Installation and Working On Multi Node Hadoop Cluster.

Expt No: 4 March 20, 2019
Author: Subalakshmi Shanthosi S (186001008)

Aim

Installation of Hadoop in multi node cluster.

Software's Used

- Ubuntu 16.04 LTS
- Hadoop 1.0.3

Description

1. A Multi Node cluster in Hadoop comprises of Two or More DataNodes.
2. A hadoop cluster can be referred to as a computational computer cluster for storing and analysing big data (structured, semi-structured and unstructured) in a distributed environment.
3. Components of Hadoop HDFS:
   (a) Master Node
      - Master node in a hadoop cluster is responsible for storing data in HDFS and executing parallel computation the stored data using MapReduce.
      - Master Node has 3 nodes :
        – NameNode
        – Secondary NameNode
        – JobTracker
      - JobTracker monitors the parallel processing of data using MapReduce.
      - NameNode handles the data storage function with HDFS.
      - The secondary NameNode keeps a backup of the NameNode data.

   (b) Slave/Worker Node-
      - This component in a hadoop cluster is responsible for storing the data and performing computations.
      - Every slave/worker node runs both a TaskTracker and a DataNode service to communicate with the Master node in the cluster.
      - The DataNode service is secondary to the NameNode and the TaskTracker service is secondary to the JobTracker.

   (c) Client Nodes -
      - Client node has hadoop installed with all the required cluster configuration settings and is responsible for loading all the data into the hadoop cluster.
      - Client node submits mapreduce jobs describing on how data needs to be processed and then the output is retrieved by the client node once the job processing is completed.

   (d) Differences between Single Node and MultiNode clusters:
      - Hadoop installed on multi node cluster is more efficient.
      - In the single node cluster, all the necessary demons like NameNode, DataNode, Resource Manager, Node manager and Application master etc run on the same machine but different ports.
      - Multi node cluster follows Master-Slave architecture. In multi node cluster (distributed mode), they run on different machines (master and slave).
      - The replication factor for multi node cluster will be more than one and it should be installed in more than one machine to satisfy the Master-Slave architecture.
      - Multi node cluster is basically used for full stack development of hadoop application and projects whereas Single Node Cluster is for testing purpose

(e) Components of Multi Node Cluster and Architecture:

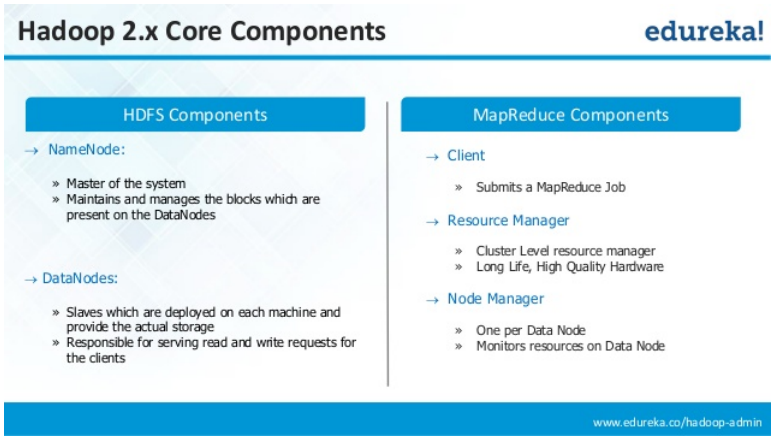

Figure 1: Hadoop Multi Node cluster components.

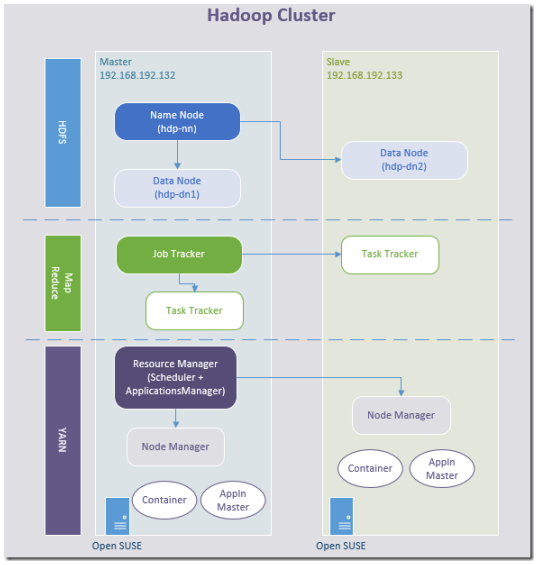

Figure 2: Hadoop Multi Node cluster architecture.

Procedure - Steps Involved:

- Configuring two Single node clusters by performing the following steps as follows.
  1. Launch Ubuntu 16.04 LTS.
  2. Login to the OS with sudo permission and install the following packages using apt-get command.
     - openssh-server
     - openssh-client
     - java jdk 8
     - javac compiler
     - hadoop 1.0.3
  3. Create a new user with sudo permission (hduser:hadoop).
  4. Log into the hduser and do the following:
     - Copy the hadoop executable to /usr/local directory.
     - Install and configure appropriate environment variables and parameters in the following configuration files:

    - conf/hadoop-env.sh : Configure JAVA_HOME and HADOOP_HOME with appropriate values.
    - conf/core-site.xml : Configure hadoop default temp directory and default file system.
    - conf/mapred-site.xml : JobTracker name and port number.
    - conf/hdfs-site.xml : Default replication factor specification.
  - Format the namenode by specified dfs.name.dir by running command : /usr/local/hadoop/bin/hadoop namenode -format .
  - Starting the local hadoop single node cluster by running command: /usr/local/hadoop/bin/start-all.sh .
  - To check the current running Hadoop Processes by running command : jps .
  - Stopping the local hadoop single node cluster by running command : /usr/local/hadoop/bin/stop-all.sh .
5. Networking:
  - Update /etc/hosts file with configuration as follows in both the machines:

```
# Update /etc/hosts for master AND slave
192.168.0.1    master
192.168.0.2    slave
```

Figure 3: Networking configuration in Master and Slave nodes.

6. Configuring SSH Access:
   - Adding hduser@master public SSH key to the authorized_keys file of hduser@ slave .

```
# Distribute the SSH public key of hduser@master
hduser@master:~$ ssh-copy-id -i $HOME/.ssh/id_rsa.pub hduser@slave
```

Figure 4: Copying SSH public key to slave from master node.

```
hduser@master:~$ ssh master
The authenticity of host 'master (192.168.0.1)' can't be established.
RSA key fingerprint is 3b:21:b3:c0:21:5c:7c:54:2f:1e:2d:96:79:eb:7f:95.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'master' (RSA) to the list of known hosts.
Linux master 2.6.20-16-386 #2 Thu Jun 7 20:16:13 UTC 2007 i686
...
hduser@master:~$
```

Figure 5: SSH to master node.

```
hduser@master:~$ ssh slave
The authenticity of host 'slave (192.168.0.2)' can't be established.
RSA key fingerprint is 74:d7:61:86:db:86:8f:31:90:9c:68:b0:13:88:52:72.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'slave' (RSA) to the list of known hosts.
Ubuntu 10.04
...
hduser@slave:~$
```

Figure 6: SSH to Slave node.

```
######################################################################
## Basic Initialization script To Install Hadoop as Distributed Mode
## Hadoop 2.6.5
## CP5261 - Big Data Analytics Lab - Semester 2 - 2017-18
######################################################################


JAVA_HOME
/usr/lib/jvm/java-7-openjdk-amd64/jre/bin/java
/usr/lib/jvm/java-7-openjdk-amd64/bin/javac

ssh-keygen
/home/hduser/.ssh/id_rsa

--------------------


Steps to install Hadoop.


sudo apt-get update
sudo apt-get install openjdk-7-jre
sudo apt-get install openjdk-7-jdk
sudo update-alternatives --config java
sudo update-alternatives --config javac
java -version
sudo addgroup hadoop
sudo adduser --ingroup hadoop hduser
su - hduser

$hduser ssh-keygen -t rsa -P ""
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

cat /proc/sys/net/ipv6/conf/all/disable_ipv6

wget https://archive.apache.org/dist/hadoop/core/hadoop-1.0.3/hadoop-1.0.3.tar.gz

clear
ls
sudo cp hadoop-1.0.3.tar.gz /usr/local/
ls
cd /usr/local
sudo tar xzf hadoop-1.0.3.tar.gz
ls
sudo mv hadoop-1.0.3 hadoop
sudo chown -R hduser:hadoop hadoop
$HOME
sudo nano /home/hduser/.bashrc
-----------
# Set Hadoop-related environment variables
export HADOOP_HOME=/usr/local/hadoop

# Set JAVA_HOME (we will also configure JAVA_HOME directly for Hadoop later on)
export JAVA_HOME=/usr/lib/jvm/java-6-sun

# Some convenient aliases and functions for running Hadoop-related commands
unalias fs &> /dev/null
alias fs="hadoop fs"
unalias hls &> /dev/null
alias hls="fs -ls"

# If you have LZO compression enabled in your Hadoop cluster and
# compress job outputs with LZOP (not covered in this tutorial):
# Conveniently inspect an LZOP compressed file from the command
# line; run via:
#
```

```
# $ lzohead /hdfs/path/to/lzop/compressed/file.lzo
#
# Requires installed 'lzop' command.
#
lzohead () {
    hadoop fs -cat $1 | lzop -dc | head -1000 | less
}

# Add Hadoop bin/ directory to PATH
export PATH=$PATH:$HADOOP_HOME/bin
--------------
sudo nano hadoop/conf/hadoop-env.sh
-----------
# The java implementation to use.  Required.
export JAVA_HOME=/usr/lib/jvm/java-6-sun
-----------

sudo mkdir -p /app/hadoop/tmp
sudo chown hduser:hadoop /app/hadoop/tmp
sudo chmod 750 /app/hadoop/tmp
sudo nano hadoop/conf/core-site.xml
----------
<property>
  <name>hadoop.tmp.dir</name>
  <value>/app/hadoop/tmp</value>
  <description>A base for other temporary directories.</description>
</property>

<property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:54310</value>
  <description>The name of the default file system.  A URI whose
  scheme and authority determine the FileSystem implementation.  The
  uri's scheme determines the config property (fs.SCHEME.impl) naming
  the FileSystem implementation class.  The uri's authority is used to
  determine the host, port, etc. for a filesystem.</description>
</property>
----------
sudo nano hadoop/conf/mapred-site.xml
----------
<property>
  <name>mapred.job.tracker</name>
  <value>localhost:54311</value>
  <description>The host and port that the MapReduce job tracker runs
  at.  If "local", then jobs are run in-process as a single map
  and reduce task.
  </description>
</property>

----------
sudo nano hadoop/conf/hdfs-site.xml
----------
<property>
  <name>dfs.replication</name>
  <value>1</value>
  <description>Default block replication.
  The actual number of replications can be specified when the file is created.
  The default is used if replication is not specified in create time.
  </description>
</property>

----------
clear

hduser@ubuntu:~$ /usr/local/hadoop/bin/hadoop namenode -format
```

```
hduser@ubuntu:~$ /usr/local/hadoop/bin/start-all.sh

hduser@ubuntu:/usr/local/hadoop$ jps

hduser@ubuntu:~$ /usr/local/hadoop/bin/stop-all.sh


-----------------
namenode [master] -  10.240.14.10       / 130.211.243.46
datanode1 [slave1] - 10.240.11.153  / 107.167.181.185


-------------ssh public key - namenode
In google computer install dashboard copy the /home/hduser/id_rsa.pub <namenode>
to datanode instance <ssh>.

-------------------------Multiple Nodes

After copy ssh to datanode.

<in master node>
hduser@master:~$ ssh master
hduser@master:~$ ssh slave

hduser@master:~$ bin/hadoop-daemon.sh start [namenode | secondarynamenode |
datanode | jobtracker | tasktracker]
or
update conf/masters
master

update conf/slaves
master
slaves

<in all machines>
update conf/core-site.xml
---------
<property>
  <name>fs.default.name</name>
  <value>hdfs://master:54310</value>
  <description>The name of the default file system.  A URI whose
  scheme and authority determine the FileSystem implementation.  The
  uri's scheme determines the config property (fs.SCHEME.impl) naming
  the FileSystem implementation class.  The uri's authority is used to
  determine the host, port, etc. for a filesystem.</description>
</property>

----------
update conf/mapred-site.xml
----------
<property>
  <name>mapred.job.tracker</name>
  <value>master:54311</value>
  <description>The host and port that the MapReduce job tracker runs
  at.  If "local", then jobs are run in-process as a single map
  and reduce task.
  </description>
</property>
----------
update conf/hdfs-site.xml
----------
<property>
  <name>dfs.replication</name>
  <value>2</value>
  <description>Default block replication.
  The actual number of replications can be specified when the file is created.
```

```
  The default is used if replication is not specified in create time.
  </description>
</property>
----------

Additional Settings done in conf/mapred-site.xml
----------
“mapred.local.dir“
    Determines where temporary MapReduce data is written. It also may be a list of
directories.
“mapred.map.tasks“
    As a rule of thumb, use 10x the number of slaves (i.e., number of
TaskTrackers).
“mapred.reduce.tasks“
    As a rule of thumb, use num_tasktrackers * num_reduce_slots_per_tasktracker *
0.99. If num_tasktrackers is small (as in the case of this tutorial), use
(num_tasktrackers - 1) * num_reduce_slots_per_tasktracker.

<in master node>
hduser@master:/usr/local/hadoop$ bin/hadoop namenode -format

hduser@master:/usr/local/hadoop$ bin/start-dfs.sh
hduser@master:/usr/local/hadoop$ jps

<in slave node>
hduser@slave:/usr/local/hadoop$ jps


<in master node>
hduser@master:/usr/local/hadoop$ bin/start-mapred.sh

hduser@master:/usr/local/hadoop$ jps

<in slave node>
hduser@slave:/usr/local/hadoop$ jps
```

Result

Hadoop Multi Node cluster is installed successfully.