Hadoop Installation & Configuration

Preparation for Hadoop Installation:

Resources Details:

Out of the following hosts, we will use only selected two for our cluster configuration.

SN	Hostname	FQDN	IP
1	bdrenfdludcf01	bdrenfdludcf01.dle.asiaconnect.bdren.net.bd	103.28.121.5
2	bdrenfdludcf02	bdrenfdludcf02.dle.asiaconnect.bdren.net.bd	103.28.121.7
3	bdrenfdludcf03	bdrenfdludcf03.dle.asiaconnect.bdren.net.bd	103.28.121.30
4	bdrenfdludcf04	bdrenfdludcf04.dle.asiaconnect.bdren.net.bd	103.28.121.67
5	bdrenfdludcf05	bdrenfdludcf05.dle.asiaconnect.bdren.net.bd	103.28.121.34
6	bdrenfdludcf06	bdrenfdludcf06.dle.asiaconnect.bdren.net.bd	103.28.121.66

Configure Hosts

Login into both hosts as root user and add all hosts in /etc/hosts file. Other hosts are optional.

 # vim /etc/hosts

103.28.121.5 bdrenfdludcf01 bdrenfdludcf01.dle.asiaconnect.bdren.net.bd

103.28.121.7 bdrenfdludcf02 bdrenfdludcf02.dle.asiaconnect.bdren.net.bd

103.28.121.30 bdrenfdludcf03 bdrenfdludcf03.dle.asiaconnect.bdren.net.bd

103.28.121.67 bdrenfdludcf04 bdrenfdludcf04.dle.asiaconnect.bdren.net.bd

103.28.121.34 bdrenfdludcf05 bdrenfdludcf05.dle.asiaconnect.bdren.net.bd

103.28.121.66 bdrenfdludcf06 bdrenfdludcf06.dle.asiaconnect.bdren.net.bd

Prefer if we login into all the machines using Putty or Any SSH Client.

IP address check in your own hosts

#ip addr show

1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

valid_lft forever preferred_lft forever

inet6 ::1/128 scope host

valid_lft forever preferred_lft forever

2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000

link/ether 08:00:27:b0:fe:53 brd ff:ff:ff:ff:ff:ff

*inet **192.168.0.104**/24 brd 192.168.0.255 scope global dynamic enp0s3*

valid_lft 6925sec preferred_lft 6925sec

inet6 fe80::7aed:2a0d:40d:bc24/64 scope link

valid_lft forever preferred_lft forever

CPU Check

 # more /proc/cpuinfo | grep 'core id' | wc -l

The count will show you no of CPU.

Memory Check:

 # more /proc/meminfo | grep -i 'Mem'

Please see the MemTotal

Set All the Host Names (If require)

To set all the host names on a system, enter the following command as root:

***# hostnamectl set-hostname* bdrenfdludcf01**

***# hostnamectl set-hostname* bdrenfdludcf02**

Create a Repository for installing any extended Linux Package using yum:

Create a new file with below contents with extension of ".repo" using root user in both hosts.

[root@ bdrenfdludcf01/root]# cd /etc/yum.repos.d

[root@ bdrenfdludcf01yum.repos.d]# vim bdrenfdludcfrhelepl7.repo

[rhel7repo]

gpgcheck = 0

enabled = 1

baseurl = https://dl.fedoraproject.org/pub/epel/7/x86_64/

name = bdrenfdludcfrhelepl7-repo-epl

[root@nakparsdev-2-vm-01 yum.repos.d]#

Important Note: If you find any **certification problem (peer certification expired)** the temporary solution is to use http rather than https in baseurl setting above.

 # yum clean all

[root@ bdrenfdludcf01yum.repos.d]# yum clean all

Test the new yum repository by installing new package.

[root@ bdrenfdludcf01yum.repos.d]# yum install telnet -y

You can study on more about yum commands from following URL:

http://yum.baseurl.org/wiki/YumCommands

Do the configuration for all hosts

OR

Copy the newly created repo file **/etc/yum.repos.d/datasciencelabrhelepl17.repo** to other hosts.

#scp /etc/yum.repos.d/ bdrenfdludcfrhelepl7.repo bdrenfdludcf02:/etc/yum.repos.d/

Test yum over the new hosts as well

Time Zone Setting (If require) using root user in all hosts

#timedatectl

#timedatectl list-timezones | grep Dhaka

Setting my own time zone using following command.

#timedatectl set-timezone Asia/Dhaka

#timedatectl set-time 13:38:00

Configure NTP (Network Time Protocol):

#yum install ntp

The above installation might not necessary if you have already installed NTP.

 # vi /etc/ntp.conf

See all the details once & no need to change anything.

#service ntpd status

#service ntpd start

Configure NTP so that it can come up even after host restart.

#chkconfig ntpd on

Checking NTP servers listing.

#ntpq -p

Stop iptables in Redhat 7 for all hosts lab purpose (N/A in production):

[root@bdrenfdludcf01~]#systemctl status firewalld

[root@bdrenfdludcf01~]#service firewalld stop

[root@bdrenfdludcf01~]#chkconfig firewalld off

Stop SELINUX (N/A in Production):

#vim /etc/selinux/config

 # Make sure SELINUX=disabled

SELINUX=disabled

Now To take effect immediately, we need to restart they system or you can run following command.

#setenforce 0

Create application users in all machines:

#adduser -g wheel -s /bin/bash -d /home/hadoop hadoop

#passwd hadoop

Note: Set password as you prefer.

Assign Administrative Role to hadoop User:

Adding a dedicated user for Hadoop in all hosts.

Uncomment the below line in /etc/sudoers:

#vim /etc/sudoers

Same thing without a password

%wheel ALL=(ALL) NOPASSWD: ALL

Configure password less SSH hadoop User:

Login in the hosts using hadoop user and follow the steps:

Generate Key by running the following command.

If command asks for input, please press enter

 # ssh-keygen

Then copy the key for his own host & for the other hosts.

Please mind that the port option for you may be '-p 22'

 # ssh-copy-id -p 2200 -i $HOME/.ssh/id_rsa.pub hadoop@bdrenfdludcf01

 # ssh-copy-id -p 2200 -i $HOME/.ssh/id_rsa.pub hadoop@bdrenfdludcf02

 # ssh -p 2200 hadoop@bdrenfdludcf01 #Test whether you can login password less

 # ssh -p 2200 hadoop@bdrenfdludcf02 #Test whether you can login password less

Please do it for the other host as follows:

Generate Key by running the following command.

If command asks for input, please press enter

 # ssh-keygen

Then copy the key for his own host & for the other hosts.

Please mind that the port option for you may be '-p 22'

 # ssh-copy-id -p 2200 -i $HOME/.ssh/id_rsa.pub hadoop@bdrenfdludcf02

 # ssh-copy-id -p 2200 -i $HOME/.ssh/id_rsa.pub hadoop@bdrenfdludcf01

 # ssh -p 2200 hadoop@bdrenfdludcf02 #Test whether you can login password less

 # ssh -p 2200 hadoop@bdrenfdludcf02 #Test whether you can login password less

Then better to secure your keys In all hosts as hadoop user:

#cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

#chmod 0600 ~/.ssh/authorized_keys

Java Installation or checking:

We can test to see if Java installed correctly with the following command using hadoop user.

allnodes$ java -version

java version "1.7.0_79"

OpenJDK Runtime Environment (IcedTea 2.5.5) (7u79–2.5.5–0ubuntu0.14.04.2)

OpenJDK 64-Bit Server VM (build 24.79-b02, mixed mode)

Or we need to download and install java version in both hosts as given in shared file.

Step 1

Upload jdk-7u71-linux-x64.tar.gz will be downloaded into your system for example in /root/download using root credential.

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.

#sudo cd /root/downloads/

#sudo ls jdk-7u71-linux-x64.gz

#sudo tar zxf jdk-7u71-linux-x64.gz

#sudo ls -l jdk1.7.0_71

Step 3

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

#sudo mv jdk1.7.0_71 /usr/local/

Step 4

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

#vim ~/.bashrc

export JAVA_HOME=/usr/local/jdk1.7.0_79
export PATH=$PATH:$JAVA_HOME/bin

Now apply all the changes into the current running system.

$ source ~/.bashrc
$ vim ~/.bashrc

Step 5

Use the following commands to configure java alternatives:

#sudo alternatives --install "/usr/bin/java" "java" "/usr/local/jdk1.7.0_79/bin/java" 1
#sudo alternatives --set java /usr/local/jdk1.7.0_79/bin/java

Now verify the java -version command from the terminal as explained above.

#java -version

Main Hadoop Installation, Configuration & Administration:

Download Binaries and Install

Assumptions is that we will have already downloaded Hadoop binaries in /root/downloads using root user in all hosts. Follow below steps in all hosts using hadoop user.

allnodes# cd /root/downloads

*allnodes# sudo tar zxvf /root/downloads/hadoop-**

allnodes# sudo mv hadoop-2.7.1 hadoop

allnodes# sudo mv hadoop /usr/local/

Set Hadoop-related environment variables

Now we need to add hadoop and Java environment variables to ~/.bashrc and source them to the current shell session.

 # su - hadoop

~/.bashrc:

export JAVA_HOME=/usr/local/jdk1.7.0_79

export PATH=$PATH:$JAVA_HOME/bin

export HADOOP_HOME=/usr/local/hadoop

export PATH=$PATH:$HADOOP_HOME/bin

export HADOOP_CONF_DIR=/usr/local/hadoop/etc/hadoop

Then load these environment variables by sourcing the profile

allnodes$. ~/.bashrc

Now you can test the Hadoop command is located by below command.

allnodes$ which hadoop

Hadoop Configurations

We are going to change a few of the configurations in the Hadoop directory defined now by HADOOP_CONF_DIR environment variable. All the current configuration changes will be applied to the NameNode and all the DataNodes. After these changes, we will apply configurations specific to the NameNode and DataNodes.

Here are the following files to focus on:

•$HADOOP_CONF_DIR/hadoop-env.sh

•$HADOOP_CONF_DIR/core-site.xml

•$HADOOP_CONF_DIR/yarn-site.xml

•$HADOOP_CONF_DIR/mapred-site.xml (This file currently does not exist in the default Hadoop installation, but a template is available. We'll make a copy of the template and rename it to mapred-site.xml)

Common Hadoop Configurations on all Nodes

Let's start with $HADOOP_CONF_DIR/hadoop-env.sh. Currently only root users can edit files in the Hadoop directory, but we'll change this after all configurations have been applied. To edit the configurations files, you can simply add a sudo before the text editor of your choice, for example

allnodes$ sudo vim $HADOOP_CONF_DIR/hadoop-env.sh

The only thing that needs changing is the location of JAVA_HOME in the file. Simply replace${JAVA_HOME} with /usr/local/jdk1.7.0_79 which is where Java was just previously installed.

Important Note: If your SSH port is 22, then you can skip configuring SSH customer port section marked as different color below.

#sudo vim $HADOOP_CONF_DIR/hadoop-env.sh

 # The java implementation to use & SSH custom port.

```
export JAVA_HOME=/usr/local/jdk1.7.0_79
export HADOOP_SSH_OPTS="-p 2200"
```

The next file to modify is the $HADOOP_CONF_DIR/core-site.xml. Here we will declare the default Hadoop file system. The default configuration is set to the localhost, but here we will want to specify the NameNode's public DNS on port 9000. Scroll down in the xml file to find the configurations tag and be sure to change the file to look like the following

```
#sudo vim $HADOOP_CONF_DIR/core-site.xml
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://bdrenfdludcf01:9000</value>
  </property>
</configuration>
```

The next file to modify is the $HADOOP_CONF_DIR/yarn-site.xml. Scroll down in the xml file to find the configurations tag and be sure to change the file to look like the following

```
#sudo vim $HADOOP_CONF_DIR/yarn-site.xml
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value>bdrenfdludcf01</value>
```

```
  </property>
</configuration>
```

The last configuration file to change is the $HADOOP_CONF_DIR/mapred-site.xml. We will first need to make a copy of the template and rename it.

allnodes$ sudo cp $HADOOP_CONF_DIR/mapred-site.xml.template $HADOOP_CONF_DIR/mapred-site.xml

Go down in the xml file to find the configurations tag and be sure to change the file to look like the following

#sudo vim $HADOOP_CONF_DIR/mapred-site.xml

```
<configuration>
  <property>
    <name>mapreduce.jobtracker.address</name>
    <value>bdrenfdludcf01:54311</value>
  </property>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

NameNode Specific Configurations

We have done all the common configurations are complete, we'll finish up the NameNode specific configurations. On the NameNode, all that remains are the following:

•Adding hosts to /etc/hosts

•Modifying the configurations in $HADOOP_CONF_DIR/hdfs-site.xml

•Defining the Hadoop master in $HADOOP_CONF_DIR/masters

•Defining the Hadoop slaves in $HADOOP_CONF_DIR/slaves

We have already put entry in all hosts/etc/hosts file. So we can skip now.

So we can now modify the $HADOOP_CONF_DIR/hdfs-site.xml file to specify the replication factor along with where the NameNode data will reside. For this setup, we will specify a **replication factor** of **2** for each data block in HDFS.

Go down in the xml file to find the configurations tag and be sure to change the file to look like the following

#sudo vim $HADOOP_CONF_DIR/hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>2</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:///usr/local/hadoop/hadoop_data/hdfs/namenode</value>
  </property>
</configuration>
```

The current path where data on the NameNode will reside does not exist, so we'll need to make this before starting HDFS.

namenode$ sudo mkdir -p $HADOOP_HOME/hadoop_data/hdfs/namenode

In your case you may create a file location under /home as well or you can follow this one for simplicity. Then the HDFS size will be limited to / partition size among all hosts configured.

Next we'll need to add a masters file to the $HADOOP_CONF_DIR directory

namenode$ sudo touch $HADOOP_CONF_DIR/masters

Then insert the NameNode's hostname in that file

#sudo vim $HADOOP_CONF_DIR/masters

```
bdrenfdludcf01
```

We will also need to modify the slaves file in the $HADOOP_CONF_DIR directory to the following. By default localhost is present, but we can remove this & add

both planned data nodes host name. As we will be installing both name node and data node in first host, so we will have to add both hosts here.

#sudo vim $HADOOP_CONF_DIR/slaves

```
bdrenfdludcf01
bdrenfdludcf02
```

Now that all configurations are set on the NameNode, we will change the ownership of the$HADOOP_HOME directory to the user hadoop.

namenode$ sudo chown hadoop:wheel -R $HADOOP_HOME

Important Note: Please do not run any command in namenode from sudo after this.

DataNode Specific Configurations

Let's now move onto the final configurations for the DataNodes. We will need to first SSH into each DataNode and only configure the $HADOOP_CONF_DIR/hdfs-site.xml file

Scroll down in the xml file to find the configurations tag and be sure to change the file to look like the following.

#sudo vim $HADOOP_CONF_DIR/hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>2</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:///usr/local/hadoop/hadoop_data/hdfs/datanode</value>
  </property>
</configuration>
```

As we are going to create data node for both hosts, hence we would need to ensure dfs.datanode.data.dir in both machines.

Just like on the NameNode, we will need to create the directory specified in the$HADOOP_CONF_DIR/hdfs-site.xml file.

datanodes$ sudo mkdir -p $HADOOP_HOME/hadoop_data/hdfs/datanode

Now that all configurations are set on the DataNode, we will change the ownership of the$HADOOP_HOME directory to the hadoop user

datanodes$ sudo chown -R hadoop $HADOOP_HOME

Important Note: Please do not run any command in namenode from sudo after this.

Start Hadoop Cluster

We can now start up HDFS from the Namenode as hadoop user by first formatting it and then starting HDFS. An important thing to notes is that every time the NameNode is formatted, all of the data previously on it is lost. This step is missing in our first configuration attempt during class.

namenode$ hdfs namenode -format

namenode$ $HADOOP_HOME/sbin/start-dfs.sh

When asked "The authenticity of host 'Some Node' can't be established. Are you sure you want to continue connecting (yes/no)?" type yes and press enter. You may need to do this several times—keep typing yes, then enter, even if there is no new prompt, since it's the first time for Hadoop to log into each of the datanodes.

You can go to http:// bdrenfdludcf01:50070 in your browser to check if all datanodes are online. If the webUI does not display, you need to troubleshoot.

Now let's start up YARN as well as the MapReduce JobHistory Server.

namenode$ $HADOOP_HOME/sbin/start-yarn.sh

namenode$ $HADOOP_HOME/sbin/mr-jobhistory-daemon.sh start historyserver

You can check to make sure all Java processes are running with the jps command on the NameNode and DataNodes (your process ids will be different though).

namenode$ jps

21817 JobHistoryServer

21853 Jps

20992 DataNode

21376 SecondaryNameNode

21540 ResourceManager

21157 NameNode

datanodes$ jps

20936 NodeManager

20792 DataNode

21036 Jps

Working with HDFS

You're now ready to start working with HDFS by SSH'ing to the NameNode. The most common commands are very similar to normal Linux File System commands, except that they are preceded by hdfs dfs. Below are some common commands and a few examples to get used to HDFS.

Common HDFS Commands

List all files and folder in directoryhdfs dfs -ls <folder name>Make a directory on HDFShdfs dfs -mkdir <folder name>Copy a file from the local machine (namenode) into HDFShdfs dfs -copyFromLocal <local folder or file name>Delete a file on HDFShdfs dfs -rm <file name>Delete a directory on HDFShdfs dfs -rmdir <folder name>

HDFS Examples

create local dummy file to place on HDFS

namenode$ echo "Hello this will be my first distributed and fault-tolerant data set\!"" | cat >> my_file.txt

list directories from top level of HDFS

namenode$ hdfs dfs -ls /

namenode$ hadoop fs -ls /

This should display nothing but a temp directory

create /user directory on HDFS

namenode$ hdfs dfs -mkdir /user

namenode$ hdfs dfs -ls /

copy local file a few times onto HDFS

namenode$ hdfs dfs -copyFromLocal ~/my_file.txt /user

*namenode$ hadoop fs -put *_HO_201610181700* /parsdev/*

namenode$ hdfs dfs -copyFromLocal ~/my_file.txt /user/my_file2.txt

namenode$ hdfs dfs -copyFromLocal ~/my_file.txt /user/my_file3.txt

list files in /user directory

namenode$ hdfs dfs -ls /user

clear all data and folders on HDFS

*namenode$ hdfs dfs -rm /user/my_file**

namenode$ hdfs dfs -rmdir /user