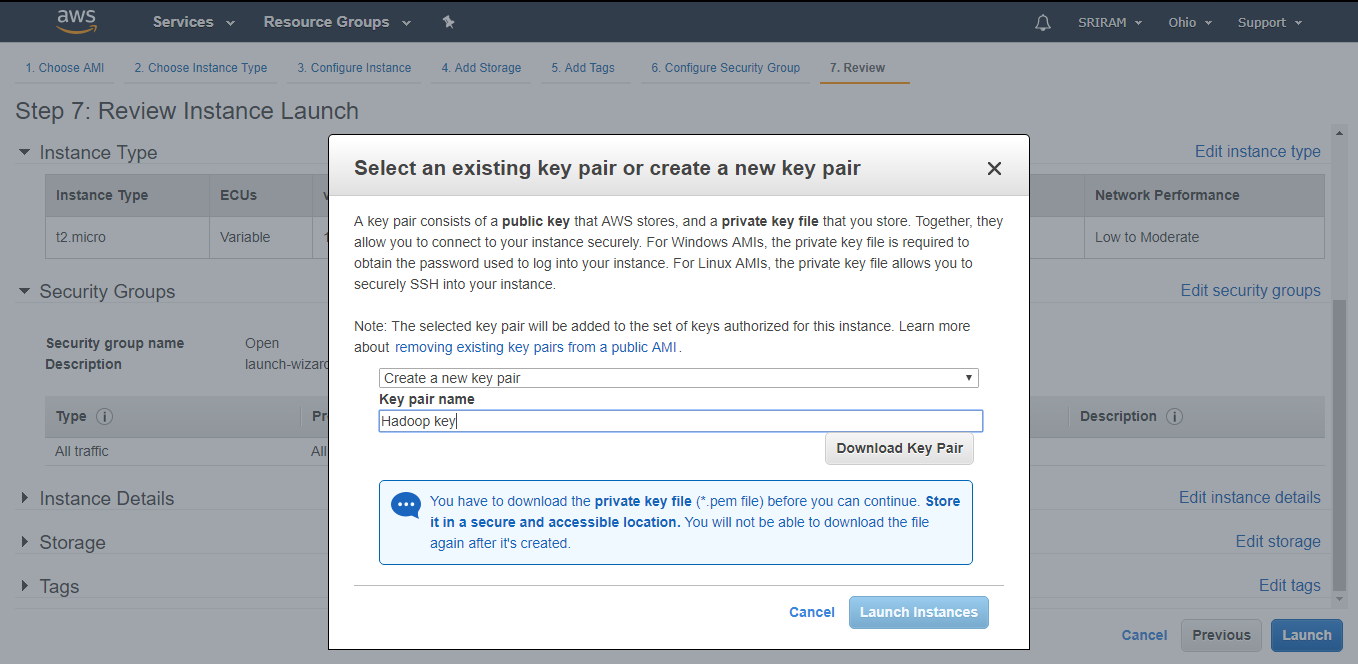
1. For creating a cluster you need to setup an image. For that firstly we need to login into our amazon account and navigate to EC2 and click launch button. Once we press that we will be navigated into next screen where we have to select an AMI for our application.



1. Once we select the type of image we are then asked to choose the instance type.



1. After we select the instance type we need to add storage which I set up to 8 gb. We then need to configure the instance, for which I set ‘4’ for to create the cluster. Then we need to configure the security group for which I have chosen the type ‘**SSH’** and source ‘**Anywhere**’ so that anyone can access my cluster.
2. At last, a review tab will pop up just to make sure we have set it up perfectly before launching it. If all the specifications match your requirements click on ‘**Launch’**.



1. Once you click launch button we will be asked to name the key pair which is a .pem file which we later use for setting up the cluster.
2. Now I viewed my cluster and renamed the nodes as:

NameNode

Datanode001

Datanode002

Datanode003

Keep track of your DNS of all created nodes so we can use them later.

export NameNodeDNS="ec2-18-216-98-190.us-east-2.compute.amazonaws.com"

export DataNode001DNS="ec2-18-219-160-224.us-east-2.compute.amazonaws.com"

export DataNode002DNS="ec2-18-219-245-223.us-east-2.compute.amazonaws.com"

export DataNode003DNS="ec2-18-217-180-236.us-east-2.compute.amazonaws.com"

export NameNodeIP="172.31.31.149"

export DataNode001IP="172.31.22.98"

export DataNode002IP="172.31.31.135"

export DataNode003IP="172.31.21.114"

export IdentityFile="~/.ssh/Hadoop-clusterkey.pem"

**SETTING UP PASSWORD-LESS SSH**

1. Now we need two tools i.e. Putty and WinSCP.

First open PuttyGen go to file->Load private key and then select the .pem file which we have downloaded and open it there. Save the private key generated.

1. Now open up Putty and in the host name fill the Public DNS of your NameNode.

We need to write the host name as username@DNS, in my case

[ubuntu@ec2-18-216-98-190.us-east2.compute.amazonaws.com](mailto:ubuntu@ec2-18-216-98-190.us-east2.compute.amazonaws.com)

Then give a name to your session so that you can recognize your name node.

1. Now on Putty navigate to Connection->SSH->Auth, then browse and load the .ppk file generated by PuttyGen. Now save this session and duplicate the NameNode and do the same for all the NameNodes we created.
2. Now open WinSCP and import all the data from Putty. Then, I set up my cluster and configured them.

**INSTALLING JAVA AND HADOOP ON THE CLUSTER**

1. Now open putty and run the command

**Sudo apt-get –y updateBashCopy** to update all the packages of all the nodes.

1. Then run the command

**sudo apt-get -y install default-jdk** on each of nodes to install latest version of java.

1. After we install java we need to set up the environment variables. Which can be done using the following code on all the nodes separately:

echo "# JAVA Variables START" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

echo "export JAVA\_HOME=/usr/lib/jvm/default-java" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

echo "PATH=\$PATH:\$JAVA\_HOME/bin" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

echo "# JAVA Variables END" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

After installation we need to restart the instances because we have set up the environment variable for java.

1. Now, we can download and install Hadoop on all of the single nodes. Before that I used the following command to create a path to save all the installation files:

**wget http://apache.forsale.plus/hadoop/common/hadoop-2.9.0/hadoop-2.9.0.tar.gz -P ~/Downloads/Hadoop**

Then install Hadoop using the command:

**sudo tar –zxvf ~/Downloads/Hadoop/adoop-\*.tar.gz –C /usr/local**

1. Now we need to setup the environment variable for Hadoop, which can be done using the following code on each node:

sudo echo -e "# HADOOP Variables START" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

sudo echo -e "export HADOOP\_HOME='/usr/local/hadoop'" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

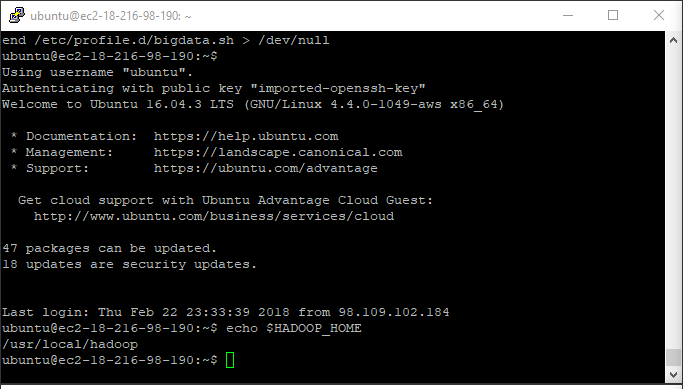
sudo echo -e "export HADOOP\_CONF\_DIR=\"\${HADOOP\_HOME}/etc/hadoop\"" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

sudo echo -e "export HADOOP\_DATA\_HOME=\"\${HOME}/hadoop\_data/hdfs\"" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

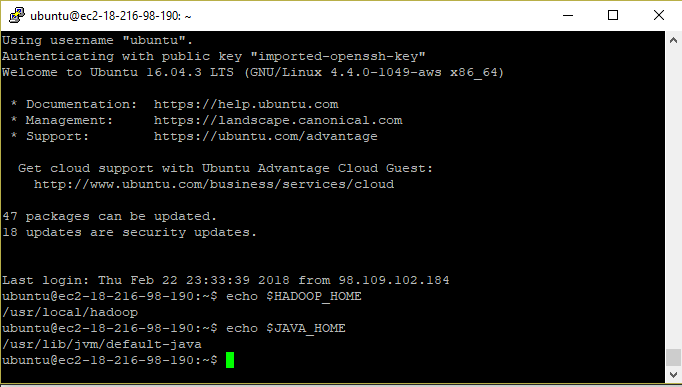
sudo echo -e "PATH=\$PATH:\$HADOOP\_HOME/bin:\$HADOOP\_HOME/sbin" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

sudo echo -e "# HADOOP Variables END" | sudo tee --append /etc/profile.d/bigdata.sh > /dev/null

1. Now to ensure we have set up java and Hadoop perfectly we can execute the following commands:

echo $HADOOP\_HOME

echo $JAVA\_HOME



1. I navigated to /usr/local/hadoop/etc/Hadoop on all nodes and changed my coresite.xml, yarn-site.xml, mapred-site.xml and hdfs-site.xml.

**Coresite.xml**

<configuration>

<!--Custom Properties-->

<property>

<name>thisnamenode</name>

<value>hadoop-master</value>

<description>NameNode is the hostname specified in the config file and etc/hosts file. It may be replaced with a DNS that points to your NameNode.</description>

</property>

<property>

<name>homefolder</name>

<value>/home/${user.name}</value>

</property>

<!--Hadoop Properties-->

<property>

<name>hadoop.tmp.dir</name>

<value>${homefolder}/hadoop\_data/hdfs/tmp</value>

<description>A base for other temporary directories.</description>

</property>

<property>

<name>fs.defaultFS</name>

<value>hdfs://${thisnamenode}:9000</value>

<description>localhost may be replaced with a DNS that points to the NameNode.</description>

</property>

<property>

<name>dfs.permissions</name>

<value>false</value>

</property>

</configuration>

**Yarn-site.xml:**

<configuration>

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

<property>

<name>mapred.job.tracker</name>

<value>${thisnamenode}:9001</value>

</property>

<!--<property>

<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>

<value>org.apache.hadoop.mapred.ShuffleHandler</value>

</property>-->

</configuration>

**Mapred-site.xml:**

<configuration>

<property>

<name>mapreduce.jobtracker.address</name>

<value>local</value>

</property>

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

</configuration>

**Hdfs-site.xml**

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<property>

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

<value>file:/usr/local/hadoop\_tmp/hdfs/namenode</value>

</property>

<property>

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

<value>file:/usr/local/hadoop\_tmp/hdfs/datanode</value>

</property>

<property>

<name>mapreduce.map.memory.mb</name>

<value>768</value>

</property>

<property>

<name>mapreduce.reduce.memory.mb</name>

<value>768</value>

</property>

1. Then I assigned a master(NameNode) and a secondary master(Datanode001) to the cluster. To do that I created a master’s file first using:

sudo touch $HADOOP\_CONF\_DIR/masters

then:

sudo rm -rf $HADOOP\_CONF\_DIR/masters echo -e "hadoop-master" | sudo tee --append $HADOOP\_CONF\_DIR/masters > /dev/null echo -e "DataNode001" | sudo tee --append $HADOOP\_CONF\_DIR/masters > /dev/null

Now, I have assigned the roles to the nodes.

1. Then I have assigned slave roles to the remaining nodes:

sudo rm -rf $HADOOP\_CONF\_DIR/slaves

echo -e "DataNode001" | sudo tee --append $HADOOP\_CONF\_DIR/slaves > /dev/null

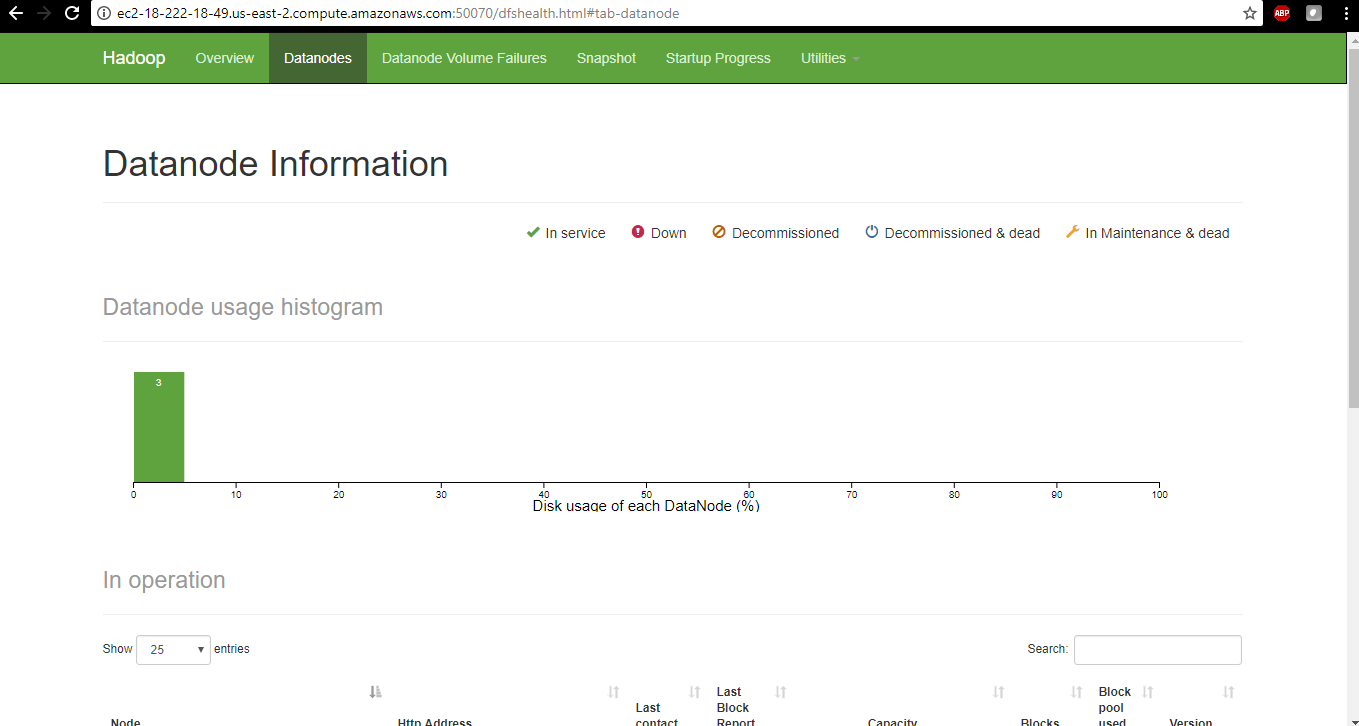
echo -e "DataNode002" | sudo tee --append $HADOOP\_CONF\_DIR/slaves > /dev/null

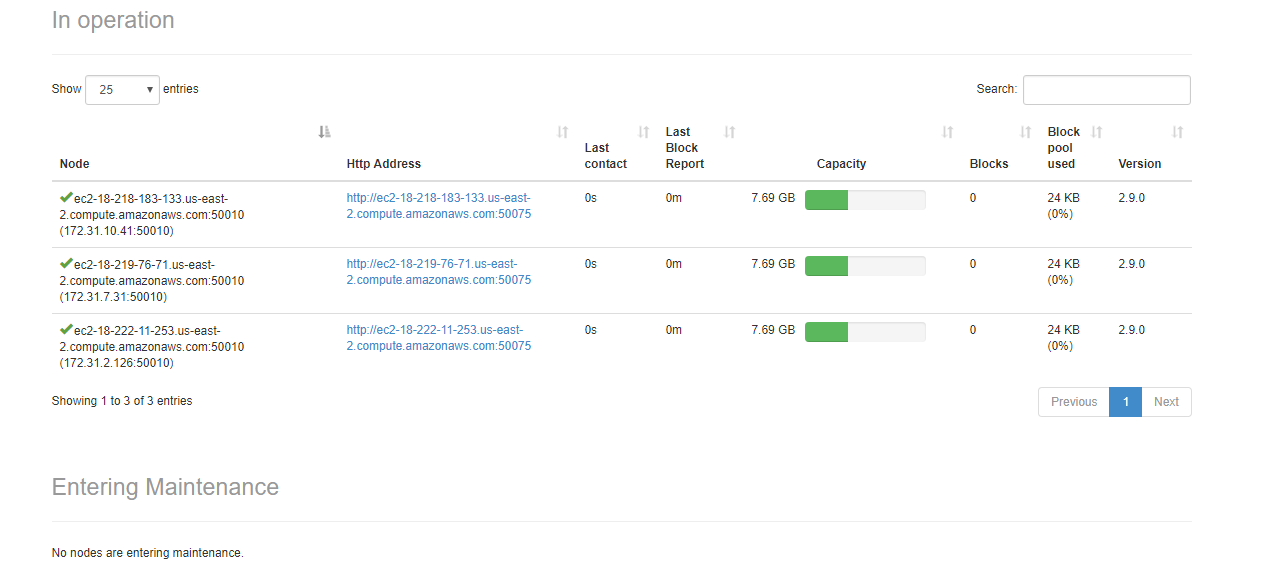
echo -e "DataNode003" | sudo tee --append $HADOOP\_CONF\_DIR/slaves > /dev/null

by doing this I have assigned slaves roles to the nodes(Datanode001, Datanode002 and Datanode003)

1. Then I used the command: start-all.sh to start the cluster.

Now, my NameNode looks like:





**RUNNING CODE ON THE CLUSTER**

1. Use Putty and WinSCP to connect to my cluster. I have zipped my .pem file (Hadoop-clusterkeypair.pem).
2. Now using WinSCP create directory /home/ubuntu/data on the namenode and copy all input files into data.
3. Create a directory /user in HDFS:

cd /usr/local/hadoop bin/hdfs dfs -mkdir /user

1. Copy the data folder to HDFS:

cd /usr/local/hadoop/ bin/hdfs dfs -put '/home/ubuntu/data' /user/

1. Create directory /home/ubuntu/wordcountf on namenode and copy the java application program, WordCount.java into wordcountf
2. Compile the java application program:

cd /home/ubuntu/wordcountf javac -classpath $HADOOP\_HOME/share/hadoop/common/hadoop-common-2.7.2.jar:$HADOOP\_HOME/share/hadoop/mapreduce/hadoop-mapreduce-client-core-2.7.2.jar:$HADOOP\_HOME/share/hadoop/common/lib/commons-cli-1.2.jar -d ./ \*.java

1. Create a folder wordcountc inside wordcountf and move all class files into wordcountc
2. Create jar file by executing the command below:

cd /home/ubuntu/wordcountf jar -cvf wordcountj.jar -C wordcountc .

This will create the jar file named wordcountj.jar

1. Now execute the program using:

cd /usr/local/hadoop bin/hadoop jar ~/wordcountf/wordcountj.jar WordCount /user/inputdata/ output2

1. Output is genetated inside two folders: output1(count of keywords in each state) and output2(top 3 states for each keyword)

cd /usr/local/hadoop hdfs dfs -cat /user/ubuntu/output1/part-r-00000 hdfs dfs -cat /user/ubuntu/output2/part-r-00000